Macintosh version

A User's Guide to CUFIM-PRO, the

Community and Urban Forest Inventory and Management Program

By

Norman H. Pillsbury

April 2015

Technical Report No. 14

Urban Forest Ecosystems Institute
California Polytechnic State University
San Luis Obispo

Quick Start Guide

Alert!!!! CUFIM-PRO is a Mac-only version.

Alert!!!! CUFIM-PRO is an .xlsm file (it includes macros). When opening an .xlsm file, Excel alerts the user with this standard message. Be sure to click on *Enable Macros*.

1. Select an appropriate computer and monitor size (see next column). Close all applications.

2. Download the CUFIM-PRO User's Guide and the *CUFIM-PRO folder* to your desktop. Use web site (http://ufei.calpoly.edu/urbanwood/links.lasso; look for CUFIM-PRO (MS Excel Program) at http://www.ufei.org/files/ufeipubs/CUFIM_Report.pdf).

   Alert!!!! Read through the User's Guide before starting with CUFIM-PRO. Understanding how the program works and the various options are crucial to proper use.

3. Launch CUFIM-PRO with Microsoft® Excel®.

4. Make sure that Preferences/Calculation is set to automatic.

5. Only enter data in yellow colored cells

2. OK to click on:

   "Go to…" labels to move from sheet to sheet, or use tabs at the bottom.

   Any action box that starts with "Click here to…"

   "Print…" boxes

   List action boxes, Combo Boxes and Check Boxes

   Colored 3-D action boxes or rectangles.

Computer Recommendation

The best advice is always to run CUFIM-PRO on the fastest Mac possible with the largest monitor.

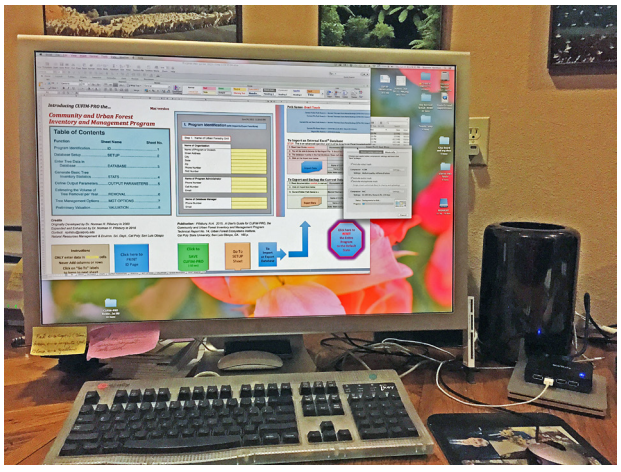
Number One Recommendation: CUFIM-PRO was developed on a 64-bit Late 2013 Mac Pro, 3 GHz, 8-core Intel Xeon E5 with AMD Fire Pro D700 graphics card, running OS 10.9.5 and a 30" monitor. The Excel® software tested is Microsoft® Excel® for Mac 2011, v. 14.4.6 (14106).

2. MacPro mid-2010. This is a 2 x 2.93 Ghz 6-core Intel Xeon Mac with 16 GB of 1333 MHz DDR3 memory running OS 10.7.5. The Excel version tested was 14.4.8 (150116), Mac 2011.

For more information, visit Part I, Section D.

Late 2013 Mac Pro

MacPro mid-2010



Late 2013 Mac Pro and 30" monitor.

Table of Contents

Acknowledgements

In the late 1990s, the California Department of Forestry and Fire Protection and the Urban Forest Ecosystems Institute (UFEI), Natural Resources Management Department at Cal Poly State University, San Luis Obispo developed a partnership to conduct urban volume and biomass studies in California. Key to this partnership was Mr. Eric Oldar, Urban Forester, California Department of Forestry and Fire Protection, Riverside, CA.

We also worked with several urban foresters, including: Mr. Robert Reid, Urban Forester, Monterey; Mr. Todd Martin, City Arborist, San Luis Obispo; Mr. Dave Pendergraft, Arborist, Visalia; Mr. Michael Breidert, Santa Clarita; Mr. Omer Davis, Santa Clarita; Ms. Cindy McCall, Lompoc; Mr. Chris Bosa, Chico.

Department faculty involved were Drs. Richard Thompson, Samantha J. Gill, and myself, UFEI, Cal Poly State University, San Luis Obispo;

In 2000-2003, this project was supported by the California Department of Forestry and Fire Protection, the Urban Forest Ecosystems Institute, Cal Poly State University, San Luis Obispo, the California Agricultural Research Initiative, and through a McIntire-Stennis Forestry Research grant.

In 2014-2015, this project was supported by the author, now retired from Cal Poly.

Preface

The Community and Urban Forest Inventory and Management program (CUFIM-PRO) is one more step in answer to the call for improved management and sustainability of California's urban forests.

This report presents an enhanced Excel-based computer program that can be used to setup and maintain a tree inventory and database, and to evaluate the urban forest in quantitative terms including volume and value. Further, it can import an existing database of tree inventory data.

It is the hope of the author that this effort will help urban communities take the next step toward sustainability of the urban forest resource.

This version is a Mac-only program.

Norman H. Pillsbury, Ph.D., Professor Emeritus
Registered Professional Forester, License Number 1807
Natural Resources Management and Environmental Sciences Department
California Polytechnic State University, San Luis Obispo, California 93407

Macintosh version

A User's Guide to CUFIM-PRO, the…

Community and Urban Forest Inventory and Management Program

Part I.

How to Design and Implement an Urban Forest Inventory

Part I. How to Design and Implement an Urban Forest Inventory

A. Introduction

The purpose of this program and user's guide is to help communities manage their urban forests specifically in relation to the potential use of woody biomass rather than the more traditional and costly practice of disposal.

This study builds on two previous studies that provided the foundation and direction for this effort. *The Elements of Sustainability in Urban Forestry* by Richard Thompson, Norman Pillsbury and Richard Hanna (1994) examined urban forestry in California and identified the elements necessary for sustainability to occur. This study also motivated the development of urban tree volume equations (*Tree Volume Equations for Fifteen Urban Species in California* by Norman Pillsbury, Jeffrey Reimer and Richard Thompson, 1998) where detailed estimates of the biomass potential for various urban species is now possible.

In 2003, Dr. Samantha Gill and myself published *A Users Guide for CUFIM, the Community and Urban Forest Inventory and Management Program*, about an Excel-based computer program I wrote that allowed urban foresters control over their tree inventory. It also allowed unprecedented options to determine volume of anticipated tree removals, and estimates of their dollar value.

In 2014-2015, I completely revamped, enhanced and significantly expanded the capability of the program, now called CUFIM-PRO. Nearly all of the original code was rewritten and when fully populated it manages over 1.2 million cells. This guide is divided into three parts. First is an introduction to the program and it's methods and procedures, and a section on *How to Design and Implement an Urban Forest Inventory*. The next two parts are, *Tree Inventory and Database Setup and Maintenance*, and *Planning for Tree Removal Volume and Value Calculations*.

In Part Two, *Tree Inventory and Database Setup and Maintenance*, the program allows users to store and maintain up to 500 tree species and up to 50,000 tree records, although that number could be doubled or tripled as discussed in the text. Each record contains the following fields: Tree sequence number, tree record number, species code, removal status, species name, species group, tree diameter at breast height (dbh), tree height, tree volume including a breakdown of volume by branch diameter size, tree location, date of measurement, and eight user-defined variables for easy input by the user.

Each species that is added to the database must be assigned a species group code, a code that identifies that tree with trees with known volume equations. It is primarily used to calculate tree volume information. CUFIM-PRO comes with 19 built-in species groups, and allows for a total of 50 groups by the user. If new species groups are to be entered, their local or standard volume equation coefficients must also be entered as described in this report.

The documentation and notes provide assistance to the user including suggestions for setting up the database and the recommended precision of measurement units. A number of examples are included to illustrate main points.

An Import template and instructions allow for importing large database entries, while three separate built-in functions are available for ease in adding, changing or deleting fewer tree records. Basic tree statistics such as number of species, size of database, averages of dbh, height, total volume and volume by diameter class are available as well as graphics to visualize the urban forest composition.

In Part Three, *Planning for Tree Removal Volume and Value Calculations*, several spreadsheets are devoted to estimating tree volume and value from trees that would be removed from the forest each year. Options include limiting the range of diameters, as well as the species of tree that would be included and/or excluded in volume calculations.

Several kinds of tables can be printed that provide hard copy of the database, and summarize the number of trees and volume by diameter class. Biomass value can also be assessed through three different methods.

The program is designed for beginners, but as skill levels increase, a number of advanced options are also available.

In the future, potentially, communities can market their biomass for wood products and show an income from the woody resource rather than only a cost for maintenance and disposal.

B. Procedures and Methods

During the 2003 study, the authors canvassed 20 communities in California to determine the type of street tree databases that were currently being used in order to understand the benefits and limitations of these programs. Four communities allowed us access to their database so we could examine how they were used and what difficulties they experienced.

The results varied from community to community. Some communities have contracted with consultants to manage their entire street tree inventory. This allows the community to focus their time on other needs, and seems to have provided some level of assurance that consistency and accuracy would be maintained over time. There is also the sense that periodic updates would continue. A disadvantage of this approach is the loss of contact with the process itself. Very few knew much about the system, and felt that making changes to the inventory procedures that were initially agreed upon would not be easy, and therefore wouldn't happen.

Other communities have maintained their own database and developed a process that is tailored to their needs, in the best way possible. None of the street tree inventory programs we learned about were designed to address volume and wood value issues relating to inventory. This means that regardless of their dedication to this process, such information cannot be obtained. Communities in general do not understand the difficulty of keeping an inventory up to date, especially given that personnel often change. The last person in charge may have kept the task high priority while the next person did not.

Some urban foresters expressed frustration because the information that is generated from existing inventory programs hasn't been helpful in convincing others in their organization of the value of the forest. Supervisors and administrators often view inventory work as low priority activities that can be farmed out, postponed for many years, or dropped altogether.

These issues complicate the goals of urban forest management. If inventory programs don't provide desirable information for decision makers, and if people aren't trained to run the programs, especially when new employees take over, then a different approach is needed if urban wood is to be utilized and valued.

In 2003, the author(s) tried to address the limitations discussed above through the development of a Microsoft® Excel® based computer program titled CUFIM: "Community and Urban Forest Inventory and Management" or, now the 2015 version, CUFIM-PRO. We recognize that no single approach will solve all of these issues. However, our objectives in developing this program focus on the following issues.

1. The program must be designed so that semi-computer literate people could use it without extensive training. This means that one does not have to be a computer programmer to run the program. Also, as work force changes occur in an organization, the program must be "simple" enough so that new hands can pick up where others left off.
2. On-the-other-hand, the program must be sophisticated enough to address inventory from the standpoint of quantity of wood and value of the resource. To date, this is what has been missing and why street tree inventory programs are not taken seriously in the urban forestry budgeting process.
3. The program must be flexible enough so that existing data parameters can be changed, and new parameters can be added as needs arise.

As alluded earlier, it would require a "super program" and "super people" to achieve this goal. Yet, there is room for significant improvement in the way inventories are managed, and the computer program presented here takes major steps in that direction. And, as with any program, there are limitations with this one as well.

The choice to select Microsoft Excel® was made for several reasons.

1. Excel is the most universally understood database program in the world. While it technically is not a true database program, it does include most of the major elements needed in a database program and was programmed to be effectively for this purpose.
2. Nearly all, if not all, graduates from universities have worked with Excel spreadsheets. This means they already have some level

of expertise. While it may be minimal for some, it will be considerable for others.

3. Learning how to use the spreadsheets developed for this program is not a difficult task for those with some spreadsheet background. If personal changes occur in a community, new people will have an easier time of picking up the system.
4. As time progresses, the expertise levels of spreadsheet users will increase, not decrease. Therefore, over time, the use of a program developed in Excel will become easier and users will become more proficient.
5. Users can quickly learn how to import and backup the database as built-in Import and Export routines are available in the 2015 version.

There are also some limitations with the use of Excel.

1. As mentioned earlier, it is not a true database program. One impact of this is that the speed of the program will decrease as the database increases. This is not significant on fast computers, but for older, slower computers, it would be problematic. Also, some routines may take a minute or more to perform even with fast computers. This is a minor issue, and on the plus side, every new generation of computers sports a dramatic increase in speed. In time, waiting even a minute won't occur.
2. The size of the database is more limited through Excel. For example, the program presented here allows a maximum of 50,000 trees. This limit was selected because larger numbers could slow the program below the willing attention span of many users. This size will be adequate for small- and medium-sized communities, however, large cities with 100,000-200,000 trees would break their database into several geographic units to benefit from the program. An outline to achieve that is included in the appendix.
3. This program is not protected in the same way that computer applications are protected. While this is possible, doing so has disadvantages. By leaving the source code open, users with higher levels of expertise, can make changes to meet changing needs. This flexibility can be very valuable to a community.

   However, there is the difficulty of protecting the database from unwanted entries and user errors. The disadvantage is that important information could be lost, or parts of the program would not work, if the user does not adhere to the rules and procedures. All is not lost, however, because of backups, and failing that, there are many people around that could fix an Excel problem. Colleges and businesses are a good source of spreadsheet expertise.

The reader can see that we have tried to balance the need for ease and flexibility of use with the need for a program that is capable of providing quantitative data from a substantial database.

C. An Outline for Establishing an Urban Forest Inventory Program for Your City

CUFIM-PRO fits into the larger picture of preparing or updating a comprehensive urban forest inventory. This section of the user's guide considers the development of a program for inventorying and maintaining a database of trees and associated attributes for a community. Without some guidance and help this can be a daunting endeavor. Prior to beginning, it is assumed that computing resources are available and the city has provided a budget for field crew labor and necessary equipment.

The steps to a successful tree inventory are outlined below along with occasional comments and recommendations. Different levels of data are needed for different management needs. The objective of acquiring and maintaining a city tree inventory will influence the amount and type of data needed.

1. Designing the Tree Inventory -- What Will It Include?

The following is organized from the lowest level of needs/management to the highest level. Each higher level of data needs includes everything in the previous levels.

a. First level data (what are the minimum requirements?)

1) Species name - normally use scientific name, but in an auxiliary field, the common name may also be included.

2) Species codes - each species is given a unique code number, usually consecutive from 1 - n.

3) Tree record number (each tree in the database is given a number, 1-n).

4) Species Group code - a number that identifies each tree with trees with known volume equations. It is primarily used to calculate tree volume information. For example, CUFIM-PRO includes volume equations for coast live oak but not canyon live oak. Coast live oak would be the key species, say, species group code #1, and canyon live oak would be coded "1," too, as it is similar in size, shape and volume characteristics.

   CUFIM-PRO includes volume equations for 19 urban species that make up the Species Group core. Species without known equations are coded to one of the 19, the one that "looks" most similar from a tree volume standpoint.

5) Diameter at breast height (dbh), measured with a diameter tape to a precision of 0.1" as that is the standard and that precision can easily be achieved. This precision also allows for other options discussed later. Use of diameter classes is <u>not recommended</u>, however, if diameter classes must be used, use 0.5" or 1" classes, and they must all be of the exact same interval.

6) Location of tree (street address, lat/long, state plane coordinates, etc.) – may be obtained in several ways. For lower level management objectives an address will suffice. However for higher level management needs, it may be necessary to obtain data that can be accessed through a GIS program such as ARC-INFO.

b. Second level data needs

7) Tree height – height is necessary for better estimates of volume. Ideally measured with precision of ±1-2 ft, but no greater than ±5'. CUFIM-PRO will work if only dbh is measured, but including tree height is strongly recommended. Height is also useful if overhead powerlines are present.

8) Date tree was measured - the date should be formatted as mmm ddd, yyyy. This information is useful for a variety of purposes such as re-inventory, growth estimates, and for recently planted trees, tree age.

9) Tree health – necessary to determine when trees will need to be removed. This can be in broad categories such as 1) tree is dead; 2) tree likely to die within one year; 3) tree likely to die within 5 years; 4) tree likely to die in 10 years (or within the return interval of the inventory); or 5) tree healthy.

10) Age (year planted) – Age will be helpful for assessing the number (and thus volume) of trees that may need to be removed within in the same year (or few years). Age is absolutely necessary for long term planning of removal schedules.

c. Third level data needs

11) GIS locations of trees – May be lat/long or state plane coordinates. These can be obtained by digitizing or with a mobile GPS unit.

12) Sidewalk damage (Low, Medium, High)

13) Distance to street (in feet)

14) Crown information (height to crown base or crown radius).

15) Right of way information

16) Historic tree status

CUFIM-PRO provides fields for the following variables. These can not be altered.

1) Species name
2) Species code
3) Tree record number
4) Species Group code
5) Dbh
6) Tree height
7) Total volume (calculated by equations)
8) Volume by branch size (segment size) (calculated by equations).

The program also includes 10 user defined variables (UDVs) with flexibility to alter the name and input format. Two are suggested, #9 and #10 below, but they could be changed by the user.

9) Location
10) Date measured
11) Variable 3
12) Variable 4
13) Variable 5
14) Variable 6
15) Variable 7
16) Variable 8
17) Variable 9
18) Variable 10

Analysis is provided for the first eight variables, however, no analysis is available for the UDV's as they are not known. Any analysis of these, other than sorting, would be done by the user separate from CUFIM-PRO. In that case, export the database, and make a copy of the file for analysis.

Before measuring UDV's, it should be clear how the data will be used. How will it be measured and in what units? Far too often variables are measured and either can't be used because of the measurement method or because no one really wants the information. Gathering tree data can be expensive and without a good rationale can be costly. Measurement for the sake of measurement should be avoided.

2. Collecting Inventory Data in the Field

a. Allocate ample time and funding for two two-person field crews to measure trees.

b. Develop a standard field data sheet, and test and refine it as needed before the inventory work begins.

c. Obtain necessary field equipment in advance for two crews plus extras in case of breakage:

1) Diameter tapes (D-tapes). Note: ALL trees between 0.5" dbh to the largest tree must be measured.

2) Height measuring equipment: clinometers are inexpensive; laser rangefinders (e.g., Impulse® 200 LR) are very expensive but save tremendous amounts of time over clinometers and they have the ability to calculate horizontal and vertical distances as well as height values. 100' cloth tapes will be needed if clinometers are used.

3) Letter authorizing field crew to make measurements, so citizens are informed of the tree measuring activities.

3. When to Use the Import Template to Input Inventory Data.

An Import Template is included in the CUFIM-PRO folder, pictured ahead in Figure 3. It serves several purposes, including:

a. Handling new or follow up inventory data when all trees are measured in the field, recorded on field forms and typed into the Import Template for importing into CUFIM-PRO. This scenario is the focus of the section ahead.

b. Importing an existing inventory from another program. That data would be downloaded and reformatted into CUFIM-PRO's Import Template for importing into CUFIM-PRO. Follow the instructions in the appendix, "*How to Import an Existing Database into CUFIM-PRO*."

c. Breaking an existing CUFIM-PRO database into two or more smaller files. See appendix on how to accomplish splitting a database.

4. Entering Data into the Import Template

An Import Template is included in the CUFIM-PRO folder, pictured ahead in Figure 3. As field data arrives at the office it should be entered into the *Import Template.xlsx* file as quickly as possible so that errors can be corrected before the field crew finishes.

a. Columns A, B, C, the green section: Enter the unique species list here, up to 500 entries allowed. Only one species name should appear in the list (column B) along with its Species Code, and Species Group Number. Information on how to assign Species Group numbers is found on the SETUP sheet, see Figure 12, ahead.

b. Columns D-T, the orange section are for the database values.

Column D: The sequence number must be sequential from 1 to n. Enter these last.

Important: CUFIM-PRO uses these numbers extensively and to be certain they are consecutive, the program numbers them and Tree Record number upon import from 1-n.

Column E, the Tree Record number starts with 1 and is numbered consecutively to the end.

Species code in column F is the same as column A (green section) except column F is for the potentially hundreds of species having the same code, while column A is just the list of species and their codes found in the database and it will only occur once.

Column G is species name. Usually this is the scientific name (also called botanical name).

Columns H and I are for the tree's dbh and height.

Columns K-T are for User Defined Variables discussed in Part II.

Important: Save the *Import Template.xlsx* file with a new name, like, *Import CityName.xlsx*, with your city in place of CityName.

5. Checking Data After Entry in Import Template

a. Employ part-time help to input data on the Import Template. Important: Use the provided *Import Template.xlsx* to organize the data properly as discussed here (also see Part II of the guide).

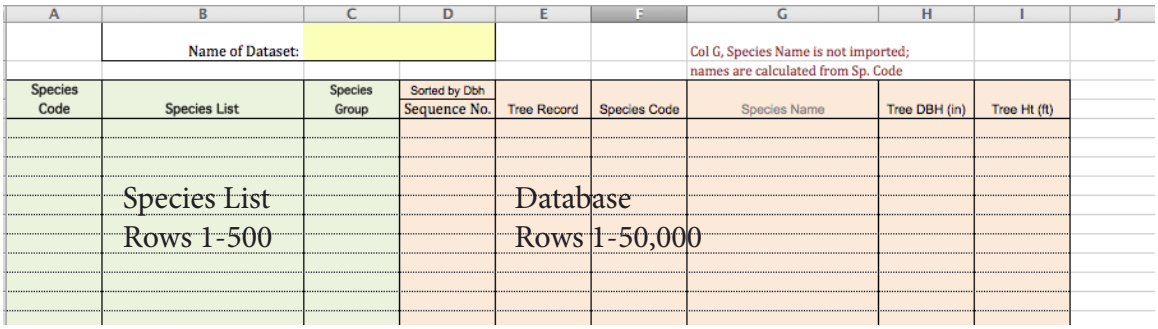
b. One employee reads values, another checks against original data sheets.

c. Examine the dbh vs ht relationships for values; are there any that seem unrealistic? Send crew back to field check or remeasure, if needed.

d. Check for negative numbers (there shouldn't be any).

e. Check for missing data. Send crew back to field to measure, if needed.

f. If your database has more than about 45,000 trees, the database should be split into two sections, e.g., *CUFIM-PRO A* and *CUFIM-PRO B* in two separate folders. See appendix for an approach to achieve this.



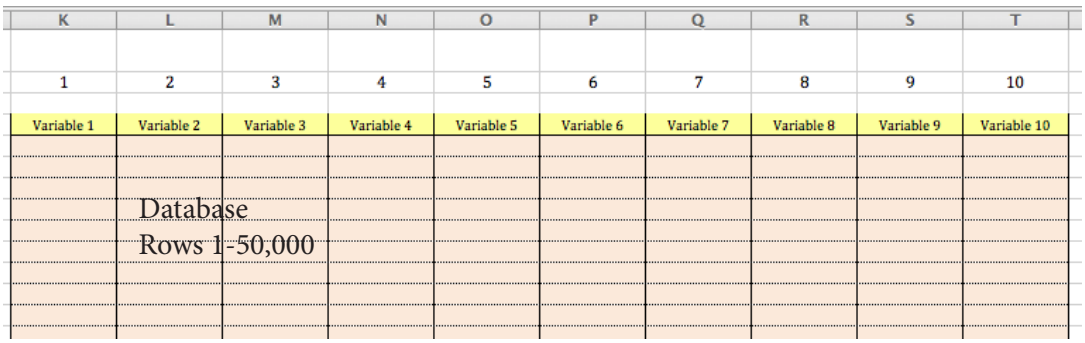


Figure 3 (from Part II). Import Template.xlsx

g. Review Import template to make sure data was entered without altering the form. Are the data in the correct columns and rows?

   Note, that ALL fields for each tree must have the correct information or CUFIM-PRO won't run properly. Be sure that text only is in text fields and numbers only are in cells intended for numbers.

h. Backup the *Import CityName.xlsx* as you proceed and again when ready for importing to CUFIM-PRO.

IMPORTANT

The Import Template must have valid data in ALL cells for CUFIM-PRO to calculate properly. One example is in the calculation of volumes on the STAT sheet. If the Species Codes are in error, the program stops calculating and returns to the original position. Review the Trouble Shooting section for more information.

To assist the user, a short program called *Import Checker* is included in the *CUFIM-PRO folder*.

When you believe the Import template is ready for Import, close all files, and open *Import Checker*. Click the run icon and it will tell you:

a. If blank cells are buried in the database (see sidebar information), and,

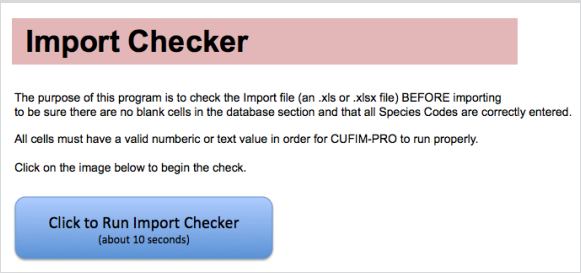
b. If invalid Species Codes are used in Col F (database section) as they have no match in Col A (Setup section). The problem may be that one or more species were left off the list in the SETUP section (green cells), or that they were incorrectly coded in the database section.

If either condition exist, CUFIM-PRO, in a new window, provides a Report showing the number of valid cells and/or the number of instances when a Species Code is invalid (see sidebar, bottom image). Print the Report and carefully review all data to find the discrepancy.

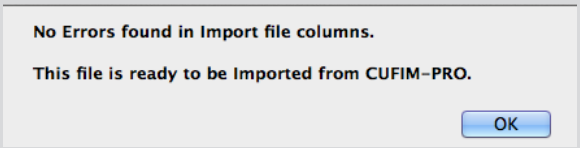
Run *Import Checker* again until all cells have valid data.

Import Checker is an .xlsm file (it includes macros). When opening an .xlsm file, Excel alerts the user with this standard message. Be sure to click on *Enable Macros*.

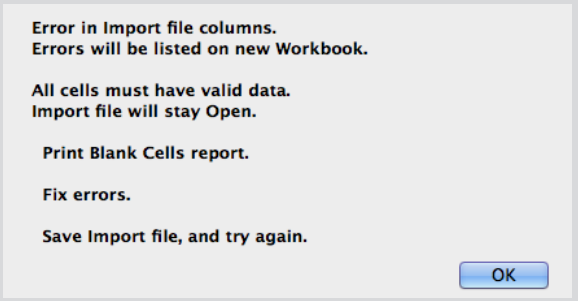
1. Click on the Run *Import Checker* icon

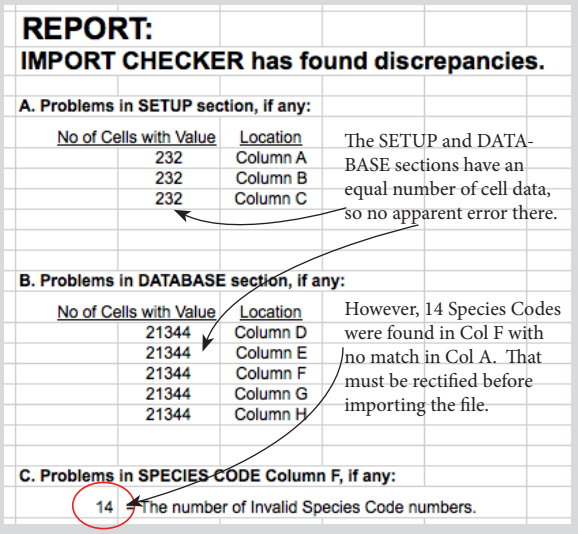


2. If no errors are found, this message is shown.



3. If errors are found, this message is shown followed by the Report below.





6. Launch CUFIM-PRO

a. When opening CUFIM-PRO, Excel displays the standard *This workbook contains macros* message box shown on the previous page in the sidebar. Be sure to click on ENABLE MACROS or the program will not work.

b. Import your database template from CUFIM-PRO's ID sheet. See Sheet 1 (ID), ahead, for detailed import instructions.

c. Check to see that it imported correctly. Scroll to the bottom of these sheets:

1) SETUP sheet (check species code, species list and species group).

2) DATABASE sheet (check all fields).

d. CUFIM-PRO will automatically format all data uniformly.

e. Now you're ready to run analysis and projections on your database.

Refer to Part II, Sheet 1, Step 1b (Import and Export Options) to complete the importing process.

7. Inventory Maintenance (years 1-5)

a. As trees are cut, fall or are otherwise removed, use the DATABASE sheet to delete trees from the system using the Delete Record section.

b. If some trees are remeasured during this period or errors need correcting, change the tree's attributes using the Change Existing Record section on the DATABASE sheet.

c. If only a few trees are planted, use the Enter or Add New Record section on the DATABASE sheet. If the entire urban forest is remeasured or many trees are to be added, see the next section.

8. New or Re-Inventory of the Urban Forest (at 5, 7 or 10 years)

a. It is recommended that the entire tree database be remeasured and updated every five years, but not more than every 10.

b. First, export all data (from the ID sheet), then from the DATABASE sheet, print all records. The Print drop box shows how many pages will be printed.

c. Set up a new CUFIM-PRO folder to house the new inventory files with a new name, e.g., CUFIM-PRO 2025. The name on the ID page, cell S28, must be changed to the new name, too. Currently it is set for the user's desktop.

d. Repeat sections 2-5 above with the new data set.

e. If the cost of a new inventory is prohibitive, a growth projection approach can be made. However, a person with excellent Excel skills is needed for it to be successful. See the next section for details.

G Growth Projections in Lieu of a New Inventory

The instructions for updating the urban forest database using the growth projection method is provided in the appendix, titled, "*Growth Projections in Lieu of a New Inventory*."

D. Compatible Computers and Testing

CUFIM-PRO has only been tested on Macintosh computers as noted below. It is known that there are significant differences between Excel 2011 for the Mac and Excel for the PC and attempting to execute on a PC is not advised. As of this writing I was unable to find a fast PC with a large monitor to make the necessary adjustments to the program for PC use.

The best advice is always to run CUFIM-PRO on the fastest Mac possible with the largest monitor.

This section discusses various testing done by the author to assist the user on selecting the appropriate Mac computer to use.

Mac Computers:

Number One Recommendation: CUFIM-PRO was developed on a 64-bit Late 2013 Mac Pro, 3 GHz, 8-core Intel Xeon E5 with AMD Fire Pro D700 graphics card, running OS 10.9.5 and a 30" monitor. The Excel® software is Microsoft® Excel® for Mac 2011, v. 14.4.6 (14106).

Throughout CUFIM-PRO there are notes as to the approximate length of time various subroutines will take. Routines that may run 10 seconds or more are identified with a note. Those that take more than 10 seconds alert the user with a beep upon conclusion.

An issue is the monitor size and the largest possible monitor is recommended as information displayed on each sheet occupies many columns and rows. Initially the sheet zoom levels were set at:

> ID = 75%, SETUP = 85%, DATABASE = 85%, STATS = 85%, OUTPUT PARAMETERS =75%, REMOVAL = 80%, MGT OPTIONS = 75% and VALUATION = 85%.
>
> These percentages should be changed to optimize viewing based on the users monitor size.

2. MacPro mid-2010. This is a 2 x 2.93 Ghz 6-core Intel Xeon Mac with 16 GB of 1333 MHz DDR3 memory running OS 10.7.5. The Excel version was 14.4.8 (150116), Mac 2011. All routines ran about twice as slow compared to the Late 2013 MacPro. This is not a deterrent for short routines (3 sec vs 6 sec), but for longer ones, there is a wait. The longest routine, about 9 minutes on the Late 2013 MacPro was 18 minutes on the Mid-2010 MacPro. Fortunately that routine is only run one time and only then if a large database is imported.

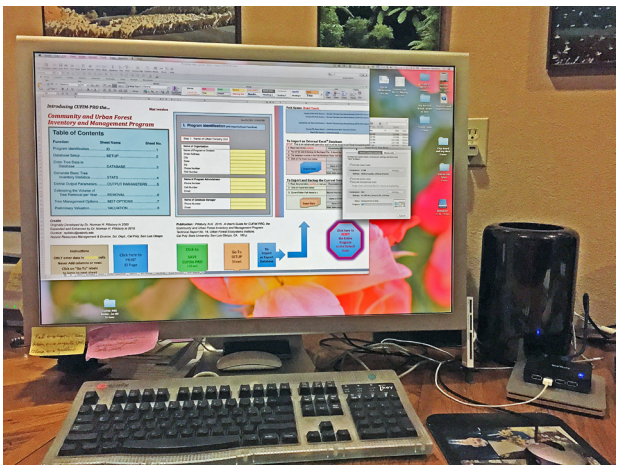
Recommendation: Running the program on this or a comparable computer would work, but it's not fast. The wait times are doable, but it would only be recommended if a faster computer wasn't available.

3. 2. G4. The 2003/4 version of CUFIM ran on a G4 desktop/laptop with Excel 2004. When I loaded CUFIM-PRO 14.4.8 (150116) on the same machine and software, some routines would not run (procedure too large) and those that did, were so intolerably slow (3 sec on Late 2013 vs about 1 minute on G4) that no one could survive the wait.

Recommendation: Don't try; it won't run.

Late 2013 Mac Pro

MacPro mid-2010



Late 2013 Mac Pro and a 30" monitor.

A User's Guide to CUFIM-PRO, the…

Community and Urban Forest Inventory and Management Program

Part II.

Tree Inventory and Database Setup and Maintenance

Part II. Tree Inventory and Database Setup and Maintenance

CUFIM-PRO, the Community and Urban Forest Inventory and Management program, is a powerful yet fairly straight-forward program to use. It can be taught to users who use Excel regularly in about 2 hours; new users will require more time.

It has many features, and an understanding of its capabilities will enable the user to develop and set up their own database.

A user who is first beginning to set up a database and collect tree data will find this program ideal to work with. If an existing database is to be brought into the program, the reader is referred to Part I and the appendix for instructions. Regardless, understanding how the program is designed will greatly aid the user.

CUFIM-PRO is an Excel spreadsheet program written with a combination of Excel functions, simple macros, and programming through the Visual Basic Application language. The user does not need to understand functions, macros or Visual Basic to run the program. CUFIM-PRO is one file consisting of eight sheets containing 14 steps for completion, if all options are selected.

The remainder of the User's Guide contains two parts, Part II and Part III (see Figure 1). Part II addresses tree inventory, database setup, and maintenance activities. This includes built-in tree parameters such as species, species' code, and volume equations, if available.

Part III, *Planning for Tree Removal Volume and Value Calculations,* is more advanced. It computes tree removal volume and value information, and allows the user to make projections of volume and potential income from trees slated for removal. The user can set constraints on the data set. For example, the minimum and/or maximum tree diameter can be defined which restricts the analysis to those limits.

Starting with Part III, there are two tracks or paths that the user can follow. Track 1 is a planning approach to making an estimate of the volume and value of trees that would be removed at a future date while Track 2 is used when it is certain that 100% of a segment of the forest will be removed within a relatively short period.

VERY IMPORTANT NOTE:

When typing on the spreadsheet, it is very important to observe these rules. Failure to follow them may result in incorrect calculations or the program might even stop working.

1. Do not type in any cell unless it is YELLOW.
2. Do not click in any cell unless it is YELLOW.
3. Do not tab or return into any cell unless it is YELLOW.

It is OK to click on:

List boxes,

Check boxes (These can be deselected by clicking on them a second time.)

"Jump" boxes (takes you to a new sheet)

"Go to" action boxes

"Print" action boxes, and

"Clear" action boxes as needed.

The program is designed so that a beginner can use just the sections needed, and as his or her skill level increases, more sophisticated analyses can be accomplished.

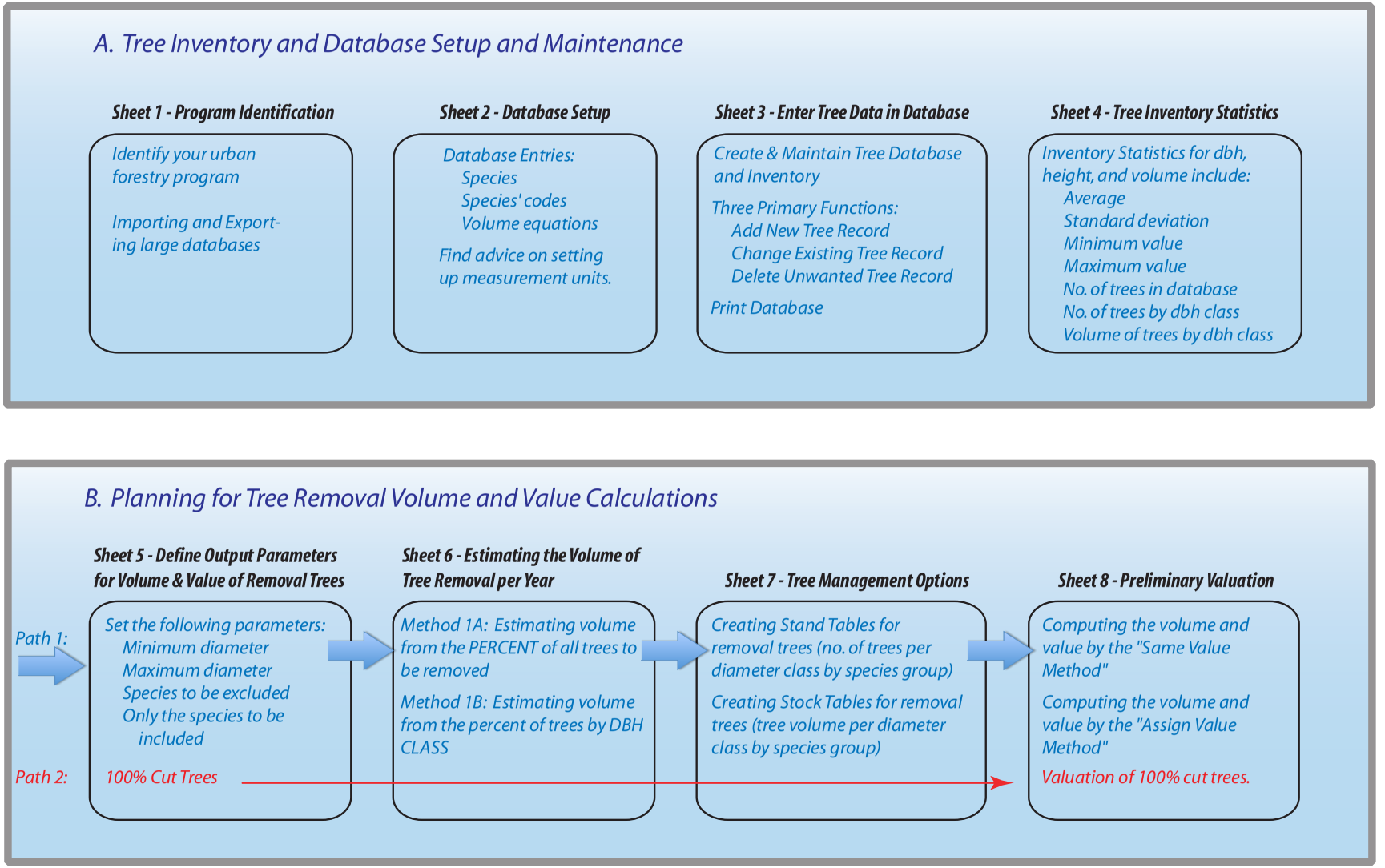
An expanded outline that shows the procedural steps is provided on the page following Figure 1.

Note: The user can download and copy this program (the CUFIM-PRO folder) to their desk top from web site (http://ufei.calpoly.edu/urbanwood/links.lasso; look for CUFIM-PRO (MS Excel Program)).

Figure 1.

Flow Chart for CUFIM-PRO…

Community and Urban Forest Inventory and Management



Outline of Showing Procedural Steps of CUFIM-PRO.

Sheet	Sheet Name	Full Name	Procedural Steps	
A. TREE INVENTORY, DATABASE SETUP AND MAINTENANCE.				
1	ID	Identification and Import/Export	Step 1a. Step 1b.	Name of Urban Forestry Unit Import and Export Options
2	SETUP	Database Setup	Step 2. Step 3. Step 4.	Species' Code, Species Name, Species Group Enter New Volume Equations Advice on Setting up Measurement Units
3	DATABASE	Enter Tree Data in Database	Step 5.	Create and Maintain Tree Database Inventory
4	STATS	Generate Basic Tree Inventory Statistics	Step 6.	Generate Basic Statistics for Tree Inventory
B. PLANNING FOR TREE REMOVAL VOLUME AND VALUE CALCULATIONS.				
5	OUTPUT PARAMETERS	Define Removal Parameters for Output	Step 7. Step 8. Step 9a. Step 9b.	Range of Tree Diameters to be Included in Volume Calculation of Removal Trees Indicate Species to be Excluded from Volume Calculation of Removal Trees Indicate Only the Species to be Included in Volume Calculation of Removal Trees Management Options for 100% Species Removal by Species and Diameter
6	REMOVAL	Estimating the Volume of Tree Removal per Year	Step 10.	Estimating the Annual Volume of Tree Removal by Randomly Selecting Trees, or by Field Marking Trees.
7	MGT OPTIONS	Tree Management Options	Step 11a. Step 12a. Step 11b. Step 12b.	Creating Stand Tables for Removal Trees Creating Stock Tables for Removal Trees Creating Stand Tables for ALL Trees in Database Creating Stock Tables for ALL Trees in Database
8	VALUATION	Preliminary Valuation	Step 13. Step 14.	Preliminary Valuation - Same Value Method and Assign Value by Species Preliminary Valuation - Assign Value Method to Species Groups or Species Groups by Tree Size

A. TREE INVENTORY, DATABASE SETUP AND MAINTENANCE.

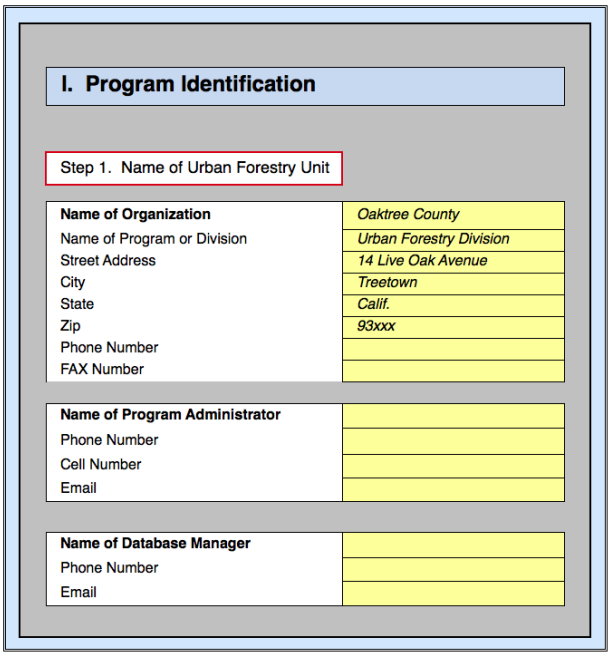
Having a good plan for your database is crucial to obtaining success when using it. Following the guidelines in Part I, *How to Design and Implement an Urban Forest Inventory*, will provide a solid foundation for your city-wide program.

This section is presented to assist the user in designing a database to be used with CUFIM-PRO, and walks the user through the various steps required for setup and use.

Sheet 1 ID (Identification and Import/Export)

Step 1a**.** Name of Urban Forestry Unit

On this sheet, enter information about your communities urban forestry program and management. Some of this information is printed out on various summary and database forms in later sheets.



I. Program Identification

Step 1. Name of Urban Forestry Unit

Name of Organization	*Oaktree County*
Name of Program or Division	*Urban Forestry Division*
Street Address	*14 Live Oak Avenue*
City	*Treetown*
State	*Calif.*
Zip	*93xxx*
Phone Number	
FAX Number	

Name of Program Administrator	
Phone Number	
Cell Number	
Email	

Name of Database Manager	
Phone Number	
Email	

Figure 2. Program Identification.

Green Save action boxes are located on all eight sheets. Clicking saves the main program, CUFIM-PRO with the date and time included on its name (Recommended).

Doing a Command-S (Control-S) saves the program with the same name.

Sheet 1 ID (Identification and Import/Export)

Step 1b**.** Import and Export Options

Two powerful routines, Import and Export, are located on Sheet 1 (ID). The Import function allows a person knowledgeable in Excel to easily bring in large .xls or .xlsx databases. However, the Export function can be completed with two clicks and requires no special Excel expertise. The Import routine is normally a one-time operation used to import a previous database or recent inventory. The Import function might also be used to re-import a backup copy.

Import setup:

Before attempting to use the Import routine, the following preparation must be done.

1. The CUFIM-PRO folder with the following list of files must be downloaded from the web to your desktop. Folders and Files included are (see Figure 4):

   a. Folder name: *CUFIM-PRO folder*

      1) File name: *Import Template.xlsx*

      2) File name: *Export Template.xlsx* (do not alter; do not move)

      3) *Import Checker.xlsm*

   b. Folder name: *CUFIM-PRO Program BACKUP Files*

      a) File name: *CUFIM-PRO.xlsm* (This is the main program. Please note the file name extension (.xlsm) is required to run the program as it includes dozens of macros and subroutines written in Visual Basic Application language. If the file extension is changed the program will not work.

   c. Folder name: *CUFIM-PRO Database BACKUP Files*

      a) Files: None. This is where your exported backup files are automatically stored.

2. In Part I, *How to Design and Implement an Urban Forest Inventory*, sections C.2 through C.5, includes explicit instructions on field measurements, compiling data and entering it into the *Import Template.xlsx* file (Figure 3).

   a. It is important to review and adhere to the instructions in the following sections of

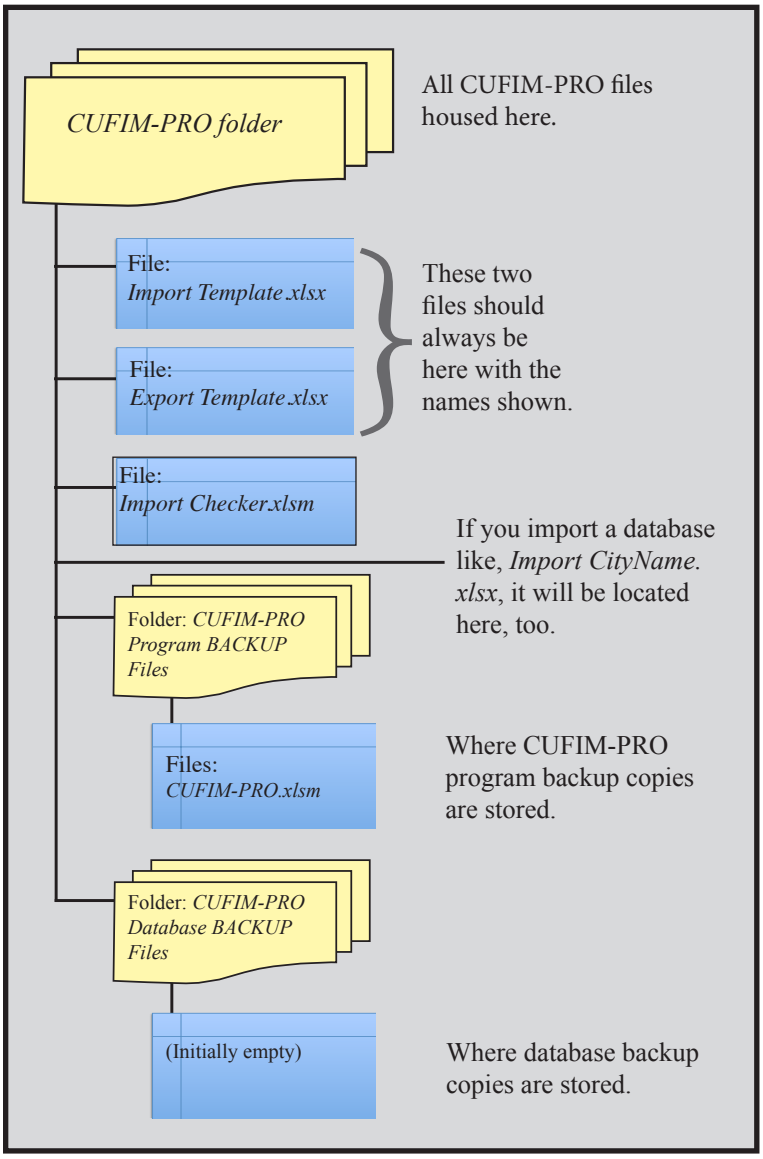


Figure 4. CUFIM-PRO folder and file structure.

Part I prior to attempting to use the Import function:

C.2. Collecting Inventory Data in the Field,

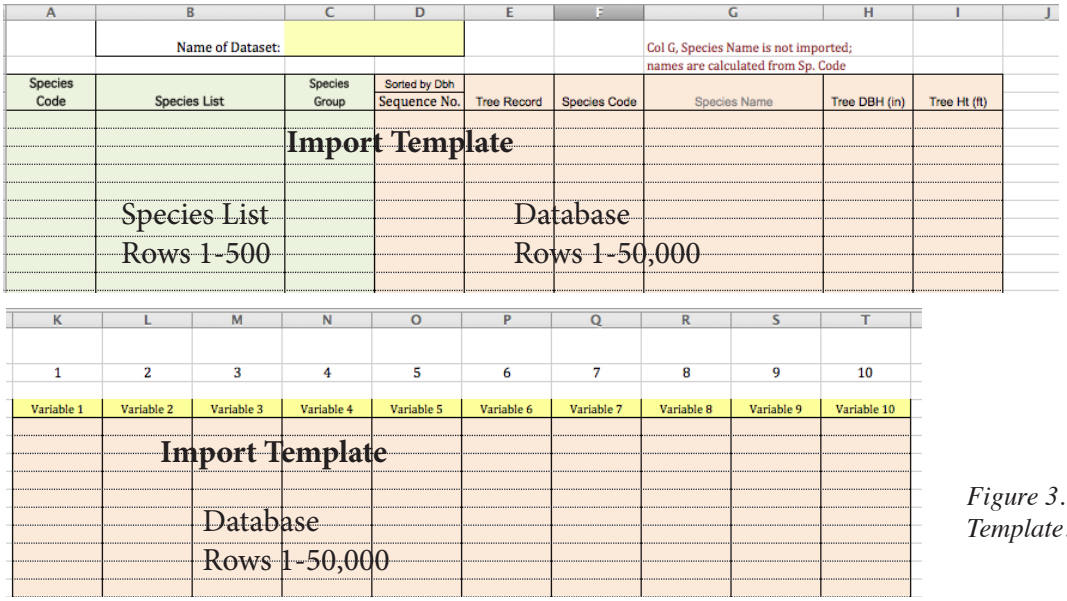
C.4 Entering Data into the Import Template

C.5. Checking Data After Entry in Import Template

And lastly, to run the included *Import Checker.xlsm* program to identify omissions in the database and that the species codes are properly matched with individual trees. Follow the procedure in Appendix E (Trouble Shooting).

3. On the ID sheet, scroll to the right. There is a section titled "Path Names" at the top; do not touch as the program reads several path names to function properly. In fact, there is only one YELLOW cell available (cell S16), (ONLY YELLOW cells are cells that you may click on), and that is the path name to the Desktop where the CUFIM-PRO folder has been placed. If cell S16 doesn't provide the proper path name automatically you must enter your desktop path name in that cell.

4. Place the completed Import Template in the CUFIM-PRO folder. It should have a different name than *Import Template*, such as *Import CityName.xlsx*, where CityName is replaced by the name of your city.



5. Important: The Import file has 1 sheet (tab) and it MUST be named "Database," without the quotes.

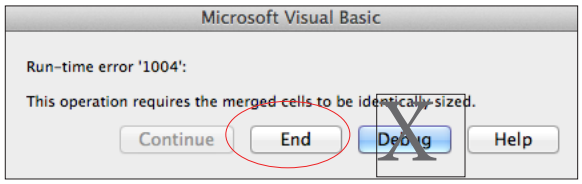
Figure 3. Import Template.xlsx

Importing the Database:

1. Launch CUFIM-PRO, select the ID sheet and click on the Import icon, Figure 5. The program will perform several steps requiring a response as shown by the message box flowchart on the right, Figure 6.

2. What if an error occurs during import or during any normal operation? Don't panic, most issues are easily resolved, but how you first respond is important.

a. First -- in the unusual case of a Microsoft Visual Basic Run-time error, DO NOT click on DEBUG. Instead click on END. Clicking on Debug will take you into the programming portion of the source code and altering any code will likely cause the program to fail.



b. Next, go to the Trouble Shooting section in the appendix for detailed information on the next steps you must take.

c. All major routines in CUFIM-PRO include Excel's *On Error* coding. If *On Error* detects a run-time error, it will immediately provide a non-Microsoft Visual Basic error message and after clicking OK, it returns to the stable state prior to the operation you initiated (example below). Thus experiencing a Microsoft run-time error is most unlikely. See headers of these two images.

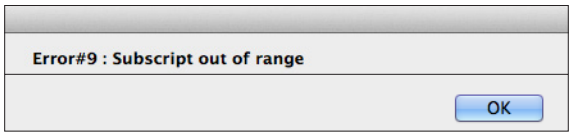


Figure 6. Flowchart for Import Steps

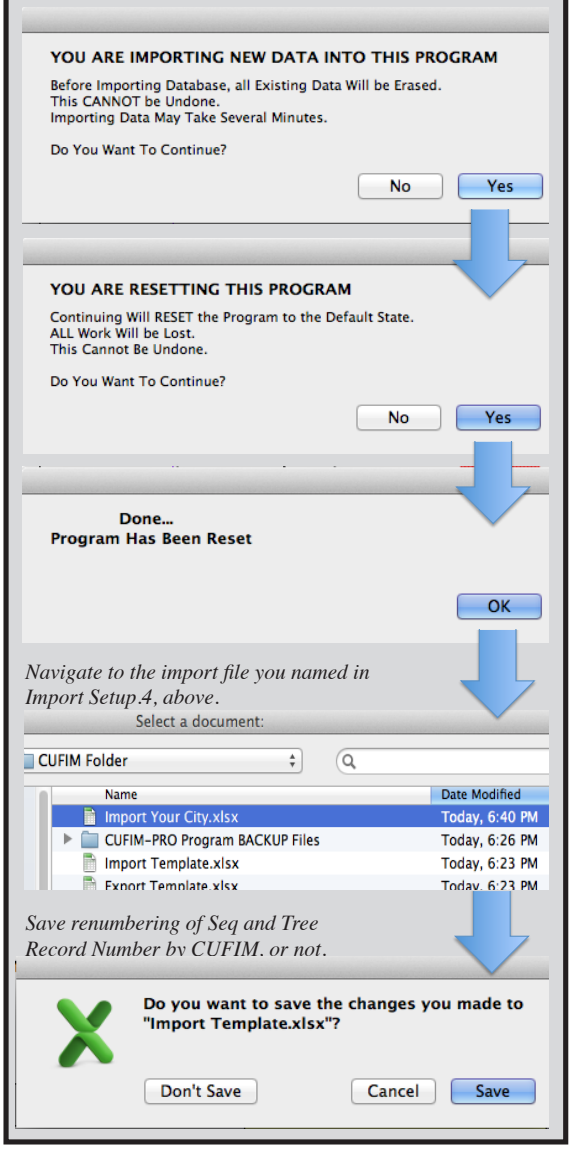


Figure 5. Import section.

To Import an External Excel® Database

STOP: This is an advanced operation and must be done by an Excel knowledgeable person!

1. Read User Guide carefully.	It's available with this program at: http://www.ufei.org/files/ufeipubs/CUFIM_Report.pdf	
2. You will be able to browse for the Import File. It should be in the same folder as this program (CUFIM-PRO).		
3. The database must be in the first Workbook Sheet (left most position).		
4. Click on the Import Icon below.	Name of Dataset Imported:	MONTEREY DATA
Import Data	Pathname to the Desktop	Stormin' Norman:Users:Norm:Desktop:
	Name of CUFIM-PRO Program Backup Folder:	CUFIM-PRO Program BACKUP Files:
	Name of CUFIM-PRO Program Backup File:	CUFIM-PRO
	Note: Make sure pathnames end with a colon (:)	Ex: CUFIM-PRO mm dd, yyyy--Time is hh mm ss.xlsm

To Export and Backup the Current Database (as an ".xlsx" File Extension)

Figure 7. Export section.

1. Read documentation carefully in manual.	It's available with this program at: http://www.ufei.org/files/ufeipubs/CUFIM_Report.pdf	
2. Click on Export Icon below.	CUFIM Folder Name:	CUFIM Folder:
3. Current Folder Path Name is =	Stormin' Norman:Users:Norm:Desktop:CUFIM Folder:	
Export Data	Name of CUFIM-PRO Database Backup Folder:	CUFIM-PRO Database BACKUP Files:
	Name of CUFIM-PRO Database Backup File:	CUFIM-PRO Database
	Note: Make sure pathnames end with a colon (:)	Ex: CUFIM-PRO Database mm dd, yyyy--Time is hh mm ss.xlsx

Note that CUFIM-PRO automatically numbers all Sequence and Tree Record numbers from 1-n. The last query from Excel in Figure 6 provides the user the option of retaining the new numbering system in the template, or not. We recommend SAVE.

3. Check to see that it imported correctly. Scroll to the bottom of the SETUP and DATABASE sheets to check all fields for proper format.

4. CUFIM-PRO will automatically format all cells.

5. Since the imported file is now part of CUFIM-PRO, the program also SAVES another copy of itself, CUFIM-PRO, to the *CUFIM-PRO Program BACKUP Files folder* with the date and time affixed to the name, e.g., *CUFIM-PRO Mar 01, 2017--Time is 22 32 00.xlsm*.

Exporting the Database:

1. CUFIM-PRO is a self-contained program, that is, it includes the program plus the database. Thus your first task after importing is to Export the database portion of the program as a backup. This is a much easier operation than using the Import function and doesn't require special Excel knowledge. Note: CUFIM-PRO uses the *Export Template.xlsx* to backup the database, and a copy of it must be in the CUFIM-PRO folder.

2. On the ID sheet, Figure 5, click on the Export icon. This message box appears, Figure 8.

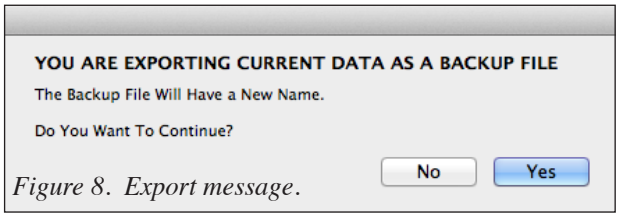


Figure 8. Export message.

3. Click Yes, and pause for the "Please wait" messages to disappear before proceeding.

4. Open the *CUFIM-PRO folder*, then the *CUFIM-PRO Database BACKUP Files folder* on your desktop. Confirm that the exported backup file is there. It will have the original name plus the date and time attached. It is an .xlsx file. Open it to confirm all data is included.

5. Having the date and time "stamped" on the name allows you to quickly see which backup is most recent. This folder can hold as many backups as desired, only limited by hard disk space. If the number of files is overwhelming, off-load all but the most recent few to an archive hard drive.

6. It's useful to set both the program backup and the database backup folders to the List View, sorted by Date Modified. That way your most recent files always remain at the head of the list.

Figure 9. A "jump box," or action box.

To navigate from sheet to sheet use a "jump box," Figure 9.

Sheet 2 SETUP

Step 2. Species Code, Species Name, and Species Group

If you have already imported a large database according to the instructions in Part I, Part II (Sheet 1, Step 1b), and the appendix you can skip the next three items (1, 2, 3).

In this step, each species must be assigned three fields of information, 1) a numeric species code, 2) a text species name, and, 3) a species group number from a built-in list. See Figure 10.

Species Code	Species List	Species Group
1	Acacia baileyana	2
2	Acacia longifolia	2
3	Acacia melanoxylon	2
4	Acer buergeranum	6

Figure 10. Species list on the SETUP sheet.

Warning, do not type in this chart (no yellow cells). Instead use the following sections on the spreadsheet to enter or delete this information.

1. To enter Species Code information, only use positive integers from 1-500 in the "*Enter New Species*" section below. Use consecutive numbers. The code MUST be valid. Set up a numbering scheme before entering codes.

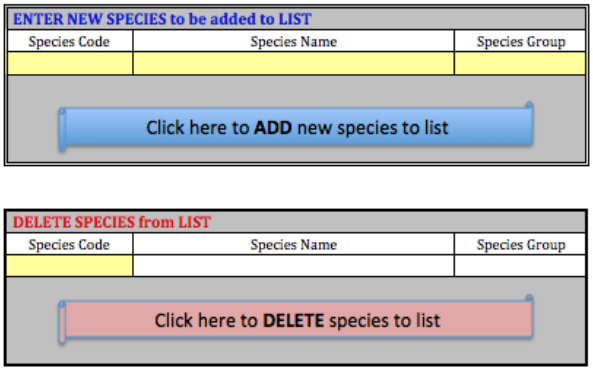


Figure 11. Adding and Deleting species from the Species List.

2. The Species Name is normally the scientific name, however, the program will accept any text in this field.

3. Species Group: Only use positive integers from 1-50. However, the species group number must be one that is already built into the program, or will immediately be defined later in Sheet 2, Step 3. Note that information for Species Groups 1-19 already comes with the program (see Sheet 2, Step 3).

The Species Group number selected should be of the group name that most closely resembles the tree type, species, branching and volume characteristics. For example, Canary Island pine would be best matched with Group 3, the Monterey pine group, or canyon live oak and interior live oak would be matched with the coast live oak species group.

Successful matching of a species to the group name requires knowledge of the tree characteristics in the inventory, as well as those of the groups listed in Figure 12, below. To assist the user, there are photos of the first 15 species (listed in Figure 12) in Pillsbury *et al* (1998) and they are provided in the appendix of this report. Each species is photographed in three sizes, small, medium, and large.

4. To delete a species, only the species code is needed, Figure 11. Be certain to Save changes frequently and Export an updated copy when done.

Sheet 2 SETUP

Step 3. Enter New Volume Equations

The program comes with 19 built-in species groups and their volume equations. It is possible to add more as equations become available. Doing so requires careful entry work and a good understanding of the process so the program will run correctly.

1. In this step the user may name a new Species Group. It must be assigned the next available Species Group number. Group numbers must stay in consecutive order.

2. Each new Species Group must have Local Volume Equation coefficients entered into the correct field. Optimally and strongly recommended, Standard Volume Equation coefficients should be entered, if available. They are desirable because Standard Volume Equations are more accurate than Local Volume Equations. If both are available, enter both local and standard coefficients. Note: the units calculated by new equations must be in cubic feet. No other units will work.

Local and Standard Volume equations are of the following form:

Local Volume Equation: $V = aD^b$

Standard Volume Equation: $V = aD^bH^c$

where V is total tree volume in cubic feet computed from Smalian's formula, D is diameter at breast height (dbh) in inches, and H is total height in feet. Regression coefficients a, b, and c are statistical parameters calculated by the author.

IMPORTANT: As a minimum, a Local Volume equation for each species MUST be entered to ensure the program will run.

Figure 12. Volume Coefficient Lookup Table and Species Group Numbers.

Volume Coefficient Lookup Table

Species used for Species Group	Species Group	Local vol coef.		Standard vol. Coef.			Percent of tree volume by stem dia class						
		a	b	a	b	c	<4	4-8	8-12	12-16	16-20	>20	Total
	-1	0	0	0	0	0							
Blue Gum	1	0.055113	2.436970	0.003089	2.151822	0.835731	4.0	10.0	10.0	11.0	11.0	54.0	100.0
Acacia	2	0.048490	2.347250	0.014058	2.186485	0.467357	9.7	21.0	17.5	15.5	12.0	24.3	100.0
Monterey Pine	3	0.019874	2.666079	0.005325	2.226808	0.668993	6.0	14.0	9.0	11.0	9.0	51.0	100.0
Monterey Cypress	4	0.035598	2.495263	0.005764	2.260353	0.630129	5.0	11.0	8.0	8.0	9.0	59.0	100.0
Carob	5	0.066256	2.128861	0.008573	1.795854	0.926668	11.0	27.0	18.0	8.0	10.0	26.0	100.0
Camphor	6	0.031449	2.534660	0.009817	2.134803	0.634042	12.9	27.7	23.1	13.6	10.6	12.1	100.0
Chinese Elm	7	0.028530	2.639347	0.010456	2.324812	0.493171	12.8	25.1	19.8	21.6	14.0	6.7	100.0
Holly Oak	8	0.025169	2.607285	0.004307	1.821580	1.062691	17.3	33.9	19.8	17.0	9.1	2.9	100.0
Jacaranda	9	0.036147	2.486248	0.011312	2.185780	0.548045	12.8	25.3	24.9	20.5	9.6	6.9	100.0
Liquid Ambar	10	0.030684	2.560469	0.011773	2.315815	0.415711	13.1	21.1	23.0	19.6	14.7	8.5	100.0
Modesto Ash	11	0.022227	2.633462	0.001287	1.762964	1.427822	12.0	28.0	21.0	14.0	9.0	16.0	100.0
Sawleaf Zelkova	12	0.021472	2.674757	0.006664	2.363178	0.551904	28.5	27.5	15.0	9.0	6.0	14.0	100.0
Chinese Pistache	13	0.019003	2.808625	0.002921	2.191572	0.943669	27.0	32.0	18.0	11.0	11.0	1.0	100.0
Southern Magnolia	14	0.022744	2.622015	0.004486	2.070408	0.845627	18.0	23.0	15.0	15.0	14.0	15.0	100.0
London Plane	15	0.025170	2.673578	0.010425	2.436420	0.391682	14.0	22.0	15.0	14.0	14.0	21.0	100.0
Sycamore	16	0.075051	2.335230	0.050119	2.177385	0.212477	12.8	25.3	24.9	20.5	9.6	6.9	100.0
Redwood	17	0.044697	2.390837	0.002438	1.694874	1.098957	6.0	14.0	9.0	11.0	9.0	51.0	100.0
Coast Live Oak	18	0.042542	2.466100	0.006526	2.319580	0.625280	18.0	23.0	15.0	15.0	14.0	15.0	100.0
Tanoak	19	0.119906	2.028700	0.004748	1.933890	0.819070	13.1	21.1	23.0	19.6	14.7	8.5	100.0

CUFIM-PRO selects the Standard Volume equation if tree heights have been entered, otherwise, the Local Volume equation is used.

3. The seven columns in Figure 12 headed by the title "*Percent of tree volume by stem dia. class*" have been determined for groups 1-19 from various studies by the author where these values were actually determined. The values for groups 20-49 would have to come from new studies. Alternatively, the user could select the values from species groups 1-19 that most closely matched the volume of the new species group being entered, however they won't be as accurate. Group 50 is reserved for non-volume trees like palms or stumps.

If you are entering this information, make sure the percentages for each stem diameter size totals 100% in the last column. This data is used to express a trees' volume by stem size for tree valuation in Step 14. From a product development standpoint, this information is essential.

Sheet 2 SETUP

Step 4. Advice on Setting up Measurement Units

The settings discussed in this section are for information only. Selecting or not selecting choices does not affect the program and none of these choices are used in any way. However, it is important to set and stick with a measurement system. This step allows the user to see the outcome of various choices.

Choices that need to be made before a new inventory is undertaken were presented in Part I and are discussed again below. If a database already exists, these choices may already be set, however, they should be reviewed to make sure they follow these guidelines.

A. Diameter measurements

1. Precision of tree diameter measurements. *It is strongly recommended that all tree diameters be measured with a D-tape to the nearest 0.1"* as this level of precision provides an excellent estimate of tree volume.

2. Diameter classes. It is not recommended that diameter classes be used. However, if classes must be used, the smallest possible diameter class interval should be selected and maintained throughout the inventory. Examples are: 0.5" or 1.0" classes. The best case would be 0.5". Trees measured this way would have a dbh value entered as the midpoint of the class, as tree volume cannot be calculated from an interval.

3. All diameter classes must be of the same interval. Do NOT make diameter classes of unequal intervals. An example of diameter classes that should NOT be used is: <6", 6-9", 9-18", 18-24", >24.

In this case the largest diameter is open ended and it's impossible to calculate tree volume. An example of a proper system for a 1.0" diameter class and a 2' height class follows.

One-inch Dbh	Interval			Two-foot Ht	Interval		
Class	Low		High	Class	Low		High
2	1.6	-	2.5	10	9	-	11
3	2.6	-	3.5	12	11	-	13
4	3.6	-	4.5	14	13	-	15

B. Height measurements

1. Precision of tree height measurements. Examples of height precision would be to the nearest 1', 2', or 5'. Ideally it would be, 1' or 2'. The smaller the precision number the better the estimate of tree volume. Once selected, all trees should be measured to that precision.

Trees would have a height value entered as the midpoint of the class, as an interval cannot be used to calculate volume.

Figure 13a. Database for columns 1-15.

1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9) %, cf	(10) %, cf	(11) %, cf	(12) %, cf	(13) %, cf	(14) %, cf	(15) %, cf
umber	Count	Count	Count	Count	Count	Count	Count	100.0	0.0	0.0	0.0	0.0	0.0	0.0
344	21,344	21,344	0	21,344	21,344	21,344	21,344	779,991	0	0	0	0	0	0

y Seq	Sort	Sort		Sort	Sort	Sort			Click here to Update Cubic Feet of Tree Volume by Branch Diameter Size					
									Cubic Feet of Tree Volume by Branch Diameter Size					
ice No.	Tree Record No.	Species Code	(Set by Random Number Generator)	Species Name	Species Group	Tree Dbh (in)	Tree Height (ft)	Volume (cf)	Vol in < 4"	Vol in 4-8"	Vol in 8-12"	Vol in 12-16"	Vol in 16-20"	Vol > 20"
1	7579	101		Eucalyptus camaldulensis	1	14.8	57.4	29.9						
2	7581	101		Eucalyptus camaldulensis	1	14.8	57.4	29.9						
3	7634	101		Eucalyptus camaldulensis	1	20.7	57.4	61.7						
4	7580	101		Eucalyptus camaldulensis	1	14.8	57.4	29.9						
5	2998	101		Eucalyptus camaldulensis	1	27.9	8.2	23.1						
6	8508	102		Eucalyptus cinerea	1	14.8	24.6	14.7						

2. All height classes must be of the same interval. As discussed before, do NOT make diameter or height classes of unequal intervals. An example of height classes that should NOT be used is: <20, 20-30', 30-40', 40-50', >50'.

Do NOT allow the largest diameter or height class to be open ended. For example, do NOT have a height class of >50'. It is not be possible to obtain accurate estimates of volume with this type of open ended class. An example of a proper system for a height precision of 2' is shown in the chart above.

Once the "rules" of measurement are determined, they must be written and passed to new employees. And most importantly, they must be followed.

Sheet 3 DATABASE

Step 5. Create and Maintain Tree Database Inventory

Note: A section devoted to transferring an existing database into this program can be found in the appendix.

This discussion starts with the database itself and assumes an existing database or new inventory has been completed and was imported. This section addresses how to add, change and delete records. The database is comprised of 25 columns, 15 which are built-in, and 10 which are user-defined variables (UDVs). Two of the UDV's are suggested as Location/Address and Date Measured, however, they can be changed as needed. See Figures 13a and 13b at bottom the last page and below.

Column 1: The Sequence Number is the number that CUFIM-PRO assigns the tree record. It is always consecutive, from 1 to n. This number is used extensively by CUFIM-PRO.

Column 2: Tree Record number is the number the user gives the tree. This must be a positive integer and normally consecutive, although over time small gaps may occur due to deletions. If data has been imported it will automatically be in order from 1-n.

Column 3: Species code value is based on the species list in Step 2 on the SETUP sheet. It is determined by the user either when setting up a database to be imported or when entering a new record (see how to enter a new record, next page).

Column 4: Removal code is an integer automatically calculated by one of the options on the REMOVAL sheet (to be discussed later). It is always a value of 1 if the tree is selected for removal, and blank if the tree is not to be removed.

Column 5: Species Name is based on Step 2 from the SETUP sheet. It is automatically calculated based on the species code assigned to the tree.

Column 6: The Species Group number is based on Step 2 from the SETUP sheet. It is automatically transferred to the DATABASE sheet.

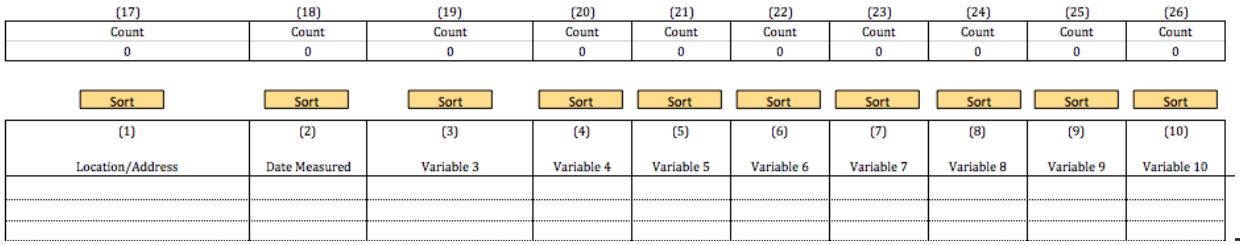
Column 7: Tree Dbh is in inches, and is a value measured in the field. The format and precision should follow the advice provided in Step 4 under SETUP.

Column 8: Tree Height is total height in feet, and is a value measured in the field. The format should follow the advice provided in Step 4 under SETUP.

Column 9: Volume is total tree volume in cubic feet. It is calculated based on built-in or added coefficients in Step 3 under SETUP. If Standard Volume coefficients are available, they will be used, otherwise, the Local Volume coefficients are used by CUFIM-PRO.

Columns 10-15: Branch volumes, sometimes called stem volume, are shown by six size classes. These values are obtained or updated by clicking the yellow "*Click here to Update Cubic Feet of Tree volume by Branch Dia Size*" rectangle. Allow 1-10 minutes for computation depending on computer speed and size of database. The source of these equations is *Tree Volume Equations for*

Figure 13b. Database for columns 17-26.



Fifteen Urban Species in California by Pillsbury, Reimer and Thompson (1998). Only update branch volumes after you finish importing, adding, changing or deleting data records, discussed below.

Headers above Columns 1-16 show:

Col 1	the maximum number in database
Cols 2-8	count, or the number of items in each column
Cols 9-15	sum of the volumes in each column in the database
Col 16	the sum of all branch sizes in percent and cubic feet.

The database is "balanced" when columns 1-3, and 5-8 all show the same value. If they don't agree, search the database (or hard copy) to find errors or missing data. These errors should be fixed using the Change or Delete functions on this sheet. The only exception is Total Height (column 8) which could be less if some heights were not measured or entered.

There are three function boxes in this section that allow the user to Enter (or add) a new record, Change an existing record, and to Delete an unwanted record.

Entering New Records: To enter or add a new record, use the blue section. The next Sequence Number and Tree Record Number are automatically generated.

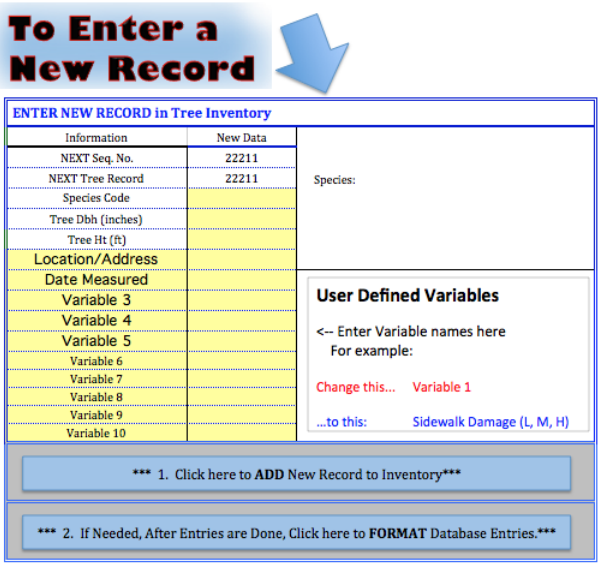


Figure 14. Function box to Enter a new record.

The user must enter Species Code, and dbh in inches. Optionally, the user may also enter total height in feet, and information about user defined variables 1-10. DO NOT add new data except in yellow cells in the function box.

User defined variables can be numeric or text. See Part I for further discussion. Enter the name of the variable and the acceptable units of measure. Keep names as short as possible. For example:

Numeric example: Dist fr Street (ft)

Text example: Sidewalk Damage (L, M, H).

You would have to develop criteria to distinguish among Low to High levels of damage.

Once the data has been typed in the yellow cells, add this information to the inventory database by clicking the blue box at the bottom, titled "****1. Click here to ADD New Record to Inventory****." On large databases there may be up to a three second delay between entries while CUFIM-PRO checks the data, enters and reformats the cells.

Newly entered data is normally formatted, however, if it is not, click on blue box #2, *"If Needed, After Entries are Done, Click here to FORMAT Database Entries."* This operation can take 1-5 minutes.

Changing Records: Follow these steps to change an existing record.

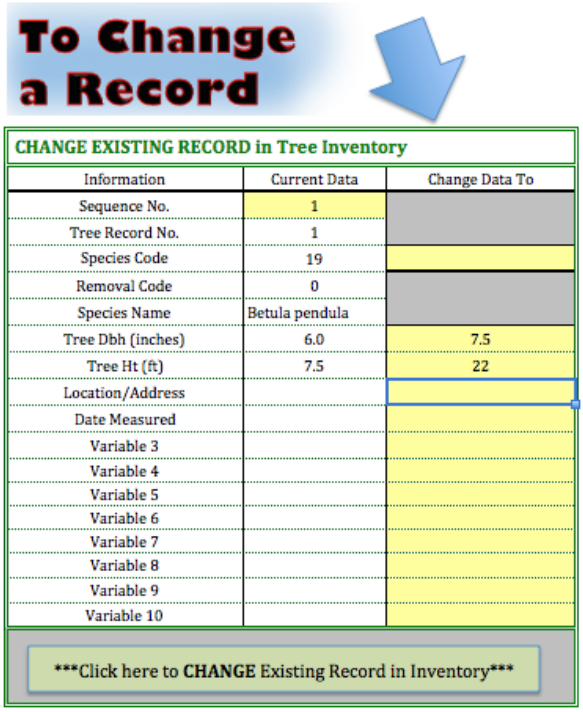


Figure 15. Function box to Change an existing record.

1. Type the tree Sequence Number in the green function box's yellow cell. A sequence number can be obtained in two simple ways. First, print a hard copy, which includes sequence numbers. Secondly, split the screen and scroll down to the desired tree and read the sequence number from column 1. DO NOT change the data except in the function box.

2. Check data about that tree in the *"Current Data"* column to be sure this is the correct tree to be modified.

3. Enter the new data in yellow cells under the column titled "*Change Data to.*"

4. Click on the green box at bottom, titled "****Click here to CHANGE Existing Record in Inventory****."

In the example above, tree sequence number 1 is a *Betula pendula*, 6.0" dbh, 7.5' tall and the new dbh is 7.5" and new height is 22'. The results of these changes will appear in the database.

Deleting Records. Deleting records is a two step process, Figure 16.

1. Type the tree Sequence Number in the red section's yellow cell.

2. Check data in next column to be sure this is the correct tree to be deleted from the database. Be careful, there is no undo option.

3. Then click the box labeled "*1. Click here to DELETE Record from Inventory*"

4. Repeat steps 1-3 as often as needed.

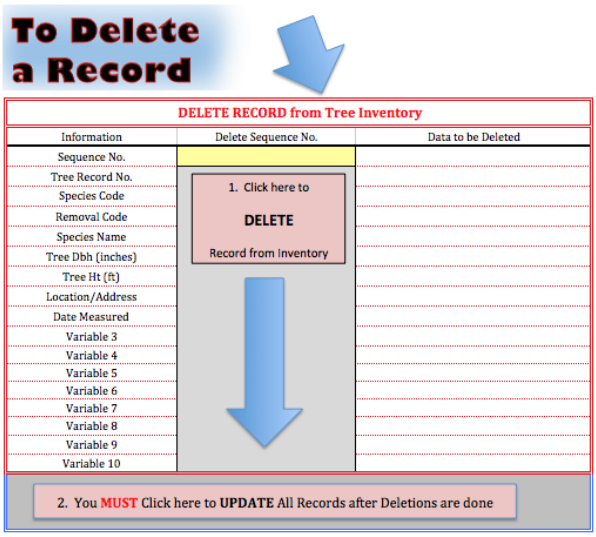


Figure 16. Function box to Delete an existing record.

5. When all deletions are completed, you MUST click on the cell labeled: "*2. You MUST Click here to UPDATE All Records after Deletions are done.*" It will take 15-30 seconds to reorder the database.

6. Use the Print box to print the Database, Figure 17.

a. Click and hold on the "*Print Options*" drop box. Two options are displayed.

1) Only the first page is printed of columns 1-15, or columns 17-26. To print the first page of all columns first select 1-15, then 17-26.

2) All pages are printed of columns 1-15 or columns 17-26, To print all columns of all data select 1-15, then 17-26.

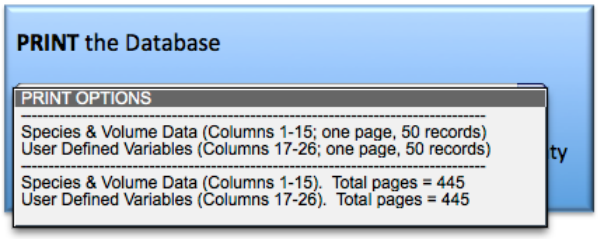


Figure 17. Print first page of database, or Print all pages.

The total number of pages will vary depending on the database size. The total number of pages that will print is shown; be sure to have an adequate supply of paper available.

When your database has 30 or more records (trees), your screen may be too small to view them without splitting the screen horizontally and scrolling to the area of interest in order to see the effect of additions, changes and deletions.

Sheet 4 STATS

Step 6. Generate Basic Statistics for Tree Inventory

Refer to Figure 18 for STATS sheet information and options.

1. Part A, "A. *Tree Inventory Statistics*," is always current.

2. To update "*B. Number of Trees per DBH Class*" and "*C. Volume by DBH Class,*" click the yellow update action boxes.

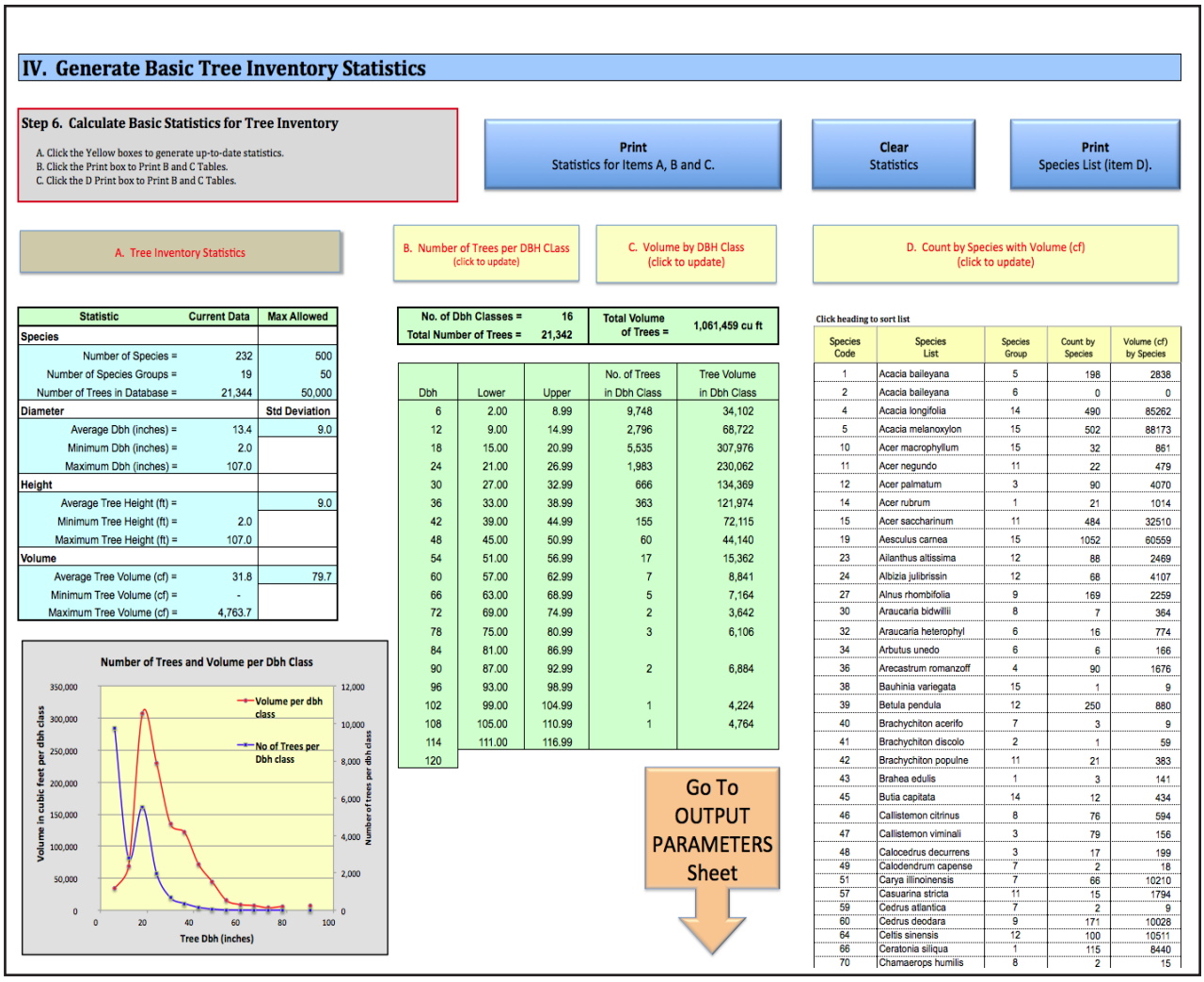


Figure 19. Options and information on the STATS sheet.

3. Click the print box for a hard copy of items A, B and C.

4. Section D is a complete list of all species in the database, the number of individuals and their collective volume It can be sorted by clicking on the yellow column headers, and printed by clicking on the Print Species List print box.

5. While unlikely, it is possible that the total number of trees and the total volume shown in Figure 19 differs slightly from the same numbers shown in the header of the DATABASE sheet. Any difference is because trees over 117" dbh are not included on the STATS sheet.

6. A diameter and volume distribution graph allows the user to picture the urban forest composition. See Figure 19.

This concludes Part II, *Tree Inventory and Database Setup and Maintenance*. With this information the urban forester can set up a community-wide database, import inventory measurements, define species groups, add, change and delete individual tree records, view basic statistics on the forest and backup the database with the export function.

Advanced features are available in Part III, *Planning for Tree Removal Volume and Value Calculations*.

A User's Guide to CUFIM-PRO, the…

Community and Urban Forest Inventory and Management Program

Part III.

Planning for Tree Removal Volume and Value Calculations

Part III. Planning for Tree Removal Volume and Value Calculations

This part of the guide describes advanced options and management tools for the urban forester.

From this point forward, there are two tracks or pathways to the concluding sheet, VALUATION: the General Planning Pathway and secondly, the 100% Cut Pathway.

1. General Planning Pathway. This approach is for planning ahead for a year or more when only an estimate of the volume and value of removal trees is needed.

For this method CUFIM-PRO uses the users estimate of the number of trees that will be removed in a future year, or over several years. It's based on the Species Grouping defined earlier. Restrictions on the database such as diameter as well as which species are excluded or included in the analysis are easily set.

Next the user is directed to the REMOVAL sheet where various options for setting percentages of tree removal are selected, and a random selection from all trees in the database is used for the estimate of volume and value.

The MGT OPTIONS sheet shows how the species are distributed by number of trees and their volume by dbh class in the urban forest.

Finally, the trees marked for removal form the basis for an estimate of their value on the VALUATION sheet.

2. 100% Cut Pathway. When 100% of an affected species or multiple affected species of the forest will be removed in the near future, the user will follow the approach described here. The cause for such a need could be fire, insect or disease outbreak or some other catastrophe such as damage due to extreme weather conditions.

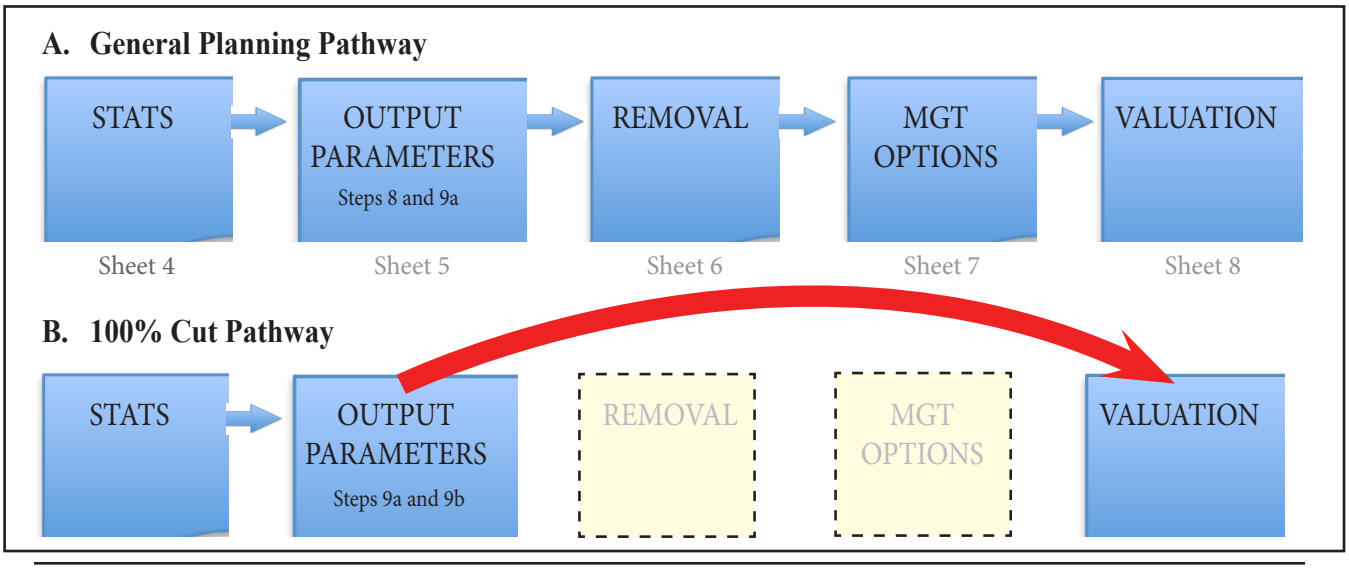
This is not a "planning" exercise but rather a specific calculation of the number of trees per dbh class that will be removed, and their value.

The user starts in Step 9a, *"Indicate only the Species to be INCLUDED in Volume Calculation,"* with a list of species obtained from the STATS sheet and a good idea of the diameter range of affected trees. In Step 9b, *"Management Options for 100% Species Removal by Species and Diameter,"* tables showing how their number and volume are distributed throughout the forest are created.

Next, the identified trees are "marked" for removal in the database where CUFIM-PRO inserts a "1" in the Removal column. There is also an option to print a report listing only removal trees and their attributes.

Last, the user MUST skip the REMOVAL and MGT OPTIONS sheets and determines the dollar value from the VALUATION sheet (see chart below).

The 100% Cut Pathway is described briefly at the end of Part III and in detail, by case study, in Part IV, *The Big Freeze*.



The two paths cannot be mixed in the same run; the user either follows the General Planning Pathway OR the 100% Cut Pathway. However, the user could follow both tracks by conducting two separate computer runs.

B. PLANNING FOR TREE REMOVAL VOLUME AND VALUE CALCULATIONS.

The main objective for the following sheets is to calculate, analyze and evaluate information for trees that would be removed in a given year.

Note that Part III is a series of planning routines that help with "what if" questions and is based on percentages of trees typically lost annually. It provides valuable information such as number of trees expected to be removed, the potential total volume removed and their estimated value. The program marks trees for removal based on user settings to arrive at volume and dollar value. In reality, the trees "marked" for "removal" are NOT removed from the database, but the program tags them as "removed" to make the calculations. Only the user can delete them using the Delete Record Function on the DATABASE sheet.

The importance of making volume and value estimations allows the urban forester to plan for resources, potential income, lead-time in developing contracts or sub-contracts, assessing manpower needs and so on. It also provides some leverage for spending prior to receiving income from removed trees. And last, it shows those in the political and funding arenas that managing street trees on an "urban forest-wide" basis can pay for its many functions such as planting, sidewalk repair, trimming around powerlines, and tree removal.

Sheet 5 OUTPUT PARAMETERS

Step 7. Range of Tree Diameters to be Included in Volume Calculation of Removal Trees

In this step the user is provided with a number of constraints that can be imposed on the data, the first which is selecting the diameter range for "removal" trees. Others will be discussed below.

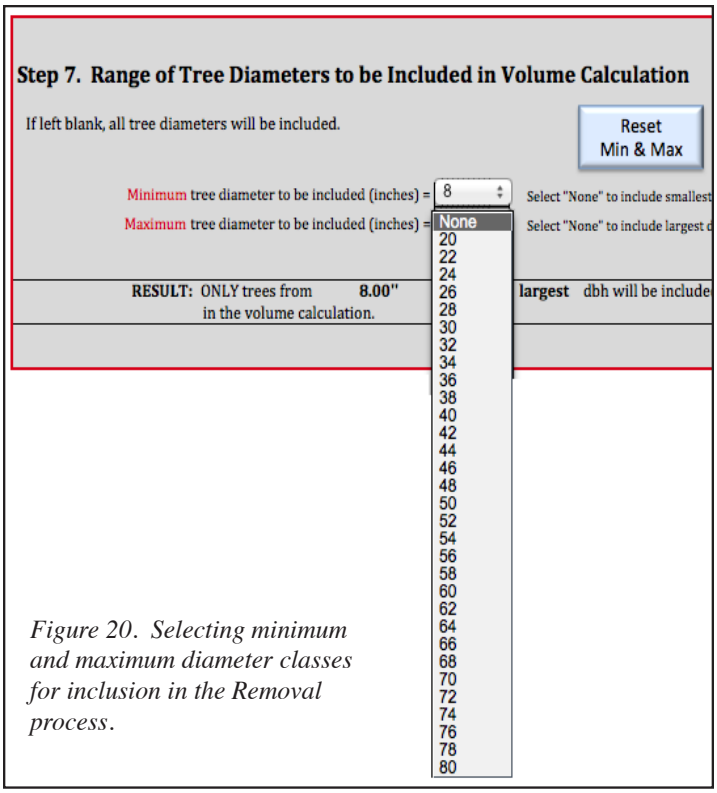


Figure 20. Selecting minimum and maximum diameter classes for inclusion in the Removal process.

Step 7 is used to limit the size of tree (dbh) that will be removed, which in effect, limits the size of tree to be included in the analysis. The user may select both the minimum and the maximum tree diameter to be included, or just one of the options, Figure 20.

1. Minimum Diameter choices:
   a) Select "None" to include the smallest tree.
   b) Select 2, 4, 6, 8,…up to 50" to exclude all trees below the selected diameter.

2. Maximum Diameter choices:
   a) Select "None" to include the largest dbh in the database.
   b) Select 20, 22, 24,…up to 80" to exclude all trees above the selected diameter.

If Min dbh is greater than Max dbh, an error message is displayed until the relationship is reversed.

Examples:

1. If it seems feasible to exclude diameters less than 10" because they don't provide volume that can be used at this time, set Min to 10, Max to None.

2. If trees over 60" dbh are valued for historic reasons and should not be included in the analysis, set Max = 60".

3. To calculate the volume and value of trees from 20 to 40 inches, set Min = 20", Max = 40".

4. To calculate volume and value of trees 10-20 inches and 40-50 inches, you will need to make two "runs" using Step 7 and Step 10 (discussed later). For run #1, set Min = 10, Max = 20. For run #2, set Min = 40, Max = 50, then combine the two results.

Sheet 5 OUTPUT PARAMETERS

Step 8. Indicate Species to be Excluded from Volume Calculation of Removal Trees

1. Enter the Species Code for any species you want excluded from Removal volume and value analysis (Figure 21). A maximum of 50 species may be excluded. See examples below for choices and results.

   Note, the more trees excluded, the longer the program will run to find alternate tree records.

2. The Species Code must be a valid code. If an invalid code is entered, an error message appears. This must be corrected before continuing.

3. Codes entered are of trees that will NOT be Included in removal volume and value calculations in forthcoming Steps 10-14.

Example:

Step 7 Dia Limits	Step 8 Excluded Trees	Step 9a Included Trees	Result
Min = 20 Max = None	a) 25 b) 14 c) 33	(blank)	The volume and value of species 25, 14 and 33 will be excluded from removal volume and value calculations. All other trees 20" and larger will be part of the analysis.

Sheet 5 OUTPUT PARAMETERS

Step 9a. Indicate Only the Species to be Included in Volume Calculation of Removal Trees

1. NOTE: The user cannot enter codes in BOTH Steps 8 and 9a. Doing so will cause multiple error messages to appear. They will remain until entries only appear in Step 8, Step 9a, or neither.

2. The Species Code must be a valid code. If an invalid code is entered, an error message appears. This must be corrected before continuing (Figure 21).

3. Species Codes entered in Step 9a are the ONLY species that will be used in the removal volume and value calculations.

4. The more trees entered in either Step 8 or Step 9a, the longer the program will run.

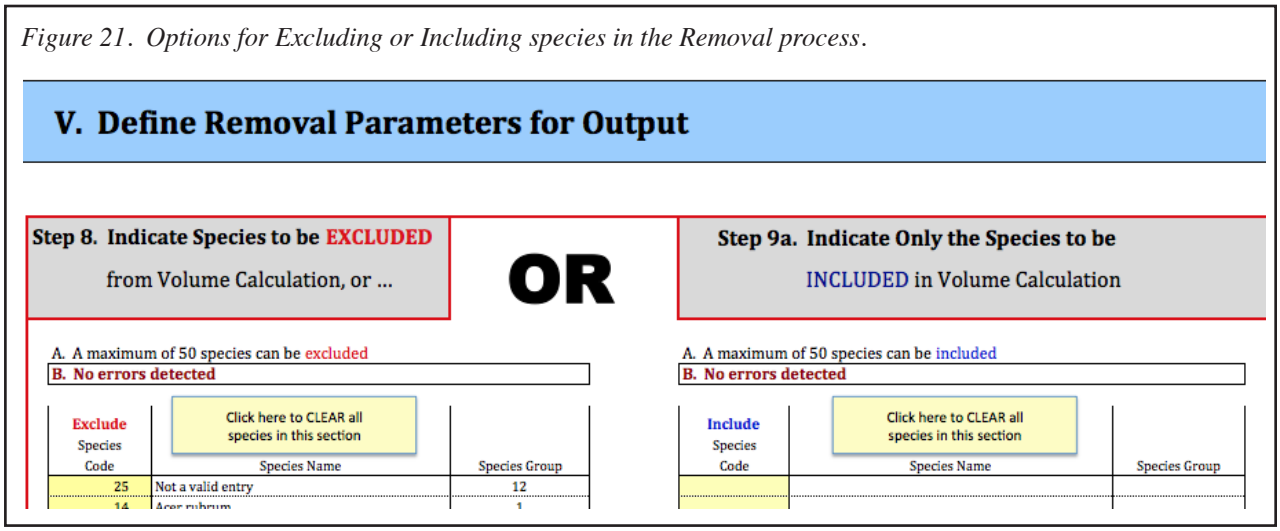
Examples:

Step 7 Dia Limits	Step 8 Excluded Trees	Step 9a Included Trees	Result
Min = 10 Max = 60	(blank)	a) 155	Quercus agrifolia trees (species code 155) that are 10-60" in diameter will be considered for removal, all other species will be excluded.

Step 7 Dia Limits	Step 8 Excluded Trees	Step 9a Included Trees	Result
Min = 20 Max = 50	a) 37 b) 84 c) 85	a) 143 b) 202	No result. Error messages are displayed. CUFIM does not allow trees to be excluded and included simultaneously.

In addition, there is a special case, called *100% Cut Pathway*, Step 9b, discussed on the MGT OPTIONS sheet, Steps 11a and 12a and in the Appendix.

Figure 21. Options for Excluding or Including species in the Removal process.



Sheet 6 REMOVAL

Step 10. Estimating the Annual Volume of Tree Removal by Randomly Selecting Trees, or by Field Marking Trees.

This sheet, along with the OUTPUT PARAMETERS sheet (Steps 7-9a), define how trees are selected for removal. Two methods of tree removal are discussed, Method 1 and Method 2.

This process allows the user to project the volumes that can reasonably be expected or anticipated to be lost each year. As noted at the beginning of Part III, trees selected from Method 1 are based on the percentage of trees expected to be removed each year, and are not the actual trees to be removed and no trees are actually deleted from the database.

The process for analyzing and evaluating the volume and value of trees removed each year can be accomplished through either of two methods.

1. Method 1 — Estimations Based on Removal Percentages. Due to budget constraints, timing or other reasons, it may be impractical to conduct the field surveys discussed in Method 2. In addition, there are a number of reasons why only a projected level of volume and value may be desirable.

a. From a general planning and logistics standpoint, it is necessary to determine the labor force size, equipment needed, and to establish a schedule for tree removal. It would be valuable to have an estimate of the magnitude of the task well before removal begins.

b. These estimates would also be valuable from a budget and policy perspective. Urban foresters may be asked to project expenses and incomes into the next several years. This information can be estimated here, providing planning offices, elected officials, and perhaps even voters requisite data needed to set policy and determine the budget for urban forestry operations.

c. It would be normal for the percentage of removal trees to vary from year to year. CUFIM-PRO allows the user to conduct a number of computer "runs" to obtain multiple estimates of projected volume and value for removal trees. This can be done by using several different percentages of expected tree removal in order to bracket the expected removal volumes and values. This approach would provide a "low," "average," and "high" value for consideration.

In Method 1, the user sets the expected percent of tree loss per year. Also, there are two variations available to the user, called Method 1A and Method 1B. Based on settings from either Method 1A or 1B, CUFIM-PRO computes the expected volume and value of removal trees.

2. Method 2 — Calculations based on Field Measurements. There are two scenarios that require field "marking" of trees.

First, if you have run CUFIM-PRO and used the results to convince your organization that now is the time for actual removal, then the next step is to send a crew to the field. They will do a risk assessment of each tree and mark it on a hard copy of the database.

In CUFIM-PRO those trees will be marked as removal trees and in fact if permission to proceed is obtained, they will be cut and deleted from the database by using the Delete Record function. If the number of trees cut is over, say, 100 or more, use the Export function (ID sheet) and edit the backup file. Be sure to immediately run the *Import Checker.xlsx* and re-import the file to CUFIM-PRO.

Secondly, the computer removal runs discussed above may be foregone and the direction is to physically mark trees on the ground and update their removal status in CUFIM-PRO, as described below.

Under either scenario, the urban forester actually marks trees on the ground for removal. Their volume and value is then calculated through a series of computations discussed in the following steps. To use Method 2, extensive field measurements and assessment data are required. This approach will provide a very accurate estimation of removal volume and value.

3. If field surveys are planned, follow these steps to enter the data in the spreadsheet.

a. In the field, record tree record number, tree location/address, Species Code and dbh for each removal tree -- trees based on the risk assessment.

b. In CUFIM-PRO select the DATABASE sheet, click on Sort the Tree Record Number, and locate Column 4, Removal.

c. Scroll down until the Tree Record Number is found, then check the species code and address to verify this is the correct tree.

d. Type the number "1" (without quotes) in the Removal column. Note that the cells are not yellow. This is the only time you are permitted to type in non-yellow cells.

e. Check the tree's dbh in column 7. If it is different than just measured in the field, update the dbh value using the Change Record function.

f. Continue this process until all trees are entered.

g. It is VERY IMPORTANT that you DO NOT run the Random Number Generator. Doing so will delete all your entries. Instead, proceed directly to Sheet 7, Step 11a.

The remainder of Part II discusses the details of how Methods 1A and 1B work.

Method 1A: Estimating Volume by Estimating the PERCENT of ALL Trees to be Potentially Removed Next Year

Select the REMOVAL sheet. There are three parts to Method 1A, Parts A, B, and C (see circled area on Figure 23 on the following page); they must be addressed in this order.

1. Part A: On the REMOVAL sheet, click Box A.

2. If selections on the OUTPUT PARAMETERS sheet (Steps 7-9a) are desired, they should be done before continuing with this section.

3. Part B: Select a value for "*Percent of All Trees to be Removed*." This is based on data from previous years, and/or knowledge about the coming year. See sidebar below for guidance.

4. Part C: Select Choice 1, 2 or 3.

Selecting Percent Removal for Method 1A

The choices available for percent of trees removed ranges from 0.5% to 10%. Selecting a realistic percentage is important. As discussed later in this section, 2% is commonly used, however, CUFIM will accept any number in this range. Selecting 10%, the upper end, means that 1/10th of the urban forest would potentially be removed in one year. Is this realistic?

This method follows the built-in curve shown as Figure 24. If larger percentages are needed for a particular dbh class, use Method 1B.

a. If diameter limits were not set in Step 7, then check Choice 1, "*No Dia. Limits*," then click on the Random Number Generator. The following two pages explains how the Excel spreadsheet and Random Number Generator work.

Method 1A, Choice 1: Example of output with short discussion.

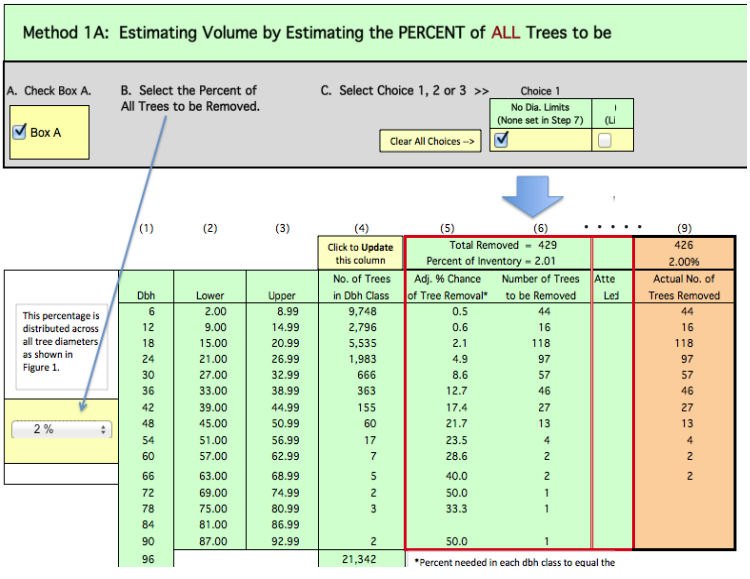
For this example, 21,342 trees were in the database, and the user requested information based on a 2% removal for the year resulting in 429 trees that would be removed (Figure 22, top of column 6). No limitations of the data were set on the OUTPUT PARAMETERS sheet.

Col 1:	Dbh class.
Cols 2 & 3:	Lower and upper limit to dbh class.
Col 4:	No. of trees currently in database, by dbh class.
Col 5:	Percent chance of tree being removed based on Figure 24 and user settings.
Col 6:	Based on percentages in column 5, the number of trees in each dbh class to be removed.
Cols 7 & 8:	Used when diameter limitations are set by the user on the OUTPUT PARAMETERS sheet; not shown on image below.
Col 9:	The "actual" number of trees found by the Random Number Generator per dbh class.

Choice 1 Discussion: The top of Column 9 shows that of the 429 trees requested for removal, 426 were selected, or 2.00%, the desired percentage. The desired number for dbh classes from 6 - 66" were met, however, for classes 72 - 90" they were not. These classes have very few trees (column 4

(continued after the next two pages)

Figure 22. Removal set-up columns.



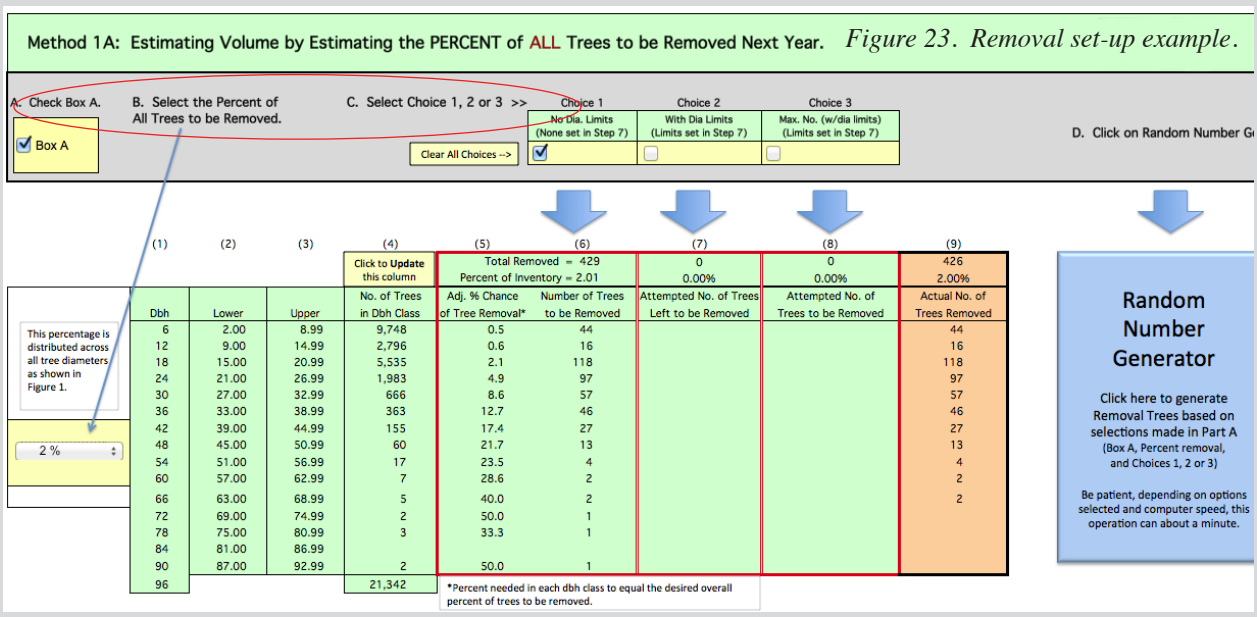


Figure 23. Removal set-up example.

How are the Number of Removal Trees per dbh Class Computed?

a. This example is when no diameter restrictions are placed on the calculation. The overall number of removal trees is calculated based on the overall percent of trees expected to be removed. (Example shown above: 21,342 trees in the database x 2% expected loss (from Part B) = 429 removal trees).

b. These 429 trees are distributed among the diameter classes based on the percent chance of tree loss shown in Figure 24 below (*Fig. 1. Percent Chance of Tree Removal;* Figure 1 in CUFIM-PRO). This graph is based on an equation that recognizes that larger trees are more likely to be removed than smaller ones, and are therefore, assigned a higher percent chance of removal.

Fig. 1. Percent Chance of Tree Removal

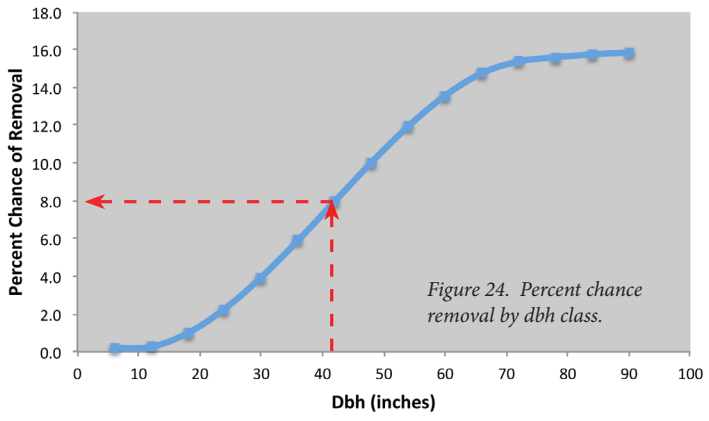


Figure 24. Percent chance removal by dbh class.

c. Follow the red arrows in Figure 24 and note that in the 42" dbh class, it is expected that 8% of the trees would be lost in a given year. The 42" dbh class contains 155 trees (column 4) x 8% = 12.40 trees to be marked for removal, not shown. This is done for each diameter class and summed. In this example, the total for all diameter classes is 198.61 trees (not shown).

d. The number of trees per diameter class in column 6 are proportionally decreased or increased, by a multiplier, to automatically provide the number of trees necessary to attain the desired overall percent that was selected in Part B (2% for this example). The multiplier, in this case, 2.160, is calculated from the number of removal trees (from (a) above) ÷ the total number of trees calculated by (b) above for all diameter classes, or 429 ÷ 198.61 = 2.160.

e. In this example, the number of removal trees in the 42" diameter class is increased from 12.40 to 27 trees to be marked for removal (2.160 x 12.40 = 26.78, or 27 trees) shown in column 6.

f. Last, the number of trees/dbh class to be removed in column 6 is divided by the total number in the corresponding dbh class (column 4) to arrive at the "*Adjusted Percent Chance of Tree Removal*" shown in column 5. For our 42" diameter class example, 27 ÷ 155 = 17.4%.

How Does the Random Number Generator Select Trees for Removal?

When the setup (Parts A, B and C on the REMOVAL sheet) is done, click on the Random Number Generator (RNG).

1. Doing so initiates a series of checks to be certain your setup is correct. If errors are found the user is notified and the routine is stopped.

2. If the setup is correct, the RNG calculates the number of trees per dbh class (Figure 22, column 4), if they haven't been previously determined, and sorts the DATABASE by diameter size.

3. For each dbh class, the RNG routine calculates the starting sequence number (column 1 on DATABASE sheet); it is labeled RxS, and the ending number is labeled RxE, where x is the number of the dbh class. In the example (Figure 22), the 6" dbh class would be R1S = 1 and R1E = 9748 (also see summary in Figure 25, left four columns).

4. The RNG loops through the first 14 dbh classes selecting random trees from each dbh class. For each dbh class it checks on user conditions that may have been set in Steps 7, 8 or 9a.

5. If the user Excluded trees of the current diameter loop on the OUTPUT PARAMETERS sheet, then the random number is discarded. If not,

6. It next checks to see if there are any trees on the Excluded Species list (OUTPUT PARAMETERS sheet), and if this tree is on the list. If so, the random number is discarded; if not,

7. A similar check is conducted to see if any trees are on the Included Species list. If no trees are present or if this tree is on the list, then,

8. This random number (tree sequence number) is a viable candidate and is stored in an array.

9. Because all trees are randomly selected it is not uncommon for a tree's sequence number to be selected more than once. To prevent this, three times as many trees as needed per dbh class are generated and, if all tests are passed, are stored in the array. Finally, all duplicate numbers are discarded leaving a unique set of random numbers per dbh class.

Random Number Generator

Click here to generate Removal Trees based on selections made in Part A (Box A, Percent removal, and Choices 1, 2 or 3)

Be patient, depending on options selected and computer speed, this operation can take about a minute.

10. This list, containing the number of trees desired in the first dbh class, is printed in a scratch column.

11. This process is repeated for all 14 dbh classes and each time the results are appended to the scratch column list, which is now the tree removal list.

12. When complete the tree removal list is sorted and each tree in the DATABASE sheet is compared to the removal list. If a match is made, a value of "1" is placed in column 4 on the DATABASE sheet. This continues until all desired trees per dbh class have been marked.

13. The total number of trees per dbh class from the removal list is printed in column 9, Figures 22 and 23.

14. The desired number to be removed, column 6, is plotted on the same graph as the actual number selected for removal, column 9, Figure 26. This provides a quick visual check on how close the computer run matched your request.

Running the Random Number Generator may take a minute with no restrictions set by the user. With restrictions it may take several minutes. Be patient and wait for it to finish the many calculations.

Seq. No. Start & End for All Trees in Database[D1S, D1E]

	Start	End	
Dbh	D xxx S	D xxx E	Diff
6	1	9,748	9,748
12	9,749	12,544	2,796
18	12,545	18,079	5,535
24	18,080	20,062	1,983
30	20,063	20,728	666
36	20,729	21,091	363
42	21,092	21,246	155
48	21,247	21,306	60
54	21,307	21,323	17
60	21,324	21,330	7
66	21,331	21,335	5
72	21,336	21,337	2
78	21,338	21,340	3
		-	
90	1	2	2

Start/End for Trees to be Removed [R1S, R1E]

Result of 3 "B" Choices	Result of 3 "A" Choices		Start	End	
		Dbh	R xxx S	R xxx E	Diff
0	44	6	1	44	44
0	16	12	45	60	16
0	118	18	61	178	118
0	97	24	179	275	97
0	57	30	276	332	57
0	46	36	333	378	46
0	27	42	379	405	27
0	13	48	406	418	13
0	4	54	419	422	4
0	2	60	423	424	2
0	2	66	425	426	2
0	1	72	427	427	1
0	1	78	428	428	1
0			0	0	
0	1	90	1	1	1

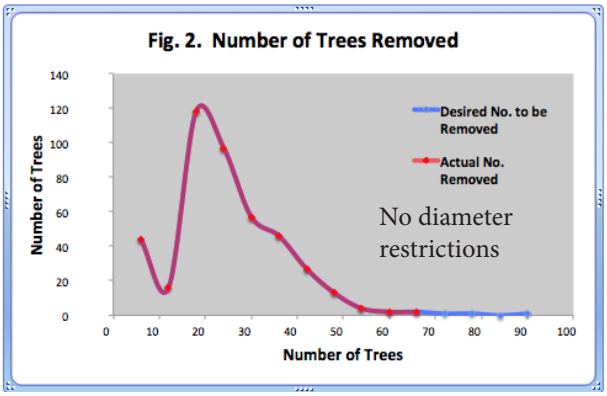
Figure 25. Sequence numbers for start and end of dbh classes.

shows only 2 or 3 trees/dbh class) and fewer yet that would be selected (column 6) and thus the likelihood of trees being randomly selected in these classes by Method 1 is very low.

If having more trees "marked" for removal is desired, set the percent of trees to be removed in Part B to a higher number (see item 3, this section).

Figure 26 shows provides a visual result of the computer run. The blue line, from column 6, is the number of removal trees requested while the superimposed red line is the result (column 9). As seen in Figure 22, the number of trees selected is identical to the requested number with the exception of a few trees in the highest diameter classes, discussed above.

Figure 26. Graphic showing both desired number of trees to be removed, and the actual number selected by the RNG.



Method 1A, Choice 2: Example of output with short discussion. For this example, diameter restrictions were placed on the OUTPUT PARAMETERS sheet such that the minimum dbh included was 10" and the maximum included was 60".

b. Because diameter limits were made in Step 7, check Choice 2 or 3. Choice 2 (Column 7) shows the number of trees to be included after diameter limits were imposed. This number (and percentage) is normally less, and sometimes significantly less, than the desired value (selected in Part B) due to the restrictions. However, if the number provided is satisfactory, then leave Choice 2 checked and run the Random Number Generator.

Method 1A, Choice 2 discussion: In the following image (Figure 27), Choice 2 was selected, column 7, resulting in a 1.78% removal versus the desired 2.00%. The Random Number Generator selected 379 trees instead of the 429 requested. If this "run" is good enough, go ahead to Sheet 7, MGT OPTIONS. If not, consider Choice 3.

The red line in the Choice 2 graph shows the diameter range selected, 10-60" (see Figure 28).

Figure 27. Example of setup for Choice 2, Method 1A.

Method 1A: Estimating Volume by Estimating the PERCENT of ALL Trees to be Removed Next Year.

A. Check Box A. ☑ Box A

B. Select the Percent of All Trees to be Removed.

C. Select Choice 1, 2 or 3 >>

Clear All Choices -->

	Choice 1	Choice 2	Choice 3
	No Dia. Limits (None set in Step 7)	With Dia Limits (Limits set in Step 7)	Max. No. (w/dia limits) (Limits set in Step 7)
	☐	☑	☐

This percentage is distributed across all tree diameters as shown in Figure 1.

2 %

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
			Click to Update this column	Total Removed = 429 Percent of Inventory = 2.01		380 1.78%	0 0.00%	379 1.78%
Dbh	Lower	Upper	No. of Trees in Dbh Class	Adj. % Chance of Tree Removal*	Number of Trees to be Removed	Attempted No. of Trees Left to be Removed	Attempted No. of Trees to be Removed	Actual No. of Trees Removed
6	2.00	8.99	9,748	0.5	44	0		
12	9.00	14.99	2,796	0.6	16	16		16
18	15.00	20.99	5,535	2.1	118	118		118
24	21.00	26.99	1,983	4.9	97	97		97
30	27.00	32.99	666	8.6	57	57		57
36	33.00	38.99	363	12.7	46	46		46
42	39.00	44.99	155	17.4	27	27		27
48	45.00	50.99	60	21.7	13	13		13
54	51.00	56.99	17	23.5	4	4		4
60	57.00	62.99	7	28.6	2	2		1
66	63.00	68.99	5	40.0	2	0		
72	69.00	74.99	2	50.0	1	0		
78	75.00	80.99	3	33.3	1	0		
84	81.00	86.99						
90	87.00	92.99	2	50.0	1	0		
96			21,342					

*Percent needed in each dbh class to equal the desired overall

Adjustment Ratio = 1.13

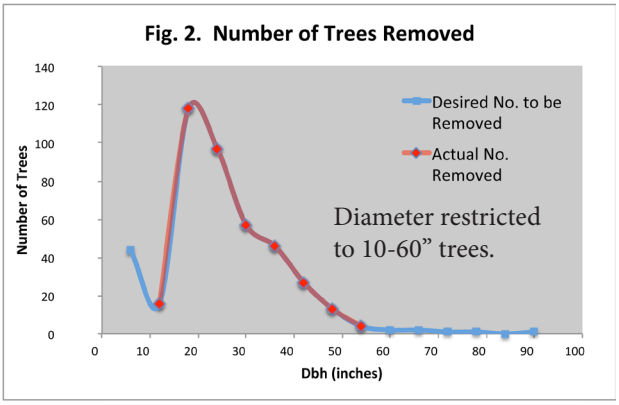


Figure 28. Graphic for Choice 2, with diameter restriction.

Method 1A, Choice 3 example:

On the other hand, if the number of removal trees (and percent) in Choice 2 is considered too low, try Choice 3 (Column 8). CUFIM-PRO increases the number of removal trees in each diameter class by an amount that, when summed, provides the desired percent initially selected in Part B.

In Figure 29, column 8, note the number of trees has been adjusted to equal the requested percentage, 2%, and that the computer run provided exactly that percentage and corresponding number of trees, column 9.

Figure 30, "*Number of Trees Removed*" shows both the total number of trees to be removed to achieve the desired percent (blue line, from column 6), and the actual number that were marked for removal by the random number generator (red line, from column 9). This provides a visual picture of desired versus actual for the computer run.

Note the red line, the actual number of trees selected for removal, is higher than what is shown for "*No Dia Limits*" in column 6 due to the dbh restriction of 10-60". CUFIM-PRO added more trees to the remaining dbh classes to compensate and produce a 2% removal rate.

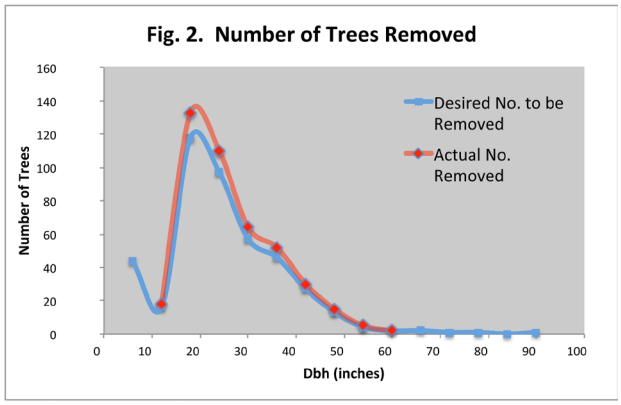


Figure 30. Graphic for Choice 3, with diameter restriction.

For various reasons however, selecting Choice 3 may or may not provide the desired removal percentage.

Figure 29. Example of setup for Choice 3, Method 1A.

Method 1A: Estimating Volume by Estimating the PERCENT of ALL Trees to be Removed Next Year.

A. Check Box A. ☑ Box A

B. Select the Percent of All Trees to be Removed.

C. Select Choice 1, 2 or 3 >> Clear All Choices -->

	Choice 1	Choice 2	Choice 3
	No Dia. Limits (None set in Step 7)	With Dia Limits (Limits set in Step 7)	Max. No. (w/dia limits) (Limits set in Step 7)
	☐	☐	☑

This percentage is distributed across all tree diameters as shown in Figure 1.

2 %

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
			Click to **Update** this column	Total Removed = 429 Percent of Inventory = 2.01		380 1.78%	429 2.01%	429 2.01%
Dbh	Lower	Upper	No. of Trees in Dbh Class	Adj. % Chance of Tree Removal*	Number of Trees to be Removed	Attempted No. of Trees Left to be Removed	Attempted No. of Trees to be Removed	Actual No. of Trees Removed
6	2.00	8.99	9,748	0.5	44	0	0	
12	9.00	14.99	2,796	0.6	16	16	18	18
18	15.00	20.99	5,535	2.1	118	118	133	133
24	21.00	26.99	1,983	4.9	97	97	110	110
30	27.00	32.99	666	8.6	57	57	64	64
36	33.00	38.99	363	12.7	46	46	52	52
42	39.00	44.99	155	17.4	27	27	30	30
48	45.00	50.99	60	21.7	13	13	15	15
54	51.00	56.99	17	23.5	4	4	5	5
60	57.00	62.99	7	28.6	2	2	2	2
66	63.00	68.99	5	40.0	2	0	0	
72	69.00	74.99	2	50.0	1	0	0	
78	75.00	80.99	3	33.3	1	0	0	
84	81.00	86.99						
90	87.00	92.99	2	50.0	1	0	0	
96			21,342					

*Percent needed in each dbh class to equal the desired overall percent of trees to be removed.

Adjustment Ratio = 1.13

1) If it is possible (as shown in Figure 29), this will be evident by observing the "*Total Removed*" and "*Percent of Inventory*" values for Choice 3 (column 8). They will be the same as for Choice 1 (column 6) except for excluded dbh classes where empty cells occur in column 9.
2) If it is not possible, the "*Total Removed*" and "*Percent of Inventory*" values will still be lower than Choice 1, but higher than Choice 2. If these values are acceptable, continue with Choice 3.
3) If neither Choice 2 or 3 are acceptable, you may decide to revise the initial "*Percent of All Trees to be Removed*" in Part B, or relax restrictions imposed on diameter in Step 7.

Calculating the Actual Number of Trees for removal using the Random Number Generator is fully explained in the sidebars on previous pages.

Regardless of whether Choice 1, 2 or 3 is selected, the actual number of trees "marked" for removal (column 9) may be lower, and depending on user restrictions considerably lower, than the desired number. If restrictions are too limiting, there simply won't be enough trees left in unrestricted dbh classes to make up the difference. An example would be restricting the diameter to trees 20" and greater when most of the urban forest trees are under 20". It's important to become familiar with the forest composition. The various statistics and graphs provided will help.

Also, some dbh classes may only have a few trees (say, 1-5) in them. The Random Number Generator may never select these trees as there is only a few chances to "capture" them. Thus, if removing these trees is important, it would be more realistic to simply add those volumes and value to the CUFIM-PRO analysis.

Your choices for increasing the number of removal trees are essentially the same as those discussed in item 3) above.

The suggested approach to a successful Removal run is:

a. Initiate the first computer run with the percent selected (in Part B) that you believe is most likely to occur, and then,
b. Bracket the percentage used in Part B by selecting a smaller, and then a higher percentage.

Note that entering Species Codes in Step 8 or Step 9a (Excluding or Including specific species) does not affect the number of trees selected for removal. If CUFIM-PRO generates a random number of a tree that was Excluded (Step 8), or is not Included (Step 9a), it immediately generates a new number and continues doing so until it finds a number that matches an included species.

3. Additional Notes on Method 1A

a. If the range of acceptable diameters, set by min and max in Step 7, is too narrow, most, if not all trees in the remaining size classes will be selected for removal. This may invalidate any analysis and projection of volume removal and value. Some practice with setting output parameters is needed, as well as caution when interpreting results.

b. Note that CUFIM-PRO uses 6" diameter classes to determine tree removal volume and value. This is too wide an interval for field measurements, but is fine for estimating removal, since trees of all diameters in the class have an equal chance of being selected.

c. Running the Random Number Generator does not provide an immediate response. A huge number of calculations are being performed. The operation can take from 1-5 minutes depending on computer speed, size of database, and output settings of diameter limitations, and exclusions or inclusions of species. However, about 10-30 runs could be made in an hour on the database providing valuable information about your urban forest that could not be obtained any other way.

d. As seen in Figure 24, the probability of removal varies from about 0.2% for 6" dbh trees up to about 16% for 90" trees. These values were derived from a community's urban forest database of 22,221 trees conditioned to provide 2% of the total number of trees when summed over all dbh classes, and to represent the normal S-shaped curve. Two percent was selected based on the average age of an urban tree before removal. If a community's trees are "all-aged" from 1-50 years, then each year about 2% will be removed (1/50th of the trees are removed per year x 100). Two percent is also a number used by some urban foresters in California.

This concludes the presentation and discussion of Method 1A. It is the easiest of the two approaches as the percent removal by dbh class is pre-determined and represents a condition found in many urban forests.

On the other hand, if the user finds Method 1A too restrictive, Method 1B allows greater flexibility through a user-defined approach.

Method 1B: Estimating Volume by Estimating the PERCENT of Trees by DBH CLASS to be Removed Next Year.

The purpose of Method 1B is to allow the urban forester the flexibility of entering their own estimates of loss by diameter class. This requires accurate knowledge and a good understanding of the urban forest structure and composition.

To learn more about the urban forest makeup, the MGT OPTIONS worksheet (Sheet 7) offers summary tables and graphs for both trees selected for removal, and for the entire urban forest. Those Steps are not part of the Removal process but provide valuable information for making the decisions on the OUTPUT PARAMETERS and REMOVAL sheets.

Working with Method 1B is very similar to 1A, except for the selection of the percentage of removal trees, therefore, to benefit from Method 1B, it is important to understand and work with Method 1A first.

There are three parts to this section.

1. Part A: On the sheet titled REMOVAL, Click Box B. Selecting Box B automatically clears Box A and all Method 1A settings.

2. Part B: The procedure used in Method 1B is the same as for Method 1A except for the approach used to determine Percent Chance of Tree Removal (column 5, Figure 31). Values for Method 1A were based on a statistical model, the values for Method 1B are based on the knowledge and experience of the user. The cells in column 5 are yellow, allowing the user to enter an estimate of the percent chance of removal per size class based on data from prior years, and expected conditions for next year.

Having this option available is a powerful tool, but can easily be misused, even abused. Entering

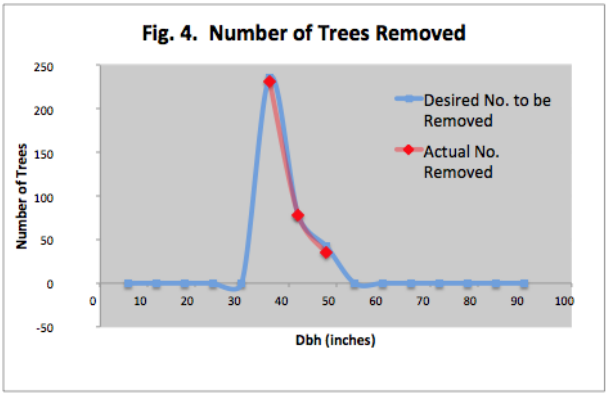


Figure 32. Graph for Choice 2, Method 1B; contiguous segment of dbh classes.

(1)	(2)	(3)	(4)	(5)	(6)		(9)
			Click to **Update** this column	Total to be Removed = 356 Percent of Inventory = 1.67			344 1.61%
Dbh	Lower	Upper	No. of Trees in Dbh Class	Percent Chance of Tree Removal	Number of Trees to be Removed	Attr. of Leoved	Actual No. of Trees Removed
6	2.00	8.99	9,748		0		
12	9.00	14.99	2,796		0		
18	15.00	20.99	5,535		0		
24	21.00	26.99	1,983		0		
30	27.00	32.99	666		0		
36	33.00	38.99	363	65.0	236		231
42	39.00	44.99	155	50.0	78		78
48	45.00	50.99	60	70.0	42		35
54	51.00	56.99	17		0		
60	57.00	62.99	7		0		
66	63.00	68.99	5		0		
72	69.00	74.99	2		0		
78	75.00	80.99	3		0		
84	81.00	86.99					
90	87.00	92.99	2		0		
96			21,342				

Figure 31. Example of setup for Choice 1, Method 1B, a contiguous segment of dbh classes.

realistic values is required for successful computer runs. CUFIM-PRO can easily handle "removal" of 5,000 trees in the database, and in some cases, 7,000. However, above 5,000 the RNG routine will likely stop calculating, reset and display an error message.

But, think about it. If the database contained 45,000 trees, removing 5,000 in one year, over 10%, would be absurd from a general planning standpoint. Therefore, an important number to watch is the value above column 6, *"Total to be removed = ___."* If this number is unrealistically high, warning signs pop-up to alert the user.

If a high percent change of removal is entered for a dbh class with thousands of trees, the longer the program must run as it keeps working to find acceptable trees, perhaps thousands of them, to fit specified conditions.

On the plus side, larger percent values can be assigned to specific dbh classes where large removals are desired. For Figure 31 and the graph in Figure 32 (previous page), if a certain species of tree in the diameter range of 33-51 inches (classes 36, 42 and 48") were deemed unsafe, a large percent removal value can be requested for that range, as shown. This could not be accomplished by Method 1A.

In a similar manner, the removals could come from two non-contiguous segments of the dbh range, as shown in the second set of charts, Figures 33 and 34. This setup cannot be completed via Method 1A, either.

Last, by leaving the percent chance removal blank for any dbh class (column 5) effectively limits removals to non-zero classes. This only works for 6" class intervals; if you need greater precision over

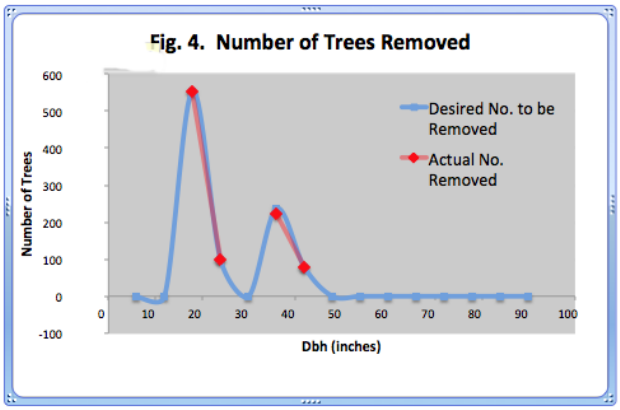


Figure 34. Graph for Choice 1, Method 1B; non-contiguous dbh classes.

Figure 33. Example of setup for Choice 1, Method 1B, a non-contiguous series of dbh classes.

Method 1B: Estimating Volume by Estimating the PERCENT of Trees by DBH CLASS to be Removed Next Yea

A. Check Box B — Box B

B. Enter Percent Chance by DBH Class

C. Select Choice:

	Choice 1	Choice 2	Choice 3
	No Dia. Limits (None set in Step 7)	With Dia Limits (Limits set in Step 7)	Max. No. (w/dia limits) (Limits set in Step 7)
Clear All Choices -->	☑	☐	☐

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
			Click to **Update** this column	Total to be Removed = 967 Percent of Inventory = 4.53		0 0.00%	0 0.00%	955 4.47%
Dbh	Lower	Upper	No. of Trees in Dbh Class	Percent Chance of Tree Removal	Number of Trees to be Removed	Attempted No. of Trees Left to be Removed	Attempted No. of Trees to be Removed	Actual No. of Trees Removed
6	2.00	8.99	9,748		0			
12	9.00	14.99	2,796		0			
18	15.00	20.99	5,535	10.0	554			554
24	21.00	26.99	1,983	5.0	99			99
30	27.00	32.99	666		0			
36	33.00	38.99	363	65.0	236			224
42	39.00	44.99	155	50.0	78			78
48	45.00	50.99	60		0			
54	51.00	56.99	17		0			
60	57.00	62.99	7		0			
66	63.00	68.99	5		0			
72	69.00	74.99	2		0			
78	75.00	80.99	3		0			
84	81.00	86.99						
90	87.00	92.99	2		0			
96			21,342					

the diameters included or excluded this should be done inconjunction with the dbh inclusions options available on the OUTPUT PARAMETERS sheet, Step 7, using the minimum and maximum menus to control diameter.

The percent chance values entered in column 5 are graphically shown at the bottom of the screen ("*Percent Chance of Removal*"), not shown here. The y-axis, Percent Chance, will automatically adjust if higher values are entered. This is interactive and the user can see the trend change as values in column 5 are entered.

And as with Method 1A, graphs of the number of trees removed for both desired and actual are provided.

3. Part C: Select Choice 1, 2 or 3, and click on the Random Number Generator.

Print options and "jump" boxes are available for quickly jumping back to the OUTPUT PARAMETERS sheet to set diameter limitations, and forward to the next sheet.

This concludes the presentation and discussion of *Estimating Volume by Estimating the Percent of Trees* by Methods 1A and 1B.

In the next two sections you will see the results of decisions made here.

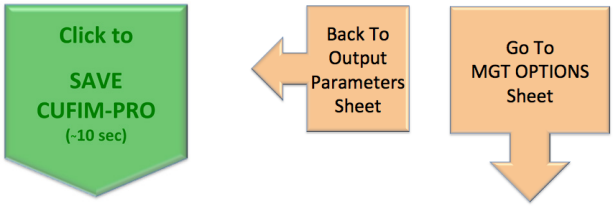


Figure 35. Examples of action or "jump boxes."

Sheet 7 MGT OPTIONS

Step 11a. Creating Stand Tables for Removal Trees.

There are two approaches relating to management options that will be discussed in this section. First, the more general sampling approach resulting from the REMOVAL sheet choices has been discussed in Steps 7, 8, 9a, and 10. This approach is a planning method where the urban forester is making estimates that lead to information about the volume and value of annual cuttings to remove dead, damaged and dying trees that would occur in a year or more. Secondly, a more specific approach will be discussed that considers a case where it is known in advance that 100% of a specific diameter range will be cut or that a particular species has failed and must be removed soon.

A. General Planning Approach

By way of review it has been shown that the user may have set diameter restrictions in Step 7, may have Excluded some species in Step 8, or, only allowed one or more species into the analysis in Step 9a (Inclusion list), and on the REMOVAL sheet made estimates of the annual tree removal percentage to arrive at the number of trees "marked" for removal. Now we further examine the selected Species Groups through the use of Stand and Stock tables.

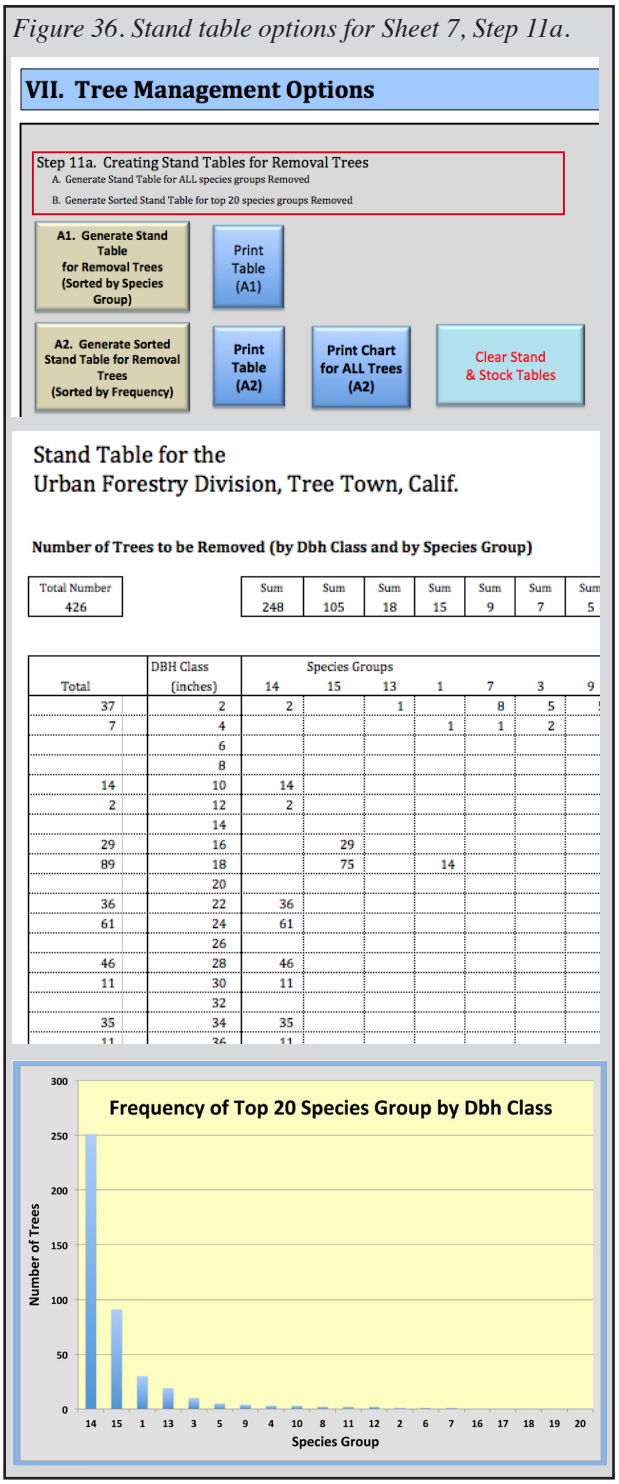
Stand Tables are tables that show the number of trees by species group, and by diameter class. The diameter class interval is set at 2 inches.

The purpose of a stand table is to set in table form the species group containing the largest number of trees, second largest, and so on, by dbh class, Figure 36. This allows the urban forester an understanding of the makeup of the forest, and along with the stock table, discussed ahead, a good understanding of the distribution of both numbers and volumes of trees.

Be aware that species groups are not the same as a species codes. Remember, all species in the urban forest must be assigned to a "group," the one that most closely resembles its volume characteristics.

1. Click Box A1 to generate a stand table for removal trees "marked" by Method 1A, 1B or Method 2 on the REMOVAL sheet. The stand table generated by Box A1 summarizes the number of trees by dbh class and species group that would be removed; a subset of the entire database (Figure 36).

Figure 36. Stand table options for Sheet 7, Step 11a.



2. If Removal Method 1A or 1B was used, the tree numbers in the stand table are those marked with a "1" in column 4 (DATABASE sheet). They were selected by the random number generator and are estimated or projected values. However, if the marked trees are based on a field assessment

and entered in the database according to Method 2, these numbers are actual trees to be removed.

3. Select "*Print Table (A1)*" to print the table generated by action box A1.

4. Normally the majority of trees to be removed will fall in only a few of the species groups. Clicking on action box A2 will sort the top 20 species groups by the frequency of trees each contains. Thus the groups that most contribute to the total number of trees to be removed can easily be detected.

Scroll down to about row 156 to see the tabular results from Box A2.

5. Select "*Print Table (A2)*" to print the top 20 table generated by Box A2.

6. A graph showing the top 20 species group, sorted from most frequent to least is provided starting about row 300. To print the chart (full page) of the information summarized from action box A2, click on *"Print Chart (A2)."*

7. If you wish to clear all tables, click on "*Clear Stand & Stock Tables*," back at the top.

Sheet 7 MGT OPTIONS

Step 12a. Creating Stock Tables for Removal Trees

Note that Step 11b was skipped, as will Step 12b. See next page for an explanation of Steps 11b and 12b.

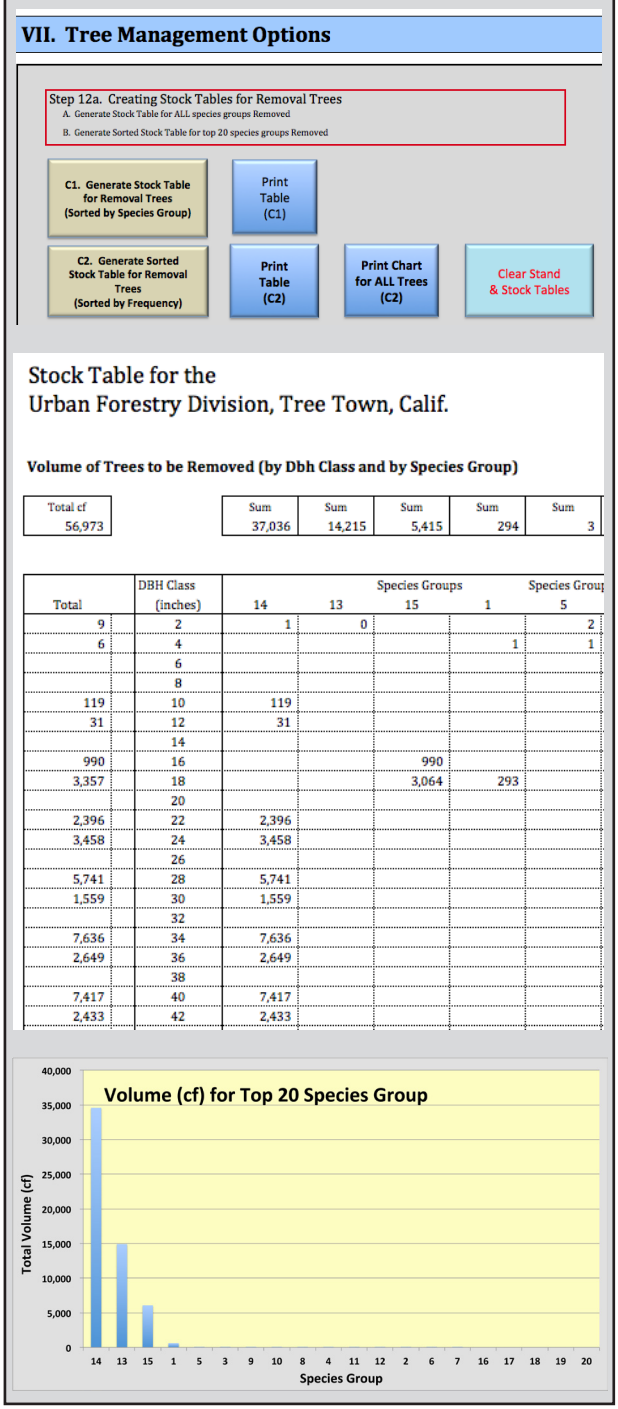
Stock Tables are tables that show the total volume in cubic feet of trees by species group, and by diameter class. The diameter class interval is fixed at 2 inches.

1. Select action Box C1 to generate a stock table for removal trees "marked" by Method 1A, 1B or Method 2 on the REMOVAL sheet. The table summarizes the volume of trees by dbh class and species group that would be removed (Figure 37).

2. If Removal Method 1A or 1B was used, the tree volumes in the stock table are those selected by the random number generator and are estimated or projected values. However, if these are trees that were marked in the field and entered in the database according to Method 2, these volumes are actual volumes to be removed.

3. Select "*Print Table (C1)*" to these print results.

Figure 37. Stock table options for Sheet 7, Step 12a.



4. Normally the majority of the volume to be removed will fall in only a few of the species groups. Clicking on action box C2 will sort the top 20 species groups by the amount of volume each species group contains (Figure 37). Thus the groups that most contribute to the total volume to be removed can easily be detected. Scroll down to about row 256 to see the top 20 table.

Stand and Stock Tables for the Entire Database - NOT part of Removal Process

NOTE: If you are doing planning "runs" or working with Removal trees, skip Steps 11b and 12 b. Steps 11b and 12b are stand-alone steps and not part of the tree removal process. Use these steps to create stand and stock tables for the entire database. The products generated from these steps are the same as shown for Steps 11a and 12a.

Urban foresters can use stand and stock tables to gain an understanding of the structure and composition of the urban forest and the distribution of volume. They provide a concise summary of the number of trees and volume by species groups and diameter class, and give a picture of the overall forest.

Sheet 7 MGT OPTIONS

Step 11b. Creating Stand Tables for ALL Trees in Database

This step is NOT part of the Removal procedure, and should be skipped if you are currently working on Removal trees. This step includes ALL records of ALL species groups and ALL diameter classes, regardless of settings made in steps 7-10. This provides an excellent picture of the diameter distribution in the urban forest, but is not designed to evaluate removal trees.

1. Clicking action box B1 allows the user to develop a Stand Table for ALL trees in the inventory database.

2. Clicking B2 sorts, by species group, the top 20 groups (groups having the greatest number of trees).

3. Print boxes allow for hard copies of tables generated from action boxes (B1 and B2), and for a print out of a chart derived from action box B2.

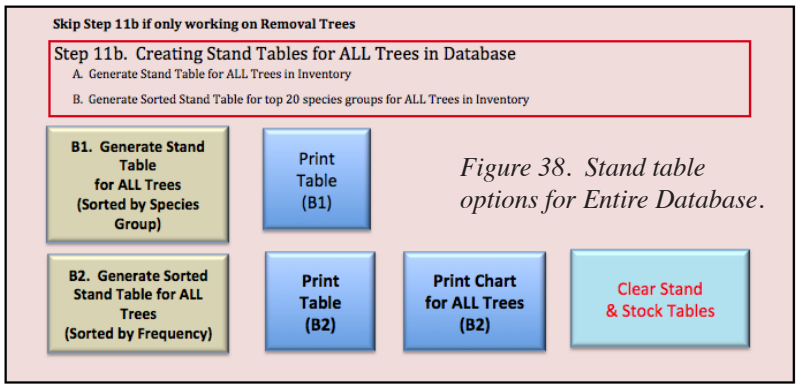


Figure 38. Stand table options for Entire Database.

Sheet 7 MGT OPTIONS

Step 12b. Creating Stock Tables for ALL Trees in Inventory

This step is NOT part of the Removal procedure, and should be skipped if you are currently working on Removal trees. This step includes ALL records of ALL species groups and ALL diameter classes, regardless of settings made in Steps 7-10. This provides an excellent picture of the volume distribution of the urban forest, but is not designed to evaluate removal trees.

1. Clicking action box D1 allows the user to develop a Stock Table for ALL trees in the inventory database.

2. Clicking action Box D2 sorts, by species group, the top 20 groups (groups having the greatest volume).

3. Print boxes allow for hard copies of tables generated from boxes (D1 and D2), and for a print out of a chart derived from box D2.

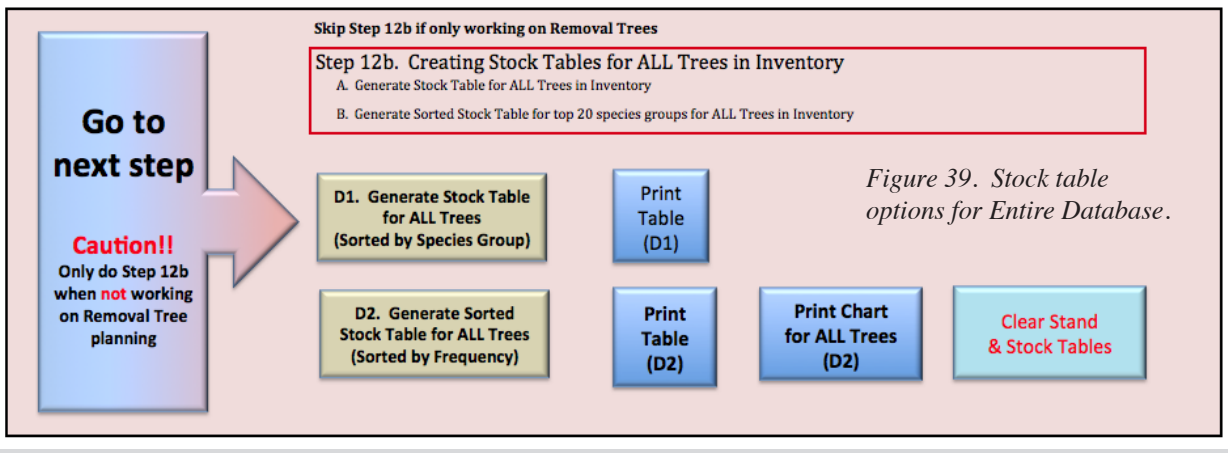


Figure 39. Stock table options for Entire Database.

5. Select "*Print Table (C2)*" to print these results.

6. To print a chart (full page) of the information summarized from action box C2, click on "*Print Chart (C2)*." Scroll down to about row 300 to view the chart.

7. If you wish to clear all tables, click on "*Clear Stand & Stock Tables*," back at the top.

What is the value of Stand and Stock tables?

Stand and stock tables answer many questions about the urban forest. They would provide important information to potential bidders to help them accurately assess the labor and equipment requirements of the job. Below are a few questions that stand and stock tables could help answer.

1. How many large, medium and small sized trees are slated for removal?

2. Is the volume in line with the number of trees per dbh class or are there a disproportionate number of small trees which have little volume?

3. When planning for actual removal in the field, will the distribution by diameter and volume favor a higher or lower price? Normally larger trees have greater value.

4. Do the various wood cutting outfits have equipment to handle the number of large trees being removed, or if most removal trees are small, could the city contract with smaller, less expensive outfits to remove the wood?

5. Are the species groups of the quality of wood that will command a higher price, such as for lumber, or will they be sold as firewood?

6. Will the cleanup of small material from many small trees be more costly than the value of the wood?

B. A Specific Approach - The 100% Cut Pathway

The Removal process discussed to this point is invaluable from a planning perspective for the many reasons discussed above. That approach will be frequently used by the urban forester as she/he prepares annual work plans, seeks financial support and lines up labor, equipment and deals with many other aspects of the process. But what if 100% of the trees of a specific diameter or species must be removed?

There are several situations where very specific information is needed to assess an immediate removal event. Perhaps Sudden Oak Death disease (*Phytophthora ramorum*) has spread into your community's oak stands and the trees must be removed before trees in neighboring towns are infected. Or, a deep, long-term freeze has killed all trees under a specific size, and 100% must be removed. In fact, there is a case study, *The Big Freeze*, on this scenario presented in Part IV.

Now the management approach is not one of planning for the future, but assessing for the present.

A variety of tools are available that allow the urban forester to become intimate with the composition of the forest. Up to this point the focus has been on Species Groups, now the focus will drill down a layer deeper, to the Tree Species level.

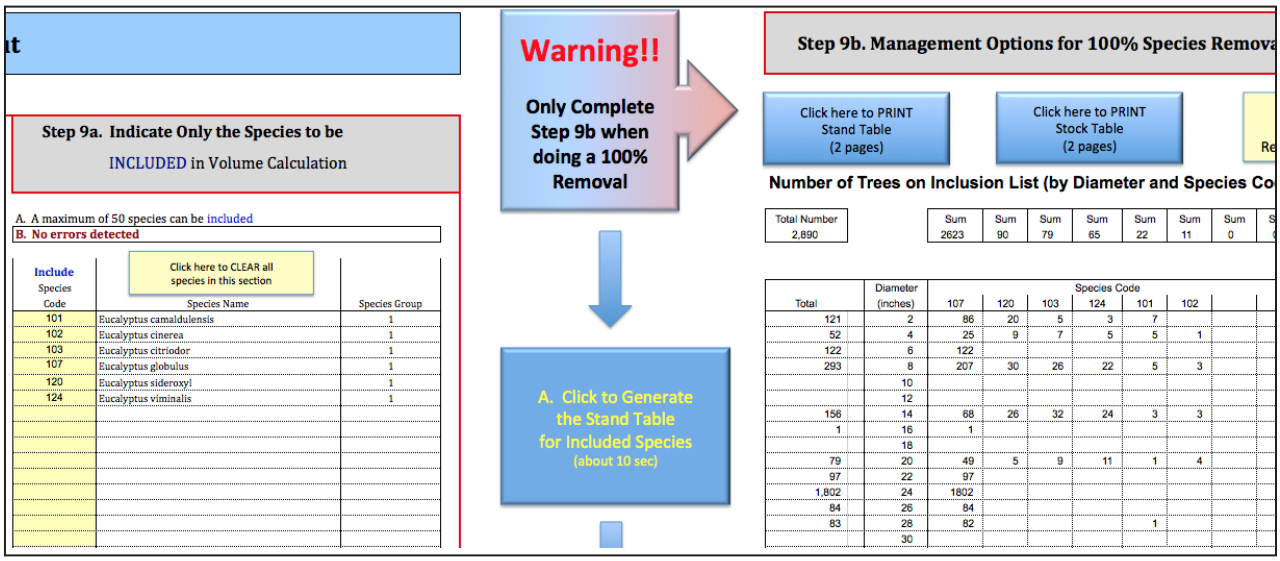
To accomplish this, on the OUTPUT PARAMETERS Sheet, Step 9b, *Management Options for 100% Species Removal by Species and Diameter*, is available.

A case study example with greater detail is presented following Part IV, however, here an overview of the process is described.

Through this approach the user, first, makes a decision about the diameter range necessary for removal (Step 7). Step 8, the Exclusion List, is not used with the *100% Cut Pathway*. Next, the species codes (up to 50) are typed into the Included species box (Step 9a), and then both Stand and Stock tables are generated, Figure 40, in Step 9b. However, unlike those in Step 11a and 12a, these tables show information for Species, not Species Groups. The user has now generated information for the listed species and the tables show tree numbers by diameter class by species, and volume by diameter class by species.

If more than one iteration is needed the process is repeated with different restriction settings. Finally, to identify the subset of trees to be removed, a click on the *Click to Mark All Trees on Inclusion List for Removal in Database* action box will mark those species on the DATABASE sheet. An option to print a report of all trees marked for removal is available.

Figure 40. Step 9a and 9b. Management options for 100% Removal.



Sheet 8 VALUATION

Step 13. Preliminary Valuation - Same Value Method and Assign Value by Species

The purpose of establishing value for urban trees is the primary objective of this project. Without a known value, urban foresters cannot plan budgets or have a say in policy considerations. In other words, this is an estimate of the potential value of removal trees for the year.

The "*Same Value Method and Assign Value by Species*" is used for situations when the unit value of removed volume can be considered equal regardless of species group, species, or tree size. It is not necessary to calculate the stand and stock tables prior to calculating value in this step, however, doing so provides valuable management information to the user and allows a more informed set of parameters on the OUTPUT PARAMETERS and REMOVAL sheets. The volume and value data that are obtained from this section are based on restrictions imposed in Steps 7-9a and on Choices 1, 2 or 3 checked on the REMOVAL sheet under Method 1A, 1B, or Method 2.

1. As shown in Figure 41, create or update the species list by clicking on the action box titled "*Click here to Create Current Species List, and Species Groups*." (Step 1).

2. Check the action box for Method 1 to create and setup new spreadsheet columns, Step 2.

3. There are six units of wood measurement available as shown in Figure 41, Step 2, and in the Wood Valuation table (Figure 42). Most often, firewood/cordwood will be selected as there are a number of wood cutting outfits in most communities that could bid on the job. However, selling by green weight or dry weight (air dried for about six months) is easily done and requires less handling and stacking. The values in the firewood group on the Wood Valuation chart are approximately equal for each unit of measurement. The table is based on "standing or cut tree measure" in units of cubic feet as the volume equations are set for cubic feet. Thus, a value of $5.60/cf is approximately equal to $350/standard cord, $420/ton dry weight, and $210/ton green weight. There is no equivalent between firewood and lumber products; lumber values are scaled to provide a range from which to choose.

The equivalent conversion values are on the VALUATION sheet, cells AN128:AQ128. These cells are in yellow and these constants could be changed. <u>They should only be changed based on hard scientific data; any other change could nullify all calculations on the VALUATION sheet.</u>

Presently there are few hardwood lumber suppliers buying or selling in either rough or finish lumber, but hopefully information about a communities urban forest provided by CUFIM-PRO would spark an interest.

There is also a range of values for each measurement unit that will assist the urban forester to assess the value of the expected annual loss of wood. For

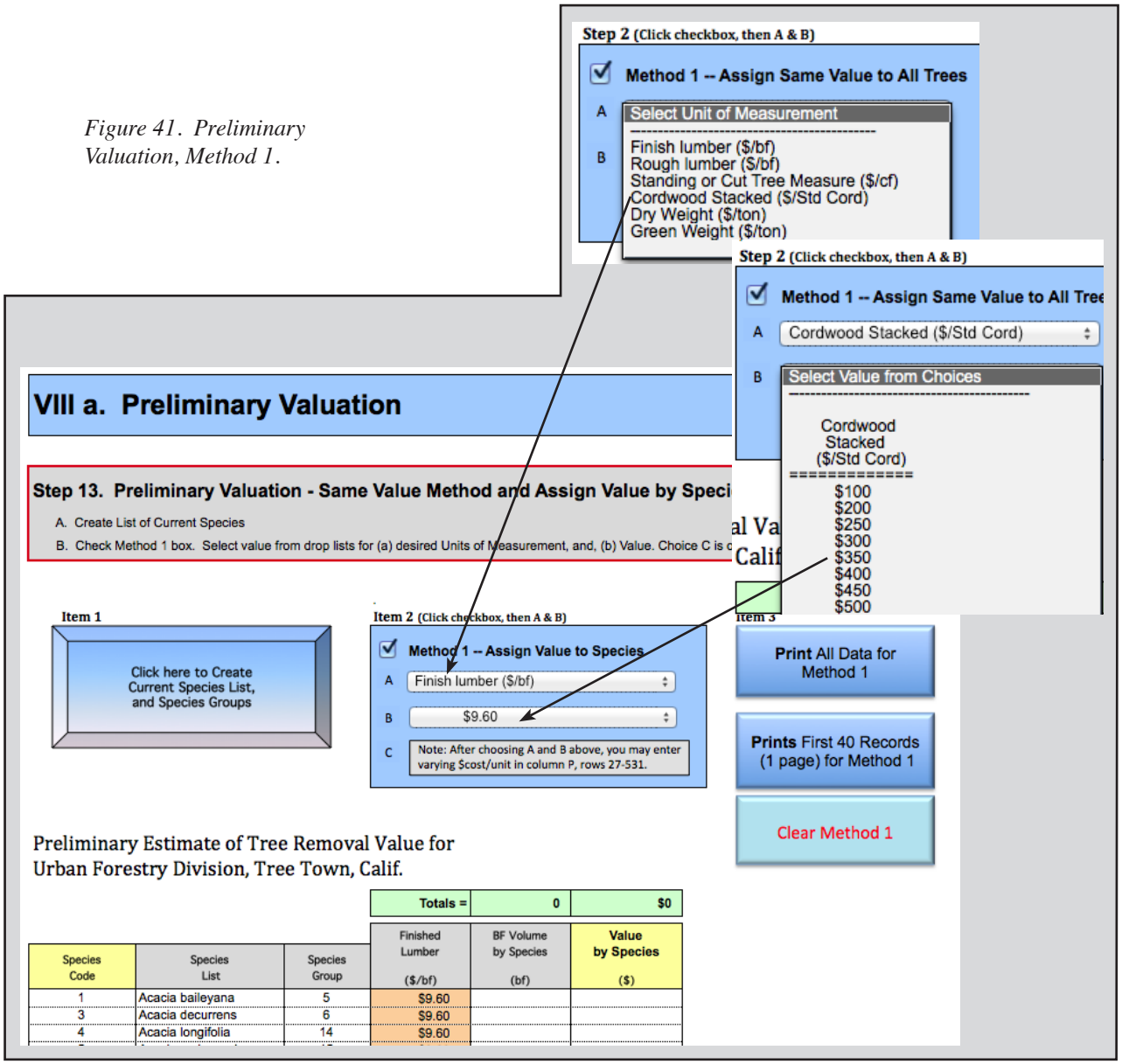


Figure 41. Preliminary Valuation, Method 1.

Figure 42. Wood valuation chart.

Wood Valuation

		7.0 bf/cf	62.5 cf/std cord*	26.5 lbs/cf	53.0 lbs/cf*
Wood Products		Firewood			
Finish lumber ($/bf)	Rough lumber ($/bf)	Standing or Cut Tree Measure ($/cf)	Cordwood Stacked ($/Std Cord)	Dry Weight (dried 6 months) ($/ton)	Green Weight (not dried) ($/ton)
3.20	2.40	$1.60	$100	$120	$60
6.40	4.80	$3.20	$200	$240	$120
8.00	6.00	$4.00	$250	$300	$150
9.60	7.20	$4.80	$300	$360	$180
11.20	8.40	$5.60	$350	$420	$210
12.80	9.60	$6.40	$400	$480	$240
14.40	10.80	$7.20	$450	$540	$270
16.00	12.00	$8.00	$500	$600	$300

* Based on study by Pillsbury and Stephens, 1978, "*Hardwood Volume and Weight Tables for California's Central Coast.*" Calif Department of Forestry and Fire Protection, Sacramento, CA.

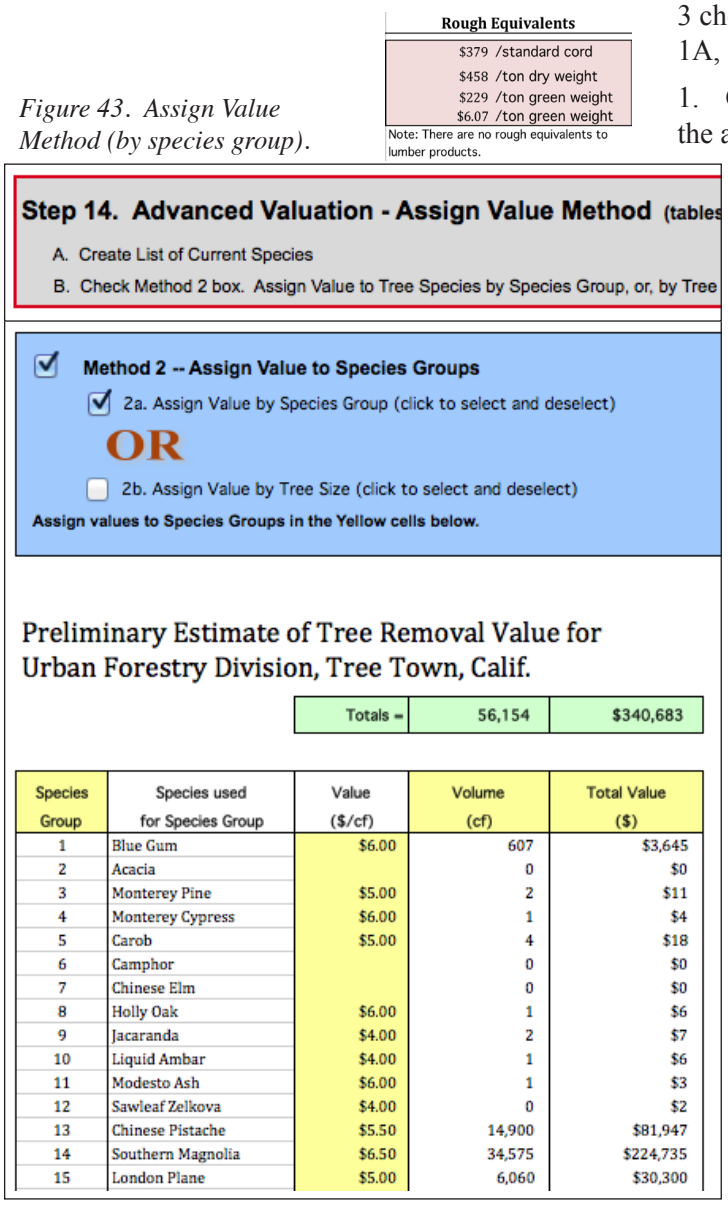
this step, multiple runs could be conducted with wood products (rough or finish lumber) in the first, and the four firewood choices in the second run to maximize the overall value.

4. Select the value you believe can be charged to buyers in units of expected sale. One standard cord equals 128 cubic feet of space stacked with split or branch wood (equivalent to 4' x 4' x 8'). The actual volume of wood in this space is 62.5 cu.ft.

5. The user has two sort options. Click on either of the 2 yellow column headers, "*Species Code*" or "*Value by Species ($)*" to sort these columns, respectively.

6. Click on the Print box to print results.

7. Note, that Step 13 can be run even if Step 14 has been completed, and visa versa.



Rough Equivalents

$379	/standard cord
$458	/ton dry weight
$229	/ton green weight
$6.07	/ton green weight

Note: There are no rough equivalents to lumber products.

Step 14. Advanced Valuation - Assign Value Method (tables

A. Create List of Current Species

B. Check Method 2 box. Assign Value to Tree Species by Species Group, or, by Tree

☑ **Method 2 -- Assign Value to Species Groups**

☑ 2a. Assign Value by Species Group (click to select and deselect)

OR

☐ 2b. Assign Value by Tree Size (click to select and deselect)

Assign values to Species Groups in the Yellow cells below.

Preliminary Estimate of Tree Removal Value for Urban Forestry Division, Tree Town, Calif.

Totals =	56,154	$340,683

Species Group	Species used for Species Group	Value ($/cf)	Volume (cf)	Total Value ($)
1	Blue Gum	$6.00	607	$3,645
2	Acacia		0	$0
3	Monterey Pine	$5.00	2	$11
4	Monterey Cypress	$6.00	1	$4
5	Carob	$5.00	4	$18
6	Camphor		0	$0
7	Chinese Elm		0	$0
8	Holly Oak	$6.00	1	$6
9	Jacaranda	$4.00	2	$7
10	Liquid Ambar	$4.00	1	$6
11	Modesto Ash	$6.00	1	$3
12	Sawleaf Zelkova	$4.00	0	$2
13	Chinese Pistache	$5.50	14,900	$81,947
14	Southern Magnolia	$6.50	34,575	$224,735
15	London Plane	$5.00	6,060	$30,300

Figure 43. Assign Value Method (by species group).

Sheet 8 VALUATION

Step 14. Preliminary Valuation - Assign Value Method to Species Groups or Species Groups by Tree Size

The "*Assign Value Method to Species Groups or Species Groups by Tree Size*" is used for situations when the urban forester wishes to assign different values to the Species Groups in one computer run. It is not necessary to calculate the stand and stock tables prior to calculating the value in this step, however, doing so provides valuable management information to the user and allows a more informed set of parameters on the OUTPUT PARAMETERS and REMOVAL sheets. The volume and value data that are obtained from this section are based on restrictions imposed in Steps 7-9a and Choices 1, 2 or 3 checked on the REMOVAL sheet under Method 1A, 1B, or Method 2.

1. Create or Update the species list by clicking on the action box titled "*Click here to Create Current Species List, and Species Groups*," if not already done in Step 13.

2. Check the action box for Method 2.

3. Check either action box 2a or 2b. You cannot have both boxes selected. You must deselect the active box before selecting the other choice.

4. <u>Box 2a — Assign Value by Species Group</u>

a. Selecting this action box generates new columns, and sums the volume per species group.

b. Next, you must enter the estimated value in units of dollars/cubic foot in the "*Value ($/cf)" column* (yellow cells). Use the Wood Valuation chart (Figure 42) to assist your selection. For example, if asking for bids on rough lumber you would enter a lower dollar value for the Monterey pine species group than for the redwood species group.

In the example shown in Figure 43, several different dollar values per species group are shown. Also, a Rough Equivalents list is provided for a relative picture of the value by different units.

c. Results can be sorted by species group, volume, and total value by clicking on the appropriate yellow column headers.

d. Click on the Print box to print results.

5. Box 2b — Assign Value by Tree Size

a. Check the action box for Method 2, and uncheck Box 2a, if not already done. Then check Box 2b.

b Selecting this action box generates new columns and sums the volume into three dbh size classes.

c. Next you must type the estimated value in units of dollars/cubic foot in the three yellow "*Value by Size*" columns. If diameter restrictions were placed in Step 7 (OUTPUT PARAMETERS sheet), or if certain species do not exist among one or more of the small, medium, or large diameter classes, no volume or value will show in the white columns on the right side of the chart.

Use the Wood Valuation chart (Figure 42) to guide your selection. Typically, larger diameters are worth more than smaller sizes. See Figure 44 for an example.

d. Value may be sorted by clicking the "*Total Value ($)*" column header.

e. Click on the Print box to print results.

6. Note, that Step 13 can be run even if Step 14 has been completed, and visa versa.

Figure 44. Advanced Valuation - Assign Value Method (by species group).

Step 14. Advanced Valuation - Assign Value Method (tables that show the value of trees to be removed)

A. Create List of Current Species

B. Check Method 2 box. Assign Value to Tree Species by Species Group, or, by Tree Size, only in cubic feet.

☑ **Method 2 -- Assign Value to Species Groups**

☐ 2a. Assign Value by Species Group (click to select and deselect)

OR

☑ 2b. Assign Value by Tree Size (click to select and deselect)

Assign values to Species Groups in the Yellow cells below.

Rough Equivalents	
$362	/standard cord
$438	/ton dry weight
$219	/ton green weight
$5.80	/ton green weight

Note: There are no rough equivalents to lumber products.

Preliminary Estimate of Tree Removal Value for
Urban Forestry Division, Tree Town, Calif.

Total Volume = 56,154

		Totals =	143	9,734	46,277	$563	$48,671	$276,353	$325,587
		Value by Size ($/cubic foot)			Volume by Size (cf)			Value by Size ($)			Total
Species Group	Species used for Species Group	Small <12"	Medium 12-24"	Large >24"	Small <12"	Medium 12-24"	Large >24"	Small <12"	Medium 12-24"	Large >24"	Value ($)
1	Blue Gum		$5.00		0	607			$3,036		$3,036
2	Acacia				0						$0
3	Monterey Pine	$4.00			2			$9			$9
4	Monterey Cypress	$4.00			1			$3			$3
5	Carob	$4.00			4			$14			$14
6	Camphor				0						$0
7	Chinese Elm				0						$0
8	Holly Oak	$5.00			1			$5			$5
9	Jacaranda	$3.00			2			$5			$5
10	Liquid Ambar	$3.00			1			$4			$4
11	Modesto Ash	$4.00			1			$2			$2
12	Sawleaf Zelkova				0						$0
13	Chinese Pistache	$3.00		$6.00	1		14,899	$2		$89,393	$89,395
14	Southern Magnolia	$4.00	$5.00	$6.00	130	5,681	28,764	$519	$28,405	$172,583	$201,507
15	London Plane		$5.00	$5.50	0	3,446	2,614		$17,229	$14,377	$31,606

PROGRAM NOTES

It should be noted again that using either Method 1A or 1B is a projection, and as such the trees "marked" for removal are not removed from the database.

Also, this is true if Method 2 is used even if the trees are actually "marked" on the ground. If those trees are actually cut, the user needs to go into the CUFIM-PROs database and actually delete the tree from the inventory list using the Delete Record function. This is not automatically done. To accomplish this:

a) Export a backup, print the database, and,

b) Use the Delete Record function (Step 5) to delete the cut trees. If fewer than 50-100 trees are being removed, the Delete Record function is easy enough, however, if several hundred trees are removed, following the Export, Edit, and Import sequence may be more time effective.

Either operation will remove the trees from the computer list.

A User's Guide to CUFIM-PRO, the…

Community and Urban Forest Inventory and Management Program

The Big Freeze

A Case Study for a Coastal California Community Using the 100% Cut Pathway

Part IV. The Big Freeze

A Case Study for a Coastal California Community Using the 100% Cut Pathway

Several "planning scenarios" have been presented during the Step-by-Step discussion of CUFIM-PRO. This example is the more specific case where it is known from the start that 100% of a portion of the trees need removal resulting from disease or insect epidemic or damage from weather or fire.

In this scenario, an unusually long and cold freeze hit Treetown in March. In the spring your field crew reported that the freeze had a major impact on all species of Eucalpytus trees and nearly every tree under about 28" dbh had died and did not resprout, and that the dead trees were unsightly resulting in multiple complaints from the community.

A. Use CUFIM-PRO to Learn About the Forest

Here are some of the ways the user can learn about the makeup and composition of eucalyptus.

1. First launch CUFIM-PRO and review the various species of eucalyptus in your community on the STATS sheet (Figure 46). Click action box D (*Count by Species with Volume*) and then click the Species List header to sort by name and scroll down to the eucalyptus group. They are coded Species Group 1 which means that from a volume standpoint, they are most similar to *Eucalyptus globulus*, blue gum.

You also note the relatively heavy volume, over 400,00 cf, compared to the entire urban forest of 840,791 cf shown in green at the top (in Figure 45). Roughly half of the forest volume is eucalyptus and the majority is in two species, *Eucalyptus globulus* and *Eucalyptus robusta.*

Clicking on the Volume by Species and Count by Species headers shows that *Eucalyptus globulus* has the largest number of trees and volume compared to any other tree in the forest.

This list is a "flat file," colored orange and was created from data on the SETUP sheet and thus has no links or embedded formulas. You may copy information from the file being careful not to alter data. If an inadvertent change is made, click on action box D to re-import the flat file.

2. Select and Copy the Species Codes for all eucalyptus on the list, and click on the "Go to OUTPUT PARAMETERS" jump box.

B. Number of Trees per DBH CLass (click to update)

C. Volume by DBH Class (click to update)

No. of Dbh Classes =	14	Total Volume of Trees =	840,791 cu ft
Total Number of Trees =	21,343		

Dbh	Lower	Upper	No. of Trees in Dbh Class	Tree Volume in Dbh Class
6	2.00	8.99	9,748	30,601
12	9.00	14.99	2,796	64,976
18	15.00	20.99	5,535	235,450
24	21.00	26.99	1,983	164,571
30	27.00	32.99	666	100,413
36	33.00	38.99	363	87,758
42	39.00	44.99	155	59,732
48	45.00	50.99	61	47,279
54	51.00	56.99	17	20,119
60	57.00	62.99	7	8,502
66	63.00	68.99	5	6,836
72	69.00	74.99	2	3,369
78	75.00	80.99	3	5,372
84	81.00	86.99		
90	87.00	92.99	2	5,815
96				

Figure 45. STATS sheet statistics for ALL trees in the urban forest.

D. Count by Species with Volume (cf) (click to update)

Click heading to sort list

Species Code	Species List	Species Group	Count by Species	Volume (cf) by Species
1	Acacia baileyana	5	64	288
3	Acacia decurrens	6	259	6568
4	Acacia longifolia	14	265	27724
324	Escallonia X exonien	9	3	4
101	Eucalyptus camaldule	1	22	218
102	Eucalyptus cinerea	1	11	253
103	Eucalyptus citriodor	1	79	1673
106	Eucalyptus ficifolia	1	115	650
107	Eucalyptus globulus	1	3694	354664
110	Eucalyptus lehmannii	1	3	14
111	Eucalyptus leucoxylo	1	1	15
113	Eucalyptus maculata	1	31	473
114	Eucalyptus nicholii	1	55	310
117	Eucalyptus polyanthe	1	130	1814
331	Eucalyptus pulverule	7	3	50
118	Eucalyptus robusta	1	1605	51482
119	Eucalyptus rudis	1	39	572
120	Eucalyptus sideroxyl	1	90	1038
123	Eucalyptus torquata	1	13	59
124	Eucalyptus viminalis	1	65	1329
367	Fagus sylvatica 'Atr	15	2	117

Figure 46. STATS sheet showing eucalyptus group.

3. Locate the Inclusion List, Step 9a, and paste the eucalpytus Species Code numbers in the yellow column, Figure 47.

Step 9a. Indicate Only the Species to be INCLUDED in Volume Calculation

A. A maximum of 50 species can be included
B. No errors detected

Click here to CLEAR all species in this section

Include Species Code	Species Name	Species Group
101	Eucalyptus camaldulensis	1
102	Eucalyptus cinerea	1
103	Eucalyptus citriodor	1
106	Eucalyptus ficifolia	1
107	Eucalyptus globulus	1
110	Eucalyptus lehmannii	1
111	Eucalyptus leucoxylo	1
113	Eucalyptus maculata	1
114	Eucalyptus nicholii	1
117	Eucalyptus polyanthe	1
331	Eucalyptus pulverule	7
118	Eucalyptus robusta	1
119	Eucalyptus rudis	1
120	Eucalyptus sideroxyl	1
123	Eucalyptus torquata	1
124	Eucalyptus viminalis	1

Figure 47. Adding eucalyptus species code numbers to the Included list box.

B. Calculating Tree Numbers and Volume

Some of the following steps resemble the process discussed in Part II of the report, however, the flow-path is different as are many details.

4. Set the minimum and maximum tree diameters in Step 7. Your crew reported that all eucalyptus under "about 28 inches" were dead and not resprouting, therefore, set minimum to "None" and maximum to "28." Figure 48a.

5. Stand and Stock Tables. In Step 9b, "*Management Options for 100% Species Removal by Species and Diameter*," you can choose to create a stand table, a stock table or both (Figure 48b). By clicking on action box A (*Click to Generate the Stand Table for Included Species*) and B (*Click to Generate the Stock Table for Included Species*) the two tables on their right are populated. See Figure 49. These are similar to the stand and stock tables provided in Step 11a and 12a, except those were based on Species Groups and these are based on Species Codes which provides greater detail.

Step 7. Range of Tree Diameters to be Included in Volume Calculation

If left blank, all tree diameters will be included.

Reset Min & Max

Minimum tree diameter to be included (inches) = None — Select "None" to include smallest dbh.
Maximum tree diameter to be included (inches) = 28 — Select "None" to include largest dbh.

RESULT: ONLY trees from **Smallest** through **28.00"** dbh will be included in volume calculations.

Figure 48a. Setting minimum and maximum diameters for The Big Freeze scenario.

Reviewing the tables in Figure 49 shows that 4,885 trees between 2-28" in diameter would be removed for the 16 eucalyptus species with a combined volume of 216,190 cu. ft. To determine the percent of total eucalyptus trees in the forest, conduct a second "run" of the stand and stock tables with the minimum and maximum diameters both set to "None" in Step 7. Doing this would allow you to quickly calculate that 82% of the eucalyptus are slated for removal amounting to 52% of the total eucalyptus volume in the forest.

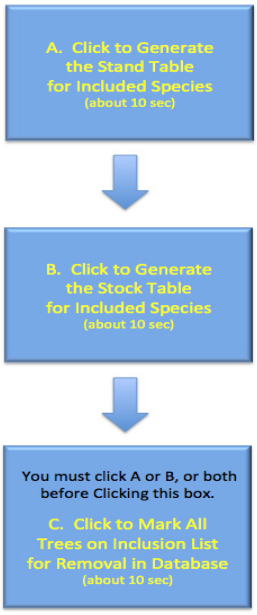


Figure 48b. 100% Cut flow chart.

On the side, the stand and stock table information could be grouped by listing the number by dbh class in three general groups: small trees (2-10") = 863, medium trees (>10-20") = 1,956, and large (>20-28") = 2066. This data along with same for volume would provide a concise statement for those only needing a summary.

Note that the order of the Species Codes in the Stand Table differs from that in the Stock Table as they are sorted in descending order, left to right, and depending on tree size the highest volume doesn't necessarily come from the greatest number.

Also, when comparing Figure 49 with Figure 46, the number of trees will differ as Figure 46 is for all eucalyptus while Figure 49 is for trees in the range of 2-28" diameter.

The last part of Step 9b is to mark the trees to be cut in the database by clicking on action box C (*Click to Mark All Trees on Inclusion List for Removal in Database*). Marking the trees by box C MUST be done at this point, as the assessment of value made on the VALUATION sheet requires that information. However, you can always return to this section and revise your output parameters for another "run," if needed.

Step 9b. Management Options for 100% Species Removal by Species and Diameter

Click here to PRINT Stand Table (2 pages)

Click here to PRINT Stock Table (2 pages)

Click here to CLEAR Stand and Stock Tables and Removal Codes from DATABASE

1. Step 9a is designed to R
2. Trees with diameter rest
3. Generate (A) Stand and

Number of Trees on Inclusion List (by Diameter and Species Code)

Total Number	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum
4,885	2623	1605	130	115	90	79	65	55	39	31	22	13	11	3	3	1

Total	Diameter (inches)	Species Code 107	118	117	106	120	103	124	114	119	113	Species Code 101	123	102	110	331	111
164	2	86		16	14	20	5	3	11	1	1	7					
133	4	25		13	35	9	7	5	19	4	5	5	4	1	1		
122	6	122															
444	8	207	1	48	42	30	26	22	17	17	14	5	8	3	2	2	
	10																
	12																
253	14	68		42	20	26	32	24	7	16	9	3	1	3		1	1
1	16	1															
1,604	18		1604														
98	20	49		11	4	5	9	11	1	1	2	1		4			
97	22	97															
1,802	24	1802															
84	26	84															
83	28	82										1					
	30																

Volume of Trees on Inclusion List (by Diameter and Species Code)

Total cf	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum	Sum
216,190	156,240	51,482	1,814	1,673	1,329	1,038	650	572	473	310	253	218	59	50	15	14	

Total	Diameter (inches)	Species Code 107	118	117	103	124	120	106	119	113	114	102	101	123	331	111	Species Code 110
47	2	36		2	1	1	2	2	0	0	1		2				
164	4	34		17	10	7	11	29	6	10	25	1	7	5			1
218	6	218															
2,328	8	897	5	309	188	147	184	188	99	100	94	15	35	39	16		12
	10																
	12																
6,149	14	1,835		944	919	571	593	294	405	240	142	52	90	15	34	15	
21	16	21															
51,477	18		51,477														
5,659	20	3,094		543	555	603	248	138	62	123	47	185	62				
5,494	22	5,494															
128,436	24	128,436															
7,293	26	7,293															
8,905	28	8,882											23				

Figure 49. Stand and stock tables for all eucalyptus species from 2 to 28" in diameter.

You may wish to review the DATABASE sheet to check that the correct trees were marked for removal. You will also see the database is now sorted with removal trees at the top.

C. Assessing Value

It is important to bypass the REMOVAL and MGT OPTIONS Sheets as that information has already been determined, and select the VALUATION sheet (see discussion and graphic on the first page of Part III). Note: Do NOT click on the Random Number Generator on the REMOVAL sheet as it will delete all trees marked for removal and you'll need to start the process again.

As discussed earlier there are two Valuation steps, each with choices. Step 14, *Advanced Valuation - Assign Value Method to Species Groups or Species Groups by Tree Size*, could be used, however, it aggregates the data into Species Groups rather than keeping it separated by individual species.

It would be more advantageous in this case to work with Step 13, which uses the Species list.

<u>Step 13, Same Value Method and Assign Value by Species:</u>

First create the current species list (Item 1 on the spreadsheet, Figure 50), then click on the Method 1

checkbox (Item 2). In the Method 1 box there are two drop down menus to select from plus one optional item.

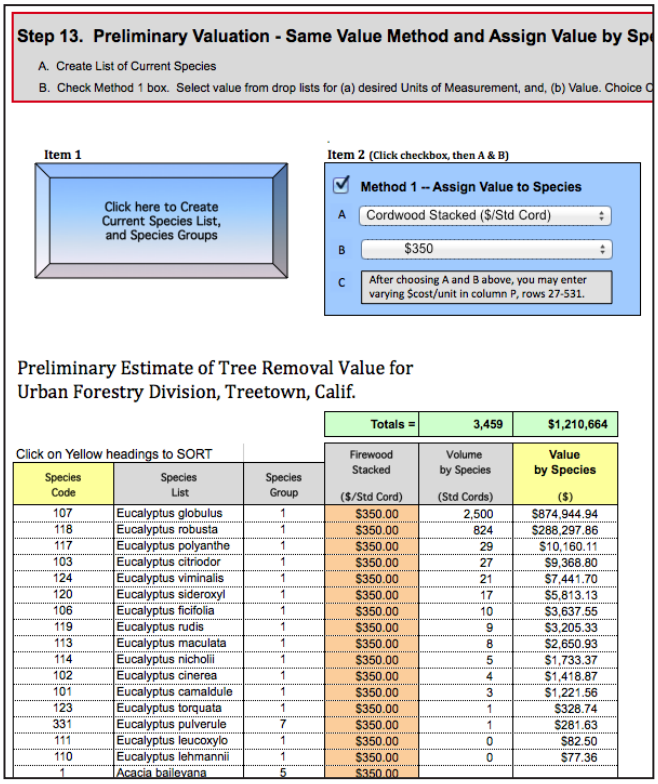
Item 2, Part A: Here the user is asked to consider the likely unit of measurement for the wood from the six choices. The only logical options for this case study would be cordwood or green weight. We'll consider both choices.

Item 2, Part B: Determine the market value for each species and make an assessment of the gross value. From the gross we will deduct expenses and profit for woodcutters in a following step. Having kept up on the selling price of eucalyptus, $350/cord is selected.

Click on the heading "*Value by Species*" to bring the eucalyptus information to the top rows in sorted order.

Item 2, Part C: The initial listing assigns the same value to all species marked for removal. In many situations this may be adequate. However, the process allows for changing the value for any or all of the species in question. The cells in column P27:P531 are colored orange, which means you could alter the cell data, CAREFULLY. Do not mouse click in other columns as it may disrupt formulas causing errors in the program.

With this option, more control over the dollar estimate is possible. For example, if your experience shows that *E. globulus* should only be valued at $275/cord but *E. robusta, citriodora and viminals* should all be valued at $400/cord, then type in those edits (Figure 51). The new total is now $1,066,763 and may better represent market conditions.



Preliminary Estimate of Tree Removal Value for Urban Forestry Division, Treetown, Calif.

			Totals =	3,459	$1,210,664
Click on Yellow headings to SORT			Firewood Stacked	Volume by Species	Value by Species
Species Code	Species List	Species Group	($/Std Cord)	(Std Cords)	($)
107	Eucalyptus globulus	1	$350.00	2,500	$874,944.94
118	Eucalyptus robusta	1	$350.00	824	$288,297.86
117	Eucalyptus polyanthe	1	$350.00	29	$10,160.11
103	Eucalyptus citriodor	1	$350.00	27	$9,368.80
124	Eucalyptus viminalis	1	$350.00	21	$7,441.70
120	Eucalyptus sideroxyl	1	$350.00	17	$5,813.13
106	Eucalyptus ficifolia	1	$350.00	10	$3,637.55
119	Eucalyptus rudis	1	$350.00	9	$3,205.33
113	Eucalyptus maculata	1	$350.00	8	$2,650.93
114	Eucalyptus nicholii	1	$350.00	5	$1,733.37
102	Eucalyptus cinerea	1	$350.00	4	$1,418.87
101	Eucalyptus camaldule	1	$350.00	3	$1,221.56
123	Eucalyptus torquata	1	$350.00	1	$328.74
331	Eucalyptus pulverule	7	$350.00	1	$281.63
111	Eucalyptus leucoxylo	1	$350.00	0	$82.50
110	Eucalyptus lehmannii	1	$350.00	0	$77.36
1	Acacia baileyana	5	$350.00		

Figure 50. Gross market value, before expenses, for eucalyptus trees slated for removal.

Preliminary Estimate of Tree Removal Value for Urban Forestry Division, Treetown, Calif.

			Totals =	3,459	$1,066,763
Click on Yellow headings to SORT			Firewood Stacked	Volume by Species	Value by Species
Species Code	Species List	Species Group	($/Std Cord)	(Std Cords)	($)
107	Eucalyptus globulus	1	$275.00	2,500	$687,456.74
118	Eucalyptus robusta	1	$400.00	824	$329,483.27
117	Eucalyptus polyanthe	1	$350.00	29	$10,160.11
103	Eucalyptus citriodor	1	$400.00	27	$10,707.20
124	Eucalyptus viminalis	1	$400.00	21	$8,504.80
120	Eucalyptus sideroxyl	1	$350.00	17	$5,813.13
106	Eucalyptus ficifolia	1	$350.00	10	$3,637.55
119	Eucalyptus rudis	1	$350.00	9	$3,205.33
113	Eucalyptus maculata	1	$350.00	8	$2,650.93
114	Eucalyptus nicholii	1	$350.00	5	$1,733.37
102	Eucalyptus cinerea	1	$350.00	4	$1,418.87
101	Eucalyptus camaldule	1	$350.00	3	$1,221.56
123	Eucalyptus torquata	1	$350.00	1	$328.74
331	Eucalyptus pulverule	7	$350.00	1	$281.63
111	Eucalyptus leucoxylo	1	$350.00	0	$82.50
110	Eucalyptus lehmannii	1	$350.00	0	$77.36
1	Acacia baileyana	5	$350.00		

Figure 51. Gross market value, before expenses for eucalyptus trees slated for removal.

D. Interpreting the Data

In round numbers, the street value of these trees is approximately 1 million dollars. However, wood cutters working with stacked cordwood incur large expenses. Depending on the operation, trucks, chain saws, loaders, insurance, and log splitters not to mention the high cost of labor for all aspects of the operation plus stacking and wood handling are all major expenses.

A more cost effective method could be selling the wood by green weight (if sold within 3-6 months of tree death) or dry weight (if sold 6 months or longer after tree death). In Figure 42, the Wood Valuation chart, shows the approximate equivalent values for various firewood measurement units. The green weight value would be about $210/ton.

Having knowledge about local wood cutting operations would provide direction on how to set up a contract for the wood and especially to set a fair price to charge wood cutters.

Note: It should be obvious that the street address or location must be a part of the database to assist in the removal process.

Literature Cited

Pillsbury, Norman H. and J.A. Stephens. 1978. *Hardwood Volume and Weight Tables for California's Central Coast*. California Department of Forestry, Sacramento. 54 p.

Thompson, R., Pillsbury, N. and J. Hanna. 1994. *The Elements of Sustainability in Urban Forestry.* Urban Forest Ecosystems Institute. Technical Report 1. Cal Poly, Natural Resources Management Department and the California Department of Forestry and Fire Protection, Riverside. 56 p

Pillsbury, N. H., J.L. Reimer and R.P. Thompson. 1998. *Tree Volume Equations for Fifteen Urban Species in California*. Technical Report No. 7, Urban Forest Ecosystems Institute, Natural Resources Management Department, California Polytechnic State University, San Luis Obispo. 56 p. plus 942 pages in Appendix.

Appendix

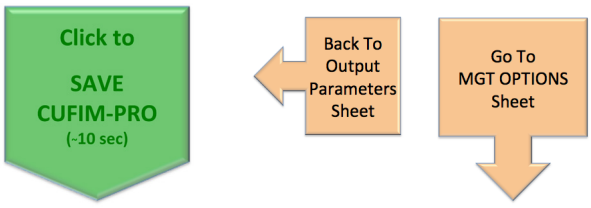
Appendix A.
Notes on Setup and Use

Notes for CUFIM-PRO setup:

1. Close all other applications.
2. Download the CUFIM-PRO User's Guide and the *CUFIM-PRO folder* to your desktop. Use web site (http://ufei.calpoly.edu/urbanwood/links.lasso; look for CUFIM-PRO (MS Excel Program) at http://www.ufei.org/files/ufeipubs/CUFIM_Report.pdf).
3. Launch CUFIM-PRO with Microsoft® Excel®.
4. Make sure that Preferences/Calculation is set to automatic.
5. Arrange desk top for maximum screen viewing space. Set sheet zoom level to 75 - 100% for best readability.
6. Adjust the computers sound level to about one-third volume so when the program outputs alert "beeps" they are sufficient but not overly loud.

Notes for Using CUFIM-PRO:

1. Only enter data in yellow colored cells
2. Click on "Go to…" labels to move from sheet to sheet, or use tabs at the bottom.
3. Click on any action box that starts with "Click here to…" (see examples, below)
4. Click on any action box that says "Print…" in it
5. Click on List action boxes, Combo Boxes and Check Boxes
6. Click on colored 3-D action boxes or rectangles
7. Do not click on cells that aren't on this list; program is not protected.



Appendix B. Troubleshooting

Because CUFIM-PRO is not a protected program, inadvertent errors could occur. This section is offered to advise the user about potential problems and solutions.

A. Microsoft® Excel® Issues. Microsoft Excel occasionally has hiccups. I have observed the following while developing the program.

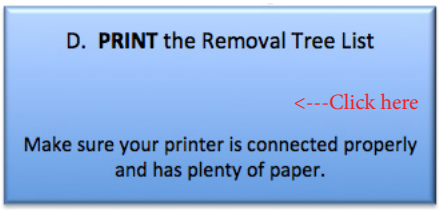
1. Problem: All sheets of Excel freeze even though non-Excel work continues unaffected.

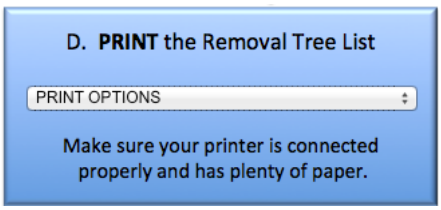
   Solution: Sometimes it'll clear on it's own in a minute but mostly I had to force quit and restart. That always cleared the problem. Also, most of the time during a freeze I was able to Save my work. Give it 10 seconds to save before force quitting.

2. Problem: Ghost images or duplicate icons (like action boxes) appear.

   Solution: Scrolling up or down past the problem area, or clicking another sheet, then back usually refreshes the screen.

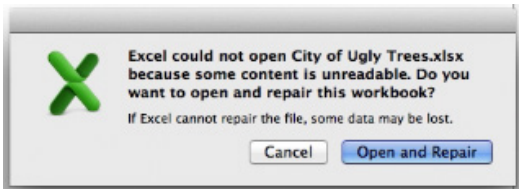
3. Problem: Drop boxes or drop menus occasionally disappear.

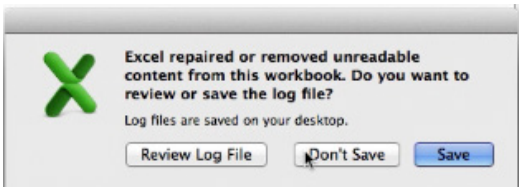




   Solution: This can be disconcerting but the fix is easy. Click on the blank area where the drop box should be located and it will reappear and stay after the cursor is removed.

4. Excel can't read a file. Occasionally Excel says it "can't" open a file, yet it may open correctly the next time without error. However, this double message shows if this condition occurs.

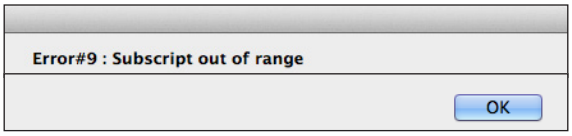




   Solution: The only viable solution is to click on "Open and Repair," and then on "Don't Save." That will allow the program sequence to continue smoothly and no errors occur. Excel forums shed no light on this intermittant problem.

B. User Issues.

NOTE: All the major routines in CUFIM-PRO include Excel's VBA *On Error* coding. If *On Error* detects a run-time error, it will immediately provide an error message and after clicking OK, it will return to the stable state and position prior to the operation you initiated (example below). Thus experiencing a run-time error is most unlikely, however, should the unexpected occur, item 1 below walks you through the steps to recovery.



1. Problem: The program aborts and takes you to the source code .

   Solution: Don't panic, most issues are easily resolved, but how you first respond is important.

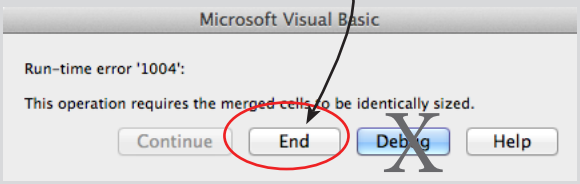
   a. First -- DO NOT click on DEBUG. Instead click on END. Clicking on Debug will take you into the programming portion of the source code and altering any code will likely cause the program to fail.

   b. Follow the steps in Figure 56, see previous page. In addition you may need to Reset the Module. Do this while a source

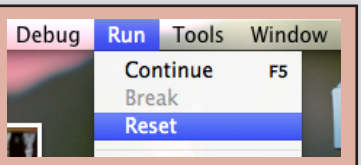
Figure 56. Flowchart for Resetting a Visual Basic Error

Step 1. Reset Calculation

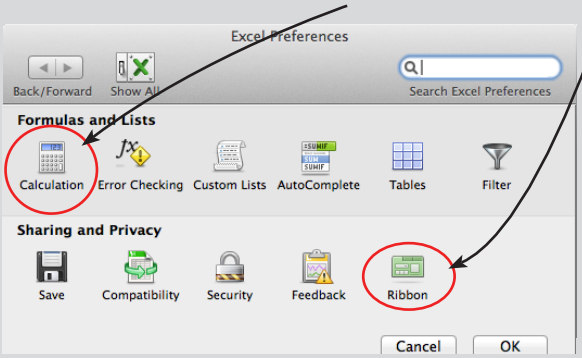
a) If the source code has been interrupted, an error box like this will appear. Do NOT click on DEBUG, instead, click END.



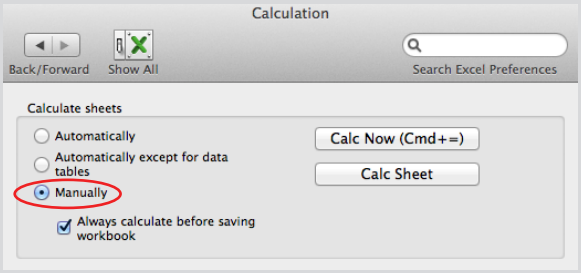
Warning: If you click on Debug, you will have to select the menu Run/Reset, first.



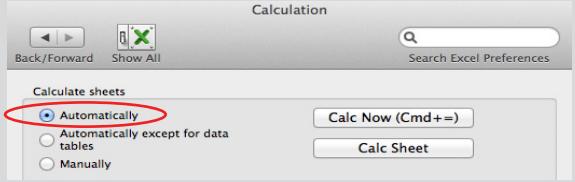
b) Next, go to the main Excel File menu and select Preferences and then "Calculation."



c) In the Calculation window you will see that calculation is set to "Manually."

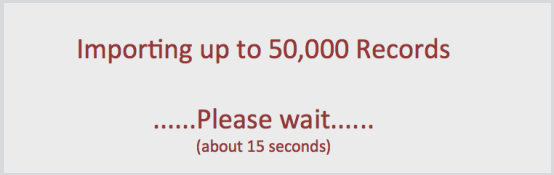


d) Select "Automatically" and close the window.

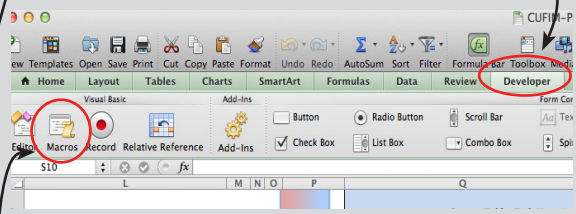


Step 2. Reset CUFIM-PRO

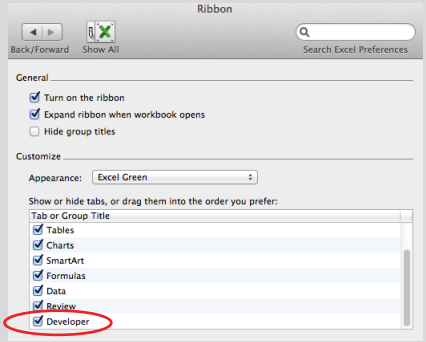
a) Return to the ID sheet where you'll see the large "Importing... Please wait" message.



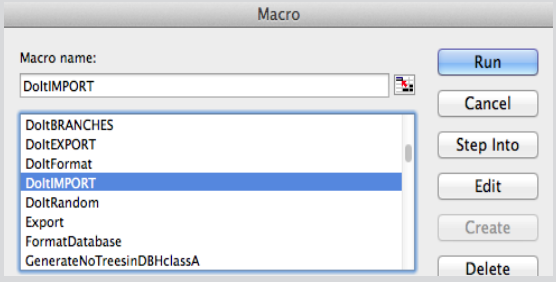
b) If the "Developer" tab doesn't show on the ribbon, on a Mac select Excel/Preferences/ ribbon.



c) Check the "Developer" box, and click OK.



d) Return to CUFIM-PRO's ID sheet, select Developer on the ribbon, then click the Macros icon. Scroll down the Macros list and select DoItIMPORT, then press RUN. Doing this allows the Macro to finish and the "Please wait" sign is removed. Now the program has been reset.



code page is activated and click on the menu "*Run/Reset*."

c. If the source code modules are still visible, select one of them and then select, menu "*Excel/Close and Return to Microsoft Excel*."

d) After following the Reset instructions (Figure 56) and items above, you will have returned the system to the normal, usable state before the error. Try to remember what was done just prior to the error and avoid that. Check to make certain the file you tried to open was the correct, fully populated database.

2. Problem with yellow cells. The most important rule to remember is to never click, tab or return into a non-yellow cell. Yellow cells are the safe zone and many non-yellow cells hold hidden text or formulas required for CUFIM-PRO to properly run.

   Solution: If you realize you've clicked into a non-yellow cell and possibly altered hidden information, go to menu Edit/Undo.

   If that doesn't correct the problem, a) immediately Export the database (ID sheet), b) close CUFIM-PRO <u>without</u> saving it, and c) reopening CUFIM-PRO will eliminate the error.

   If you saved the file before realizing the problem, locate a copy of the original or download another copy from the web and place it in the proper folder (see Figure 4). Now, import the latest database file and this will preserve all work and correct inadvertent changes to CUFIM-PRO.

3. Problem with blank cells (missing data) in the database. When entering data, either from the *Enter New Record* box (Step 5, DATABASE sheet), or when setting up the *Import Template.xlsx* for inputting a large database, all cells in these columns for each tree record MUST have a value:

Col No.	Variable Name
1	Sequence Number
2	Tree Record Number
3	Species Code
5	Species Name
6	Species Group
7	Tree Dbh

   Tree height is not included as the user can't Sort by height. Since height may or may not be included in the measurement scheme, there may be blank cells in this column. Blank cells won't affect volume calculations as long as dbh is included.

   The User Defined Variables, columns 17-26 may also fall into this requirement depending on the use of the data. Any variable that will be sorted, must also have all cells filled.

   This is required because if a column containing blank cells is sorted those tree records are forced to the bottom of the entire database list, an obvious error.

   Solution: Initially you may not be aware that this error has occurred. To check for blanks, split the DATABASE screen horizontally so you can view the first and last row of data. Click on each sort button available and check the first and last cells. If any blanks occur in the sorted column, those tree records are in the wrong position. Make a note of all tree record and Sequence numbers with blank cells and which attributes were missing.

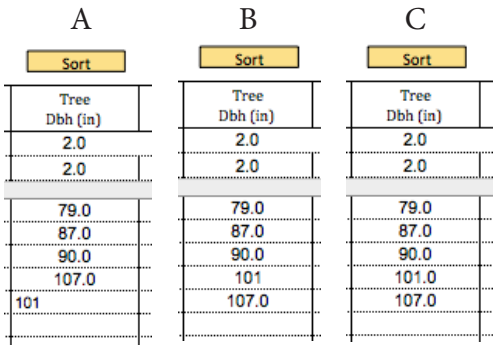
   To correct the error, first determine what value should have been inputted into each blank cell. Next, use the Change Record box on the DATABASE sheet to input the correct value. Then to check that each value was entered properly, scroll into the database to view those trees. Also, Sort each column again to make sure all blanks were filled.

Other errors that may cause incorrect answers or even cause the program to abort are:

4. The database may have an entry that is incompatible with the other values in the column, such as a text entry in a numeric column. This won't happen when using the Enter New Record function in CUFIM-PRO but potentially could occur when cutting and pasting in the Import Template. Text data will not Sort properly and causes an "unseen" error.

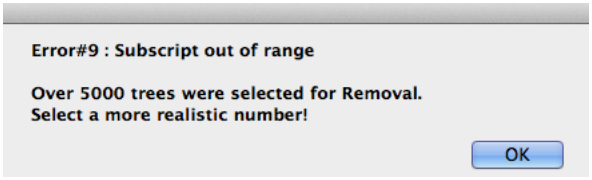
   In the split screen example below, the Dbh value of 101 is actually text in A and B, but a value in C. When Sorted it may be forced

to the bottom (A) or formatted incorrectly (B). The correct location and format is C.



Solution: Look for any formatted values that are left justified or have incorrect decimal places compared to other values. Test the rows, one-by-one to isolate bad data, and format properly.

5. The REMOVAL sheet, Box B, Part 1 asks the user to input the percent of removal trees. Typing in percentages that add to more than 5000 removal trees most likely will cause the Random Number Generator to stop all calculations and reset, after providing this message box:



Solution: First, don't input percentages that cumulatively add to over 5000 trees. Secondly, if that should occur, follow the steps to Reset CUFIM-PRO (Figure 56, above).

6. Unknown error.

   Solution 1: Quit Excel, and Restart.

   Solution 2: Export the database to create a backup copy. Replace CUFIM-PRO from with an original copy or download a clean version from the web. Re-import the database and repeat the operation causing the error. If the error is repeated, it's not a CUFIM-PRO error, rather, it's a problem in the database. See the sections above for possible causes.

Some Don'ts

1. Don't change the name of the program, CUFIM-PRO or the names of any folders or files unless instructions have directed you to do so.
2. Don't change the name of any of the 10 program's sheets (also called tabs)
3. Don't delete any sheets, such as the *Scratch Workspace* or the sheet labelled *Don't Remove*. They are required for proper operation.
4. Don't move files out of folders. Maintain the folder and file structure shown in Figure 4 at all times.
5. The Species Code and Species Group Code must be a valid number. The Sorting options will produce errors if they are not.

Some Notes

1. CUFIM-PRO was developed on the new MacPro (late 2013) with Microsoft Excel version 14.4.6 (141106). This version of Excel is purported to be fully compatible with the Windows version of Excel on a PC. However, the reverse is not true. Therefore, if running on a PC, do NOT make changes to the program if there is a chance it will be also be running on a Mac as it may not be compatible. Further, source code changes should only be made by someone strong in the Visual Basic Application language and who has studied CUFIM-PRO's source code and fully understands how the many parts work together.
2. It may be tempting to make changes directly in the database on the DATABASE sheet in non-yellow cells. Don't do it. Use the Enter New Record, Change Record or Delete Record functions provided. They have built in checks to help eliminate entry or change errors. If many changes are needed, such as from a re-inventory, follow the Export/Edit/Import sequence discussed in Part II. Following that, run the *Import Checker.xlsx* to search for errors.
3. If you have an archived database file and wish to re-import it, the file must first be placed back in the *CUFIM-PRO* folder in the *CUFIM-PRO Database BACKUP Files* folder.

Appendix C. How to Import an Existing Database into CUFIM-PRO

This section discusses how to import an existing database into CUFIM-PRO. If you are a regular user of Excel, you will be able to follow these steps. If not, you may have to find an advanced user to assist you. There are many tricky details that can cause problems, but hopefully these tips and suggestions will ease the process.

Some of the operations discussed below can take several minutes to complete depending on your computer speed and size of database. If your database is over 40-45,000 records, you should break it into smaller groups.

It is recommended that before trying to import the data that you read the entire Users Guide to understand how the program works. This knowledge will assist you with the entire process. In the guide, it specifically instructs the user to type only in yellow colored cells. Many operations discussed here will instruct you to work in non-yellow cells, however, after completion, you must avoid doing so.

A. Exporting the Old Database

The process described here will reformat the "look" of the old database. You may have to reformat borders and cell contents as well as adjust row and column spacing as you proceed. Saving your work frequently and backing up your dataset is recommended.

1. There are a number of different tree inventory programs available. Most have a function that allows the user to export the data. Export in a format that Excel can open, such as tab or csv delimited.

2. Use the File/Open command to import the file (exported in step 1) into a blank Excel spreadsheet. Immediately *Save As*... as an .xlsx or .xls file. For purposes here we'll call this file *OrigDB.xlsx*.

3. Open *OrigDB.xlsx* and make sure the data looks the way you expected. This is an opportunity to examine your data carefully and make sure that the set is complete. As a minimum make sure numbers and text are not mixed in the same column, except when needed (as in an address). Look for spelling errors and numbers that are obviously incorrect (negative numbers, values that are too large, etc.). Eliminate any records whose contents don't match expected data. The better the data being imported, the better the projections you will ultimately make. This process may even involve field checking to make sure the data are correct.

The discussion below is divided into two parts to achieve this process, *B. Preparing the* OrigDB.xlsx *file and Importing the Species List section into Sheet 2, SETUP*, and, *C. Preparing and Importing into Sheet 5, DATABASE.* Both parts use the *Import Template.xlsx*.

B. Preparing the *OrigDB.xlsx* file and Importing the Species List section into Sheet 2, SETUP.

1. In this part you will assign (1) a species code, if needed, and, (2) a Species Group number to each tree in the *OrigDB.xlsx* file. For Species Groups, see the discussion in Part II of the Users Guide to assist in this process; they are specifically defined for CUFIM-PRO. Code stumps and non-volume trees like palms with a Species Group code of 50.

2. If species codes came with the dataset, make sure the codes are values and not text. If they are text, insert a column to the right of the column of codes and in the first data row, say row 3, type: =VALUE(A3), and fill down to the bottom of the dataset. See Figure 51. Select the data in the new column, copy and Paste Special Values onto itself to eliminate the formula.

3. Arrange the first three columns in this order: species code, species name-normally the scientific name, and species group number (Figure 52a).

Next, sort the entire array of data by the species code column.

4. Insert a new column to the right of the Species Group column and label it "Unique Sp. Code."

	A	B	C	D	E
	Species Code	**Numeric Sp Code**	**Botanical Name**	**Species Group**	**Unique Sp Code**
	1	=VALUE(A2)	Acacia baileyana	5	=B2
	1	=VALUE(A3)	Acacia baileyana	5	=IF(B3=B2,"",B3)
	1	=VALUE(A4)	Acacia baileyana	5	=IF(B4<>MAX($E$2:E3),B4,"")
	3	=VALUE(A5)	Acacia decurrens	6	=IF(B5<>MAX($E$2:E4),B5,"")
RESULT:					
	Species Code	**Numeric Sp Code**	**Botanical Name**	**Species Group**	**Unique Sp Code**
	1	1	Acacia baileyana	5	1
	1	1	Acacia baileyana	5	
	1	1	Acacia baileyana	5	
	3	3	Acacia decurrens	6	3

Figure 51. How to extract unique species code numbers from a database.

Type in a formula that, when filled down, will increment by 1 each time a row with a different species code is encountered. An example of how to do this is shown in Figure 51. Note that a different formula is in each of the first 3 data rows, but the formula in the 3rd row is filled down to the last data row.

You MUST Copy the cells in this column and Paste Special Values to eliminate the formulas. In the example in Figure 51, select columns B-E, copy and paste into a second tab in the *OrigDB.xlsx* spreadsheet.

5. In the second tab, select the 4 columns of data (Species Code, Species Name, Species Group, Unique Sp. Code) excluding headers and blank lines, starting from upper left to lower right (use split screen). Tab three times to highlight the first cell in the 4th column (Unique Sp. Code). Click on the menu Data/Sort, and make sure that the correct column (first value in Unique Sp Code) is designated for sorting, and that "No header row" and Ascending are selected. Then, Sort, and Save.

6. With the screen split horizontally, scroll down until numbers in the Unique Sp Code column end. Select just the data (no headers) for the first three columns, down to and including the last row where the data in the Unique Sp Code column ends. Don't select cells in the Unique Sp Code column.

7. Copy the three columns (Species code-numeric, Species Name, and Species Group), and activate the *Import Template.xlsx* found in the CUFIM-PRO folder (Figure 52a), now named *Import CityName.xlsx*. On its Database tab, click on cell A5, and paste over the green section. You may have to replace borders or format columns to your liking.

8. The maximum number of species allowed is 500; check to be sure that is the case.

C. Preparing and Importing into Sheet 5, DATABASE.

1. On the Database tab of the original exported dataset, *OrigDB.xlsx*, arrange the following columns from left to right: (1) Tree Record Number (you may or may not need to create this), and (2) Species Code (the numeric version). Create consecutive numbers in the Tree Record Number column. Use the Edit/Fill/Series function. Do not sort the data!

2. Copy the data in the two columns listed above (no headers or blank lines), activate *Import YourCityName.xlsx* and paste into cell E5 on the Database tab (the orange Tree Record column).

3. Very Important! Review the species codes in the list (column A) making certain that ALL ARE associated with only one species and that

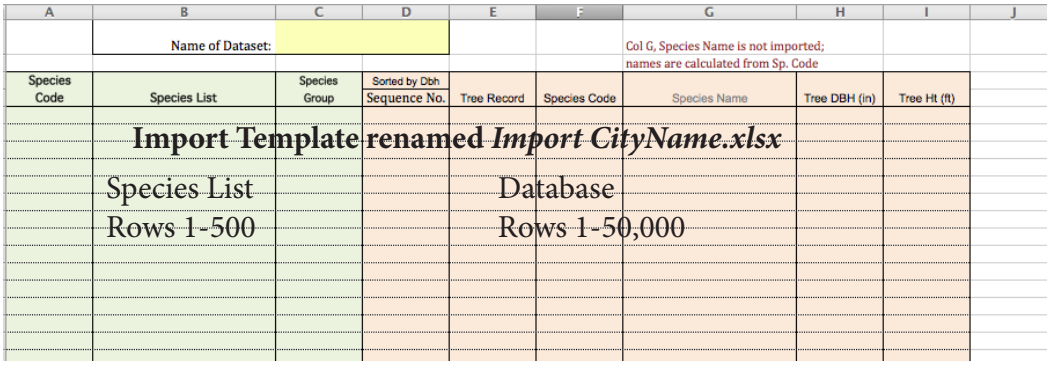


Figure 52a (from Part II). Import Template.xlsx.

the species NAME is spelled correctly. If there are errors, they must be fixed prior to importing the database.

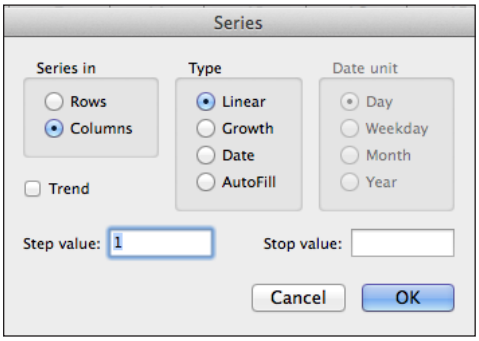
4. In the original exported dataset, *OrigDB.xlsx,* organize tree diameter and tree height (if available) in side-by-side columns in this order. If tree data is metric, convert to English units. If tree diameters or heights are given as a range, e.g., 4-8", use the mid-point. Make sure they are values, not text. Do not sort because when they are pasted into *Import YourCityName.xlsx* they must match the data already placed there.

Copy the diameter and height columns (no headers or blanks), and paste into cell H5 in the *Import YourCityName.xlsx* sheet. Check to be certain that the rows in columns E-I contain the correct information for each tree's record. Do not sort because the User Defined Variables must also properly match.

5. After all data has been entered, notice that the Tree Sequence Number column (column D) is empty. Click on cell D5, and type the number 1. Click on D5, hold shift key and click on the last cell of column D where tree data occurs (last Tree Record Number), then,

a. Edit/Fill/Series

b In the Series window select Series in Columns, Type is Linear, Step Value is 1. Click OK.



6. User Defined Variables

Your original dataset may have several other variables or descriptors that you want to keep, such as:

Tree location (address)
Date measured
Tree condition
Sidewalk damage

Enter the name of each variable in columns/cells K4:T4 of the *Import YourCityName.xlsx* file, in the yellow fields.

Copy and paste variable data from the file *OrigDB.xlsx* to the correct column, and, again, check to be sure the data for each tree is correct.

Follow the same procedures and cautions described earlier. You may have to widen some columns to see the cell content.

The last item to include is the dataset name, in cell C1 of the *Import YourCityName.xlsx* file. Name your dataset like, "CityName Data" where CityName is the name of your city. Be sure the sheet name is: Database.

D. Ready for Importing the Dataset?

Check the new database carefully and fully. Mistakes in transporting and reformatting the data could cause huge errors. Test the database by:

1. Look for missing data and errors in species coding by running *Import Checker.xlsm*. Many errors can be detected and resolved prior to importing with this program (see Part I, C.5).

2. If errors occur, find the source and fix them.

3. Note that the species name is not imported to avoid typos; the species name is generated from the species code from the SETUP sheet in CUFIM-PRO.

When you have thoroughly tested the database, it's ready to import into CUFIM-PRO. Place the *Import YourCityName.xlsx* file in the CUFIM-PRO folder, but not in either of the sub-folders.

The import instructions are in *Part II, Tree Inventory and Database Setup and Maintenance*, Sheet 1, Step 1b (Import and Export Options).

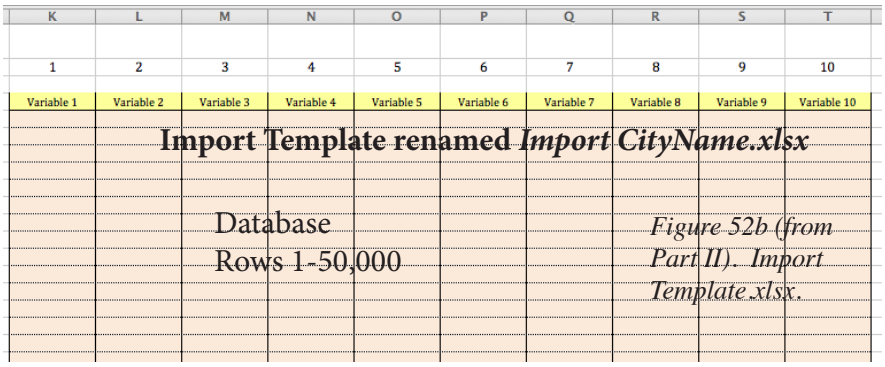


Figure 52b (from Part II). Import Template.xlsx.

Appendix D. How to Split the CUFIM-PRO Database

When the CUFIM-PRO database reaches 40-45,000 trees, it will need to be split into two files.

This operation should only be attempted by someone with excellent Excel skills.

1. First, Sort the database on the DATABASE sheet by Tree Record Number (column 2).

2. Export the database from the ID sheet. It will have a name like: *CUFIM-PRO Database mmm dd, yyyy--Time is hh mm ss.* Quit Excel.

3. Duplicate the *CUFIM-PRO folder*, twice.

4. Rename the new folders with names such as:

CUFIM-PRO A folder, and *CUFIM-PRO B folder*, and both sub-folders, as:

CUFIM-PRO A Program BACKUP Files

CUFIM-PRO A Database BACKUP Files

and in the second set,

CUFIM-PRO B Program BACKUP Files

CUFIM-PRO B Database BACKUP Files

and both program files, one in each set, such as:

CUFIM-PRO A, and *CUFIM-PRO B*.

By example, the structure of the *CUFIM-PRO A folder* would look like Figure 55.

5. From the *CUFIM-PRO A folder/ CUFIM-PRO A Program BACKUP Files*, locate the *CUFIM-PRO A* file and launch it.

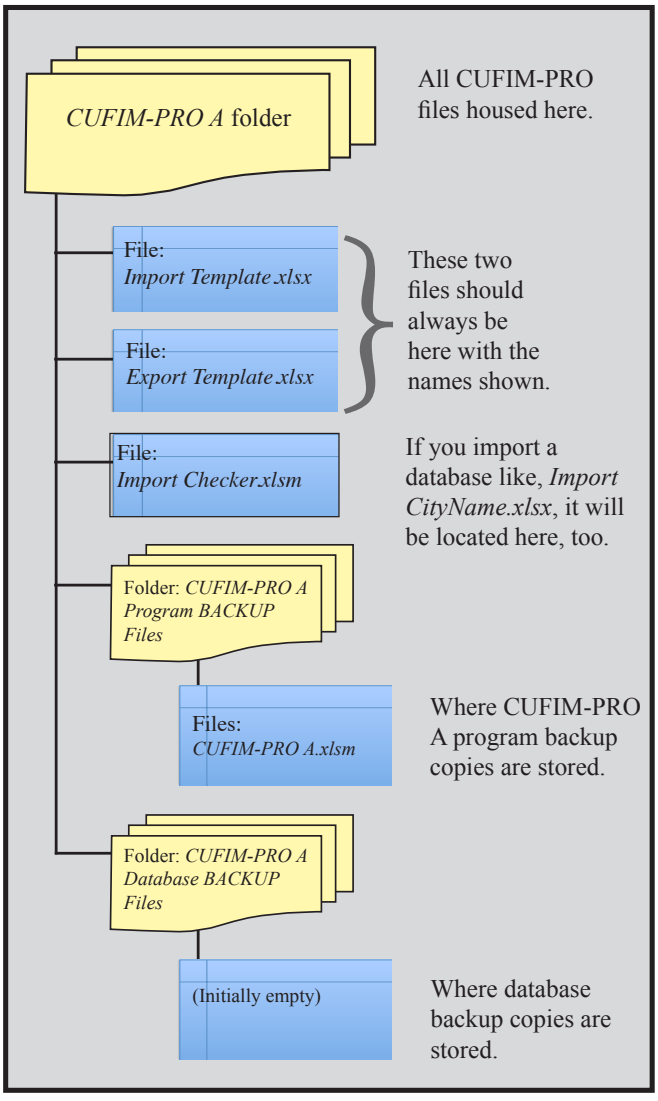
Select the second one-half of the tree records (for this example, say, 22,000 to 45,000), located in cells O22,000+54 and AO45000+54, that is, cells O22054:AO45054, and delete them from the database (Edit/Clear/All). This leaves just the first half of the original database.

6. Export the this first "half" of the database from the ID sheet. Save *CUFIM-PRO A* and close it.

7. Open *CUFIM-PRO B* from the *CUFIM-PRO B folder/ CUFIM-PRO B Program BACKUP Files*. Select cells O54 to AO21999+54, or O54:AO22053 and delete the first half of the original database by using the Edit/Delete/Shift Cells Up menu options. This leaves just the second half of the original database.

8. Export the second "half" of the database using the Export function. Then Save *CUFIM-PRO B* and confirm the Export was saved to the correct folder (*CUFIM-PRO B Database BACKUP Files*) and note the file name. The name should be *CUFIM-PRO B Database mmm dd, yyyy--Time is hh mm ss*.

Figure 55. After database has been split, this is the file structure for CUFIM-PRO A *folder.*.



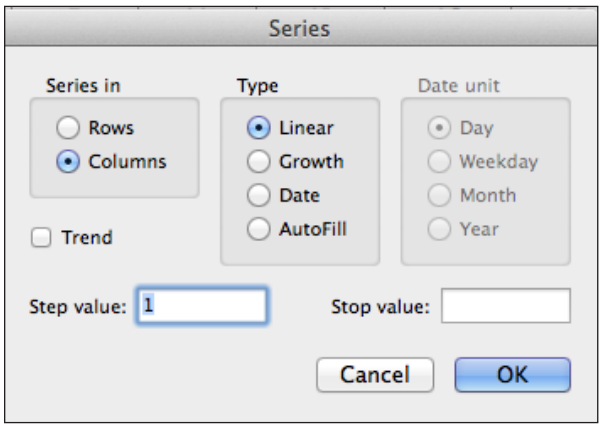
9. With *CUFIM-PRO B* activated and viewing the DATABASE sheet, split the screen horizontally so you can view the first and the last entry (approximately 22,050 rows down). Now, Sort by Tree Record Number.

10. If there are numbers in O54, clear the column from O54 to O50053. In cell 054 type "1" without the quotes.

11. Click on O54, hold shift key and click on the last cell of column O where tree data occurs (last Tree Record Number), then,

a. Edit/Fill/Series

b. In the Series window select Series in Columns, Type is Linear, Step Value is 1. Click OK.



12. Repeat steps 10 and 11 for Tree Record Number, column 2, starting in cell P54.

13. To format this "half" of the database: Under the blue "To Enter a Record" function box, click item #2, *"*** 2. If Needed, After Entries are Done, Click here to FORMAT Database Entries.***"*

14. Select the ID sheet, and double click on cell S28 that currently says "*CUFIM-PRO Database*." Add "B" so it reads: "*CUFIM-PRO B Database*."

15. Save and close *CUFIM-PRO B*.

16. Open *CUFIM-PRO A* and repeat steps 9-15.

17. Finally, Export the new databases from both *CUFIM-PRO A* and *CUFIM-PRO B* to be sure the system is functioning properly.

Appendix E. Growth Projections in Lieu of a New Inventory

The instructions for updating the urban forest database using the growth projection method are outlined here. These should be done by a person with excellent Excel skills.

1. Warning: This method will only approximate the growth between inventories. It will never be as good as a complete re-inventory.
2. Warning: This method is valid only if the tree diameters of the current database are measured to 0.1" and heights to ±5', although to the nearest foot provides greater accuracy.
3. Warning: A 10-year interval is needed in order to obtain measurable dbh and height growth. Data from an interval less than 10 years may have large variances and not provide reasonable results.
4. First, export all data (from CUFIM-PRO's ID sheet), then from the DATABASE sheet, Print all records for a backup. The Print dropbox shows how many pages will be printed.
5. Set up a new CUFIM-PRO folder to house the new inventory files with a new name, e.g., CUFIM-PRO 2025. Its name on the ID page, cell S28, will must be changed to the new name, too, with exact spelling.
6. At 10 years from the last measurement period, measure 10-20 trees per species group for dbh and height. Trees should include all sizes, small, medium and large. An example of measurements is shown in Figure 53.

   This is the blue gum Species Group. It would include blue gum *(E. globulus)* but also might include red gum *(E. camaldulensis)*, and any other tree the you defined as part of that Species Group. The trees measured should include one or two trees of each species of their species group.
7. Calculate diameter and height growth from the data in Figure 53 by these equations.

$$\text{Percent dbh growth for interval} = \frac{\text{Ave dbh in 2025 - Ave dbh in 2015}}{\text{Ave dbh in 2015}} \times 100$$

$$\text{Percent dbh growth for interval} = \frac{26.5 - 25.5}{25.5} \times 100 = 3.92\%$$

$$\text{Percent height growth for interval} = \frac{83.8 - 76.6}{76.6} \times 100 = 9.40\%$$

Follow the same approach and calculations for each species group. These percentages will be used to calculate new volumes, i.e., volumes projected to the year 2025, in this example.

Keep in mind that these values are only approximate especially given the small sample size and that we are mixing trees of different ages that certainly grow at different rates, as well as different species in the species group. However, this rough estimate may be all your program can afford at this time.

Species Group: blue gum				
Tree Record Number	Dbh in 2015	Height in 2015	Dbh in 2025	Height in 2025
25	41.0	122	42.0	128
71	6.0	18	7.0	30
72	21.0	78	21.5	84
91	26.0	82	26.9	88
121	16.0	58	16.6	66
588	44.0	118	45.8	124
599	35.0	94	36.1	100
821	32.0	90	33.4	96
822	19.0	70	19.6	78
823	15.0	36	15.8	44
Average =	25.5	76.6	26.5	83.8

Figure 53. Example of data needed for a 10-year growth projection.

Volume Coefficient Lookup Table

							V	W
							2015-2025	
Species used for Species Group	Species Group	Local vol coef. a	Local vol coef. b	m dia class 0	m dia class >20	m dia class Total	% Dbh Growth	% HT Growth
	-1	0	0					
Blue Gum	1	0.055113	2.436970	.0	54.0	100.0	1.0392	1.0940
Acacia	2	0.048490	2.347250	.0	24.3	100.0	1.0110	1.0230
Monterey Pine	3	0.019874	2.666079	.0	51.0	100.0	1.0370	1.0400
Monterey Cypress	4	0.035598	2.495263	.0	59.0	100.0	1.0090	1.0120
Carob	5	0.066256	2.128861	.0	26.0	100.0	1.3030	1.0390
Camphor	6	0.031449	2.534660	.6	12.1	100.0	1.0000	1.0000
Chinese Elm	7	0.028530	2.639347	.0	6.7	100.0	1.0392	1.0519
Holly Oak	8	0.025169	2.607285	.1	2.9	100.0	1.0110	1.0230
Jacaranda	9	0.036147	2.486248	.6	6.9	100.0	1.0370	1.0400
Liquid Ambar	10	0.030684	2.560469	.7	8.5	100.0	1.0090	1.0120
Modesto Ash	11	0.022227	2.633462	.0	16.0	100.0	1.3030	1.0390
Sawleaf Zelkova	12	0.021472	2.674757	.0	14.0	100.0	1.0400	1.0500
Chinese Pistache	13	0.019003	2.808625	.0	1.0	100.0	1.0392	1.0519
Southern Magnolia	14	0.022744	2.622015	.0	15.0	100.0	1.0110	1.0230
London Plane	15	0.025170	2.673578	.0	21.0	100.0	1.0370	1.0400
Sycamore	16	0.075051	2.335230	.6	6.9	100.0	1.0090	1.0120
Redwood	17	0.044697	2.390837	.0	51.0	100.0	1.3030	1.0390
Coast Live Oak	18	0.042542	2.466100	.0	15.0	100.0	1.0000	1.0000
Tanoak	19	0.119906	2.028700	.7	8.5	100.0	1.0100	1.0250
	20							

Figure 54. Illustration on how to add growth data to the Volume Coefficient Lookup Table.

WARNING: This method of projection should NEVER be used more than once. After the next 10-year interval, all trees in the urban forest must be re-measured in order to have confidence in the database numbers and management recommendations.

8. In CUFIM-PRO, select the SETUP sheet and expand column V and W as needed, Figure 54.

a) In cell V33-34, type "% Dbh Growth", and in cell V35, type the percent you calculated in step 7 above divided by 100 plus 1. E.g., 1 + (3.92%÷100) = 1.0392. Cell V35 was used as this is the line for blue gum.

b) Repeat step 8a in column W for height.

c) Continue inserting growth rates for each species group in columns V and W on the appropriate line.

d) All cells in Columns V and W on the SETUP sheet must have a value entered. If a species group isn't found in your community, type a "1" (without the quotes) in the cell. Doing this retains the 2015 dbh and height values; otherwise the value incorrectly goes to zero.

If a city has 10 species groups, then about 150 trees would be measured to estimate the growth rate of dbh and height for those groups (15 trees per group x 10 groups). This is a huge time and money saver compared to measuring thousands of trees.

9. Select the DATABASE sheet to calculate diameter and height growth.

a) Expand columns AP and AQ as needed. In cell AP53, type "Dbh in 2025" and in AQ53 type "Ht in 2025."

b) In cell AP54, the Dbh in 2025 column, carefully enter this formula and press enter:

=IF(LOOKUP(T54,SETUP!$I$35:$I$53)=T54,LOOKUP(T54,SETUP!$I$35:$V$53)*U54,"")

c) In cell AQ54, the Ht in 2025 column, enter this similar formula, except it first checks to see if a height value has been entered.

=IF(V54>0,IF(LOOKUP(T54,SETUP!$I$35:$I$53)=T54,LOOKUP(T54,SETUP!$I$35:$W$53)*V54,""),"")

See side bar, below, for explanation.

d) Fill down both equations to include all rows (all trees) in the database. This could be 5,000 to 45,000 rows to fill down.

e) Doing so increments the dbh and height of the species groups in your city which should be all the trees in your database. Scroll down and spot check to make certain that the new dbh and height values are larger than the original values. See Figure 55.

f) First, for the dbh column, split the screen and select all 2025 dbh values. Then Copy, and Paste Special/Values onto itself to replace the formulas in column AP. Repeat the process for tree heights in column AQ.

U	V	W	AO	AP	AQ	A
(7)	(8)	(9) %, cf	(26)			
Count	Count	100.0	Count			
21,344	1	1,070,423	0			
Sort			Sort			
Tree Dbh (in)	Tree Height (ft)	Volume (cf)	(10) Variable 10	Dbh in 2025	Ht in 2025	
101.0		4224.0		104.959		
20.7	62.0	65.8		21.480	67.828	
19.0		72.0		19.745		
22.1		104.1		22.966		
20.7		88.4		21.480		
38.4		399.8		39.891		

Figure 55. Result of adding dbh and height growth equations in columns AP and AQ. Note that the second tree in the database has a height, and the new height shows in column AQ, but not the ones without an original height.

g) Select and copy the 2025 dbh values, all 5,000 to 45,000 of them, and paste them into column U starting at U54. Do the same for 2025 height values, starting in cell V54. This will replace all 2015 values with 2025 dbh's and heights.

h) The volume shown in column W will automatically be updated to estimate 2025 volumes. Check to assure yourself that they are larger than those on your hard copy print out.

i) Click on "*Click her to Update Cubic Feet of Tree volume by Branch Diameter Size*" heading that starts near cell W52.

j) Your database dbh's, heights and volumes have now been updated.

9. Export and backup. Earlier you set up a new folder for the 2025 data. When backing up this data must go into that folder.

   a) Set up a CUFIM-PRO 2025 folder if you haven't already done so. See item #5 above.

   b) On the ID sheet and click on Export. When the export process is finished, open the exported file and check that it came through as expected.

   c) Now you are ready to conduct new CUFIM-PRO analyses as needed. They can be compared to previous printouts from 2015.

Calculation of DBH Growth

This formula looks in the SETUP sheet first to determine if the species group is present, and secondly to retrieve the dbh percent growth between 2015 and 2025, for this example, and multiply it by 2015 dbh to calculate an estimate of the 2025 dbh. The height equation does the same.

=IF(LOOKUP(T54,SETUP!$I$35:$I$53)=T54,LOOKUP(T54,SETUP!$I$35:$V$53)*U54,"")

Appendix F. Fifteen Urban Species in California for Matching Volume Characteristics

The follow photos are for the 15 urban species listed on the SETUP sheet. The purpose of presenting them here is matching a different species to one of the 15 with the most similar growth pattern and volume characteristics.

They are:

Acacia *(Acacia longifolia)*
American Sweet Gum *(Liquidambar styraciflua)*
Blue Gum *(Eucalyptus globulus)*
Camphor Tree *(Cinnamomum camphora)*
Carob *(Ceratonia siliqua)*
Chinese Elm *(Ulmus parvifolia chinensis)*
Chinese Pistache *(Pistacia chinensis)*
Holly Oak *(Quercus ilex)*
Jacaranda *(Jacaranda mimosaifolia)*
London Plane *(Platanus acerifolia)*
Modesto Ash *(Fraxinus velutina 'Modesto')*
Monterey Cypress *(Cupressus macrocarpa)*
Monterey Pine *(Pinus radiata)*
Sawleaf Zelkova *(Zelkova serrata)*
Southern Magnolia *(Magnolia grandiflora)*

The source of this study is:

Pillsbury, N. H., J.L. Reimer and R.P. Thompson. 1998. *Tree Volume Equations for Fifteen Urban Species in California*. Technical Report No. 7, Urban Forest Ecosystems Institute, Natural Resources Management Department, California Polytechnic State University, San Luis Obispo. 56 p. plus 942 pages in Appendix.

*Figure 56.1. Camphor Tree (*Cinnamomum camphora*) (l. to r., small, medium, large).*

*Figure 56.2. Chinese Elm (*Ulmus parvifolia chinensis*) (l. to r., small, medium, large).*

*Figure 56.3. Holly Oak (*Quercus ilex*) (l. to r., small, medium, large).*

*Figure 56.4 Jacaranda (*Jacaranda mimosaifolia*) (l. to r., small, medium, large).*

*Figure 56.5 American Sweet Gum (*Liquidambar styraciflua*) (l. to r., small, medium, large).*

*Figure 56.6 Monterey Pine (*Pinus radiata*) (l. to r., small, medium, large).*

*Figure 56.7 Blue Gum (*Eucalyptus globulus*) (l. to r., small, medium, large).*

*Figure 56.8 Monterey Cypress (*Cupressus macrocarpa*) (l. to r., small, medium, large).*

*Figure 56.9 Acacia (*Acacia longifolia*) (l. to r., small, medium, large).*

*Figure 56.10 Carob (*Ceratonia siliqua) *(l. to r., small, medium, large).*

*Figure 56.11 Modesto Ash*Fraxinus velutina 'Modesto' *(l. to r., small, medium, large).*

*Figure 56.12 Southern Magnolia (*Magnolia grandiflora) *(l. to r., small, medium, large).*

*Figure 56.13 Sawleaf Zelkova (*Zelkova serrata*) (l. to r., small, medium, large).*

*Figure 56.14 London Plane (*Platanus acerifolia*) (l. to r., small, medium, large).*

*Figure 56.15 Chinese Pistache (*Pistacia chinensis*) (l. to r., small, medium, large).*

Appendix G.
Source Code for CUFIM-PRO

Macros in alphabetical order (June 14, 2015)

```
Sub ActualNoTreesRemoved()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("REMOVAL").Select
Dim StandRandom#(15, 1)     'Data Matrix for Stand Table [15 dbh classes]
Dim B#(15, 1)               'Array for upper dbh class values

B(1, 1) = Range("K26")   'Upper diameter for dbh class 1
B(2, 1) = Range("K27")
B(3, 1) = Range("K28")
B(4, 1) = Range("K29")
B(5, 1) = Range("K30")
B(6, 1) = Range("K31")
B(7, 1) = Range("K32")
B(8, 1) = Range("K33")
B(9, 1) = Range("K34")
B(10, 1) = Range("K35")
B(11, 1) = Range("K36")
B(12, 1) = Range("K37")
B(13, 1) = Range("K38")
B(14, 1) = Range("K39")
B(15, 1) = Range("K40")   'Upper diameter for dbh class 15

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in database

Sheets("DATABASE").Select
'================================
   'For A = ;  searches total database
   'Dim B array is number of dbh classes
   'For C =;   is number of dbh classes
   '54 Selects first possible tree in database
   'If Cells(A,21)<>;    if dbh of current tree is less than upper dbh of dbh class
                        'in C loop dbh class, then....
'================================

   For A = 54 To MaxNo + 54
      If Cells(A, 18) <> "" Then            'Removal Code Col.
         For C = 1 To 15
            If Cells(A, 21) < B(C, 1) Then
               StandRandom(C, 1) = StandRandom(C, 1) + 1: C = 15
            End If
         Next C
      End If
   Next A

Sheets("REMOVAL").Select

'===========
'Outputs Stand Table data to Actual no.of Trees removed column
If Range("Z5") = True Then            'Box B selected
   Range("Q26").Select
   For C = 1 To 15
      Range(Cells(C + 25, 34), Cells(C + 25, 34)).Select
      If StandRandom(C, 1) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = StandRandom(C, 1)
      End If
   Next C

   Range("AH20").Select
End If

'Outputs Stand Table data to Actual no.of Trees removed column
If Range("AA5") = True Then            'Box A selected
   Range("Q26").Select
   For C = 1 To 15
      Range(Cells(C + 25, 17), Cells(C + 25, 17)).Select
      If StandRandom(C, 1) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = StandRandom(C, 1)
      End If
   Next C

   Range("Q20").Select
```

```
End If

GoTo ENDTREES:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err) & vbCr &
vbCr & "Check for missing values in Database." & vbCr
& "Or for invalid species codes."

ENDTREES:


'==========
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub




Sub ADDnewrecord()
'
' ADDnewrecord Macro
' Macro recorded 8/22/2002 by Norman Pillsbury
'
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

If Range("R25") = "" Or Range("R25") < 1 Then
   RndMsg = MsgBox("You MUST include a valid Spe-
cies Code.")
   Sheets("DATABASE").Select
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
Exit Sub
End If


If Range("R26") = "" Or Range("R26") < 1 Then
   RndMsg = MsgBox("You MUST include a valid Tree
Dbh.")
   Sheets("DATABASE").Select
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
Exit Sub
End If

Sheets("DATABASE").Select
If Range("CU23") <> "Sort Tree Seq Number" Then
   Application.Run "SortTREESEQNUMBER"
End If

'Does this if Sequence Number equals 1.
   Application.Run ("LoadDBeqn")  'Loads eqns into S54,
T54 and W54.
If Range("R23") = 1 Then
   Range("R23:R25").Select
   Selection.Copy
   Range("O54").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=True
   Application.CutCopyMode = False
   Range("R26:R27").Select
   Selection.Copy
   Range("U54").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=True
   Application.CutCopyMode = False

   Range("R28:R37").Select
   Selection.Copy
   Range("AF54").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=True
   Application.CutCopyMode = False

   Range("O54:AO50053").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With
   Range("R25").Select
End If

   'Seq No., Tree Record No., and Species Code. Does this
if Seq. No. is greater than 1.
If Range("R23") > 1 Then
   Range("R23:R25").Select
   Selection.Copy
   Range("O50053").End(xlUp).Select
   ActiveCell.Offset(1, 0).Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=True
   Application.CutCopyMode = False

   'Tree dbh and tree height
   Range("R26:R27").Select
   Selection.Copy
```

```
   Range("U50053").End(xlUp).Select
   ActiveCell.Offset(1, 0).Select

   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=True
   Application.CutCopyMode = False

   'Species Group Number (this brings down equation for
   'Need to skip this the first time
   If Cells(23, 18) <> 1 Then
      ActiveCell.Offset(-1, -2).Select
        Selection.Copy
      ActiveCell.Offset(1, 0).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False
   End If

   'Species Group Name
   If Cells(23, 18) <> 1 Then
      ActiveCell.Offset(-1, 1).Select
      Selection.Copy
      ActiveCell.Offset(1, 0).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False
   End If

   'Volume
    If Cells(23, 18) <> 1 Then
      ActiveCell.Offset(-1, 3).Select
      Selection.Copy
      ActiveCell.Offset(1, 0).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False
   End If

   '=========================
   'User Defined Variables 1 through 10
   Range("R28:R37").Select
   Selection.Copy
   Range("AF50053").End(xlUp).Select
   ActiveCell.Offset(1, 0).Select

   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=True
   Application.CutCopyMode = False
End If

   '=========================
   Range("R25:R37").Select
   Selection.ClearContents

   Range("CU23").Select
   ActiveCell.FormulaR1C1 = "Sort Tree Seq Number"
GoTo ENDADD:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err) & vbCr &
vbCr & "Check New Tree data carefully." & vbCr & "And
try again."

ENDADD:
Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

   Range("R25").Select
End Sub


Sub AddNewSpecies()
'
' AddNewSpecies Macro
' Macro recorded 6/5/02 by NRM

Sheets("SETUP").Select

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

For A = 1 To 536
   If Cells(A + 35, 2) = Cells(14, 2) Then
      CodeMsg = MsgBox("Sorry, that Species Code has
been used.")
      Exit Sub
   End If
Next A

      Range("B14:D14").Select
      Selection.Copy
      Range("B34").Select
      Selection.End(xlDown).Select
      ActiveCell.Offset(1, 0).Select

      Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False

      Application.CutCopyMode = False

       Range("B36:D540").Select
      Selection.Sort Key1:=Range("B36"),
```

```
Order1:=xlAscending, Key2:=Range("C36" _
        ), Order2:=xlAscending, Key3:=Range("D36"),
Order3:=xlAscending, Header _
        :=xlNo, OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

      Range("B14:D14").Select
      Selection.ClearContents
      Range("T14").Select

   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic

End Sub



Sub AssignValuebyGroup()
'
' AssignValuebyGroup Macro
' Macro recorded 7/22/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

If Range("X10") = False Then
   Range("X9") = False
   Range("Y22").Select
   Selection.Clear

Exit Sub
End If

If Range("X9") = False Then
   'Range("Y22:AK76").Select
   'Selection.Clear

   Range("AL21").Select
   Selection.AutoFill Destination:=Range("AL21:AL76"),
Type:=xlFillDefault
   Range("AL21:AL76").Select
   Selection.AutoFill Destination:=Range("Y21:AL76"),
Type:=xlFillDefault
   'Selection.AutoFill Destination:=Range("Y22:W22"),
Type:=xlFillDefault

   Range("Y22").Select
   Selection.Clear

   Range("O25:O26").Select
   Selection.Borders(xlRight).LineStyle = xlContinuous

   Range("W9").Select

Exit Sub
End If

If Range("X8") = True Then
   Range("X9") = False
Exit Sub
End If

If Range("X9") = True Then
   Range("X8") = False

   Columns("Z:AI").ColumnWidth = 0
   Range("Y22").Select
   Selection.Clear

   Range("Y25:Y76").Select
   Range("Y76").Activate
   Selection.Borders(xlRight).LineStyle = xlContinuous
   Range("Y25").Select
   ActiveCell.FormulaR1C1 = "Value"
   Range("Y26").Select
   ActiveCell.FormulaR1C1 = "($/cf)"
   Range("Y25:Y26").Select
   With Selection
      .HorizontalAlignment = xlCenter
   End With

   Range("W25:Y26").Select
   With Selection.Font
      .Name = "Geneva"
      .Size = 9
   End With
   Selection.Font.Bold = True
   Range("V30").Select

   Columns("X:X").ColumnWidth = 16.17
   Columns("Y:Y").ColumnWidth = 9.5

   Range("Y27:Y76").Select
   With Selection.Interior
      .ColorIndex = 36
      .Pattern = xlSolid
   End With

   Range("Y27:Y76").Select
   Selection.NumberFormat = "$#,##0.00  "
   With Selection
      .HorizontalAlignment = xlRight
   End With

   Range("Y25").Select
   Selection.Borders(xlTop).LineStyle = xlContinuous
```

```
    Range("Y27:Y76").Select
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Range("Y76").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Range("W1").Select

    Range("Y25:Y26").Select
    Selection.AutoFill Destination:=Range("Y25:AK26"),
Type:=xlFillDefault
    Range("Y25:AK26").Select
    Range("AJ25").Select
    ActiveCell.FormulaR1C1 = "Volume"
    Range("AJ26").Select
    ActiveCell.FormulaR1C1 = "(cf)"
    Range("AK25").Select
    ActiveCell.FormulaR1C1 = "Total Value"
    Columns("AK:AK").ColumnWidth = 12
    Range("AK26").Select
    ActiveCell.FormulaR1C1 = "($)"
    Range("AJ27:AK76").Select
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("Y76:AK76").Select
    Range("AK76").Activate
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Range("AK23").Select
    Columns("AJ:AJ").ColumnWidth = 12

    Range("Y26:AK26").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous

    Range("AJ17:AL17").Select
    Selection.Borders(xlBottom).LineStyle = xlNone

End If

    '====================================
    ' 1 -- BEGIN VOLUME CALCULATION FOR SPE-
CIES GROUPS
    '====================================

    Sheets("VALUATION").Select
    Dim C(50, 1)

    Range("Y22").Select
    Selection.Clear
    Range("Y23").Select

    Sheets("DATABASE").Select

    'sort by removal, then by species GROUP code
    'Range("R54:R55").Select

    If Range("CU23") <> "Sort Removal/GROUP" Then
       Range("P54:AC50053").Select
       Selection.Sort Key1:=Range("R54"),
Order1:=xlAscending, Key2:=Range("T54" _
          ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
          False, Orientation:=xlTopToBottom

       Range("CU23").Select
       ActiveCell.FormulaR1C1 = "Sort Removal/GROUP"
    End If

    Range("R54").Select
    For A = 1 To 50000

       If Cells(A + 53, 18) = 1 Then
          C(Cells(A + 53, 20), 1) = C(Cells(A + 53, 20), 1) +
Cells(A + 53, 23)
       End If

       If Cells(A + 53 + 1, 18) = "" Then
          A = 50000
       End If

    Next A

    'sort by species GROUP code
    Sheets("VALUATION").Select
    Range("W27:X76").Select
    Selection.Sort Key1:=Range("W27"),
Order1:=xlAscending, Header:=xlGuess, _
       OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

    Range("W27").Select
    For A = 1 To 50
       Range(Cells(A + 26, 36), Cells(A + 26, 36)).Select
       ActiveCell.FormulaR1C1 = C(Cells(A + 26, 23), 1)

       If Cells(A + 27, 23) = "" Then
          A = 50
       End If
    Next A

    Range("AK27").Select
    ActiveCell.FormulaR1C1 = "=IF(RC[-1]>0,RC[-
1]*RC[-12],)"
    Range("AK27:AK76").Select
    Selection.FillDown

    Range("AJ27:AJ76").Select
    Selection.NumberFormat = "#,##0   "
    Range("AK27:AK76").Select
    Selection.NumberFormat = "$#,##0   "
    Range("AK76").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous

    Range("Y22").Select
    ActiveCell.FormulaR1C1 = "Totals ="
     With Selection.Font
      .Name = "Geneva"
      .Size = 9
```

```
    End With
    Selection.Font.Bold = True
    With Selection
        .HorizontalAlignment = xlRight
        .VerticalAlignment = xlCenter
    End With

    Range("AJ22").Select
    ActiveCell.FormulaR1C1 = "=SUM(R[5]C:R[54]C)"
    Selection.Font.Bold = True
    With Selection.Font
        .Name = "Geneva"
        .Size = 9
    End With
    Selection.AutoFill Destination:=Range("AJ22:AK22"),
Type:=xlFillDefault
    Range("AJ22:AK22").Select
    Range("AK22").Select
    Selection.NumberFormat = "$#,##0   "
    Columns("AK:AK").ColumnWidth = 13.17

    Range("Y22:AK22").Select
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Selection.Borders(xlTop).LineStyle = xlContinuous
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    With Selection.Interior
        .ColorIndex = 35
        .Pattern = xlSolid
    End With
    With Selection
        .VerticalAlignment = xlCenter
    End With



    Range("AJ25:AK26").Select
    With Selection.Interior
        .ColorIndex = 36
        .Pattern = xlSolid
    End With

    Range("W25:W26").Select
    With Selection.Interior
        .ColorIndex = 36
        .Pattern = xlSolid
    End With

    Range("W9").Select
    '====================================
    'Dim statusMode
    'statusMode = Application.DisplayStatusBar
    'Application.DisplayStatusBar = True
    'Application.StatusBar = "Please Standby --- Recalcu-
lating Entire Data Set…"
    'MsgBox ("Please Standby  ---  Recalculating Entire
Data Set…")
    '====================================


GoTo ENDASSIGNGROUP:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDASSIGNGROUP:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

    With Application
        .Calculation = xlAutomatic
        .MaxChange = 0.001
    End With
    '====================================
    'Application.StatusBar = False
    'Application.DisplayStatusBar = statusMode

    '====================================
End Sub



Sub AssignValuebyTreeSize()
'
' AssignValuebyTreeSize Macro
' Macro recorded 7/12/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

    'Range("Z27:AB76").Select
    'Selection.Clear

If Range("X10") = False Then
    Range("X9") = False
    Range("X8") = False
    Range("Y22").Select
    Selection.Clear

    Range("X4:X6").Select
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlMedium
```

```
      .ColorIndex = 3
   End With

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

Exit Sub
End If


If Range("X9") = True Then
   Range("X8") = False

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

Exit Sub
End If



If Range("X8") = False Then
   Range("Z24:AI76").Select
   Selection.Clear
   Columns("Y:Y").ColumnWidth = 9.5
   Columns("Y:AK").ColumnWidth = 0
   Columns("AL:AN").ColumnWidth = 9.5

   Range("W11:AN19").Select
   With Selection.Interior
      .ColorIndex = 37
      .Pattern = xlSolid
   End With
   Range("W11:AN11").Select
   Selection.Borders(xlTop).LineStyle = xlContinuous
   Range("AN11:AN19").Select
   Selection.Borders(xlRight).LineStyle = xlContinuous
   Range("W19:AN19").Select
   Selection.Borders(xlBottom).LineStyle = xlContinuous

   Range("W9").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

   Exit Sub
   End If

If Range("X8") = True Then

   Range("AJ11:AN19").Select
   Selection.Interior.ColorIndex = xlNone
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   Selection.Borders(xlEdgeLeft).LineStyle = xlNone
   Selection.Borders(xlEdgeTop).LineStyle = xlNone
   Selection.Borders(xlEdgeBottom).LineStyle = xlNone
   Selection.Borders(xlEdgeRight).LineStyle = xlNone
   Selection.Borders(xlInsideVertical).LineStyle = xlNone
   Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone

   Range("AI11:AI19").Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   Selection.Borders(xlEdgeLeft).LineStyle = xlNone
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone
   Range("W21").Select

   Range("W11:AI19").Select
   With Selection.Interior
      .ColorIndex = 37
      .Pattern = xlSolid
   End With
   Range("W9").Select

End If
```

```
    Range("AJ11:AL17").Select
    Selection.Clear

    Columns("Z:AB").ColumnWidth = 9.5
    Range("Y27:Y76").Select
    Selection.ClearContents
    Selection.Interior.ColorIndex = 36
    Columns("Y:Y").ColumnWidth = 0

    Range("Z25").Select
    ActiveCell.FormulaR1C1 = "Small"
    Range("AA25").Select
    ActiveCell.FormulaR1C1 = "Medium"
    Range("AB25").Select
    ActiveCell.FormulaR1C1 = "Large"
    Range("Z26").Select
    ActiveCell.FormulaR1C1 = "<12""""
    Range("AA26").Select
    ActiveCell.FormulaR1C1 = "12-24""""
    Range("AB26").Select
    ActiveCell.FormulaR1C1 = ">24""""

    Range("Z25:AB26").Select
    With Selection
        .HorizontalAlignment = xlCenter
    End With
    With Selection.Font
        .Name = "Geneva"
        .Size = 9
    End With
    Selection.Font.Bold = True
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone
    Range("Y27:Y76").Select
    Selection.AutoFill Destination:=Range("Y27:AB76"),
Type:=xlFillDefault
    Range("Y27:AB76").Select
    Range("W9").Select
'==================
    Range("AA24").Select
    ActiveCell.FormulaR1C1 = "Value by Size ($/cubic
foot)"
    With Selection
        .HorizontalAlignment = xlCenter
    End With
    Selection.Font.Bold = True
    With Selection.Font
        .Name = "Geneva"
        .Size = 10
        .Underline = xlUnderlineStyleNone
    End With

    Range("Z24:AB24").Select
    'Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Range("Z24:AB26").Select
    Selection.Copy
    Range("AC24").Select
    ActiveSheet.Paste
    Application.CutCopyMode = False

    Range("AD24").Select
    ActiveCell.FormulaR1C1 = "Volume by Size (cf)"

    Range("AC24:AE26").Select
    Selection.Copy
    Range("AF24").Select
    ActiveSheet.Paste
    Application.CutCopyMode = False
    Range("AG24").Select
    ActiveCell.FormulaR1C1 = "Value by Size ($)"
    Range("AC27:AH76").Select
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("Z76:AH76").Select
    Range("AH76").Activate
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Columns("AC:AH").Select
    Range("AC3").Activate
    Selection.ColumnWidth = 9.5
    Range("AE33").Select
    Columns("AI:AI").ColumnWidth = 12
    Range("AG24").Select
    Selection.Copy
    Application.CutCopyMode = False
```

```
    Range("AI24").Select
    ActiveCell.FormulaR1C1 = "Total"
    Range("AI25").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Selection.Borders(xlEdgeLeft).LineStyle = xlNone
    Selection.Borders(xlEdgeTop).LineStyle = xlNone
    Selection.Borders(xlEdgeBottom).LineStyle = xlNone
    Selection.Borders(xlEdgeRight).LineStyle = xlNone
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone
    ActiveCell.FormulaR1C1 = "Value"
    With ActiveCell.Characters(Start:=1, Length:=0).Font
        .Name = "Geneva"
        .FontStyle = "Bold"
        .Size = 9
    End With
    Range("AI26").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Selection.Borders(xlEdgeLeft).LineStyle = xlNone
    Selection.Borders(xlEdgeTop).LineStyle = xlNone
    Selection.Borders(xlEdgeBottom).LineStyle = xlNone
    Selection.Borders(xlEdgeRight).LineStyle = xlNone
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone
    ActiveCell.FormulaR1C1 = "($)"
    With ActiveCell.Characters(Start:=1, Length:=0).Font
        .Name = "Geneva"
        .FontStyle = "Bold"
        .Size = 9
    End With
    Range("AI24").Select
    With Selection
        .HorizontalAlignment = xlCenter
    End With
    With Selection.Font
        .Name = "Geneva"
        .Size = 9
    End With
    Selection.Font.Bold = True
    Range("AI26").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Range("AI24:AI76").Select
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Range("AI24:AI26").Select
    With Selection.Font
        .Name = "Geneva"
        .Size = 10
    End With
    Range("W9").Select

'====================
    Range("Y27:Y76").Select
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Selection.NumberFormat = "$#,##0.00 "
    With Selection
        .HorizontalAlignment = xlRight
    End With

    Range("Z24:AB24").Select
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    With Selection.Borders(xlBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Range("AC24:AE24").Select
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    With Selection.Borders(xlBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Range("AF24:AH24").Select
```

```
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    With Selection.Borders(xlBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Range("AI24").Select
    Selection.Borders(xlTop).LineStyle = xlContinuous
    Range("AI76").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Range("AI25:AI26").Select
    With Selection
        .HorizontalAlignment = xlCenter
    End With
    Range("AF27:AI76").Select
    Selection.NumberFormat = "$#,##0  "
    With Selection
        .HorizontalAlignment = xlRight
    End With
    Range("W9").Select

    Range("Z24:AB24").Select
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With

    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    Range("AC24:AE24").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    Range("AF24:AH24").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
    End With

    Range("AJ22:AK22").Select
    Selection.AutoFill Destination:=Range("AJ22:AK76"),
Type:=xlFillDefault
    Range("AJ22:AK76").Select

    Range("AI24:AI26").Select
    With Selection.Interior
        .ColorIndex = 36
        .Pattern = xlSolid
    End With

    '====================================
    ' 2 -- BEGIN VOLUME CALCULATION FOR SPE-
CIES by TREE SIZE
    '====================================

    Sheets("VALUATION").Select
    Dim D(50, 3)

    Range("Z27:AB76").Select
    Selection.Clear
    Range("W9").Select

    Sheets("DATABASE").Select

    'sort by removal, then by species GROUP, then by DBH
code
```

```
  If Range("CU23") <> "Sort Removal/GROUP/DBH"
Then
     Range("P54:AC50053").Select
     Selection.Sort Key1:=Range("R54"),
Order1:=xlAscending, Key2:=Range("T54" _
        ), Order2:=xlAscending, Key3:=Range("U54"),
Order3:=xlAscending, Header:=xlGuess, OrderCus-
tom:=1, MatchCase:= _
        False, Orientation:=xlTopToBottom

     Range("CU23").Select
     ActiveCell.FormulaR1C1 = "Sort Removal/GROUP/
DBH"
  End If

  Range("R54").Select
  For A = 1 To 50000
    If Cells(A + 53, 18) = 1 Then
      If Cells(A + 53, 21) < 12 Then
        D(Cells(A + 53, 20), 1) = D(Cells(A + 53, 20), 1)
+ Cells(A + 53, 23)
      End If

      If Cells(A + 53, 21) >= 12 And Cells(A + 53, 21)
<= 24 Then
        D(Cells(A + 53, 20), 2) = D(Cells(A + 53, 20), 2)
+ Cells(A + 53, 23)
      End If

      If Cells(A + 53, 21) > 24 Then
        D(Cells(A + 53, 20), 3) = D(Cells(A + 53, 20), 3)
+ Cells(A + 53, 23)
      End If

      If Cells(A + 53 + 1, 18) = "" Then
        A = 50000
      End If
    End If
  Next A

  'sort by species GROUP code
  Sheets("VALUATION").Select
  Range("W27:X76").Select
  Selection.Sort Key1:=Range("W27"),
Order1:=xlAscending, Header:=xlGuess, _
    OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

  Range("AC27").Select
  For A = 1 To 50
    For B = 1 To 3
      Range(Cells(A + 26, B + 28), Cells(A + 26, B +
28)).Select
      ActiveCell.FormulaR1C1 = D(Cells(A + 26, 23),
B)
    Next B
    If Cells(A + 27, 23) = "" Then
      A = 50
    End If
  Next A

  Range("AF27").Select
  ActiveCell.FormulaR1C1 = "=IF(RC[-6]>0,RC[-
6]*RC[-3],"""")"

  Range("AF27:AH27").Select
  Selection.FillRight

  Range("AF27:AH76").Select
  Selection.FillDown

  Range("AI27").Select
  ActiveCell.FormulaR1C1 = "=IF(RC[-6]>0,RC[-
6]*RC[-3],"""")"

  Range("AF27:AH76").Select
  Selection.FillDown

  Range("AI27").Select
  ActiveCell.FormulaR1C1 = "=IF(SUM(RC[-3]:RC[-
1])>0,SUM(RC[-3]:RC[-1]),)"
  Selection.AutoFill Destination:=Range("AI27:AI76"),
Type:=xlFillDefault
  Range("AI27:AI76").Select
  Range("AI76").Select
  Selection.Borders(xlBottom).LineStyle = xlContinuous
  Range("W9").Select

  Range("AC27:AE76").Select
  Selection.NumberFormat = "#,##0   "
  Range("AF27:AI76").Select
  Selection.NumberFormat = "$#,##0   "
  Range("AK76").Select
  Selection.Borders(xlBottom).LineStyle = xlContinuous

  Range("AB22").Select
  ActiveCell.FormulaR1C1 = "Totals ="
   With Selection.Font
    .Name = "Geneva"
    .Size = 9
  End With
  Selection.Font.Bold = True
  With Selection
    .HorizontalAlignment = xlRight
  End With

  Range("AC22").Select
  ActiveCell.FormulaR1C1 = "=SUM(R[5]C:R[54]C)"
  Selection.Font.Bold = True
  With Selection.Font
    .Name = "Geneva"
    .Size = 9
```

```
  End With
  Selection.AutoFill Destination:=Range("AC22:AI22"),
Type:=xlFillDefault
  Range("AC22:AI22").Select
  Range("AI22").Select
  Selection.NumberFormat = "$#,##0  "
  Columns("AC:AI").ColumnWidth = 13
  Range("W9").Select

  Range("Z27:AB76").Select
  Selection.NumberFormat = "$#,##0.00  "
  With Selection
    .HorizontalAlignment = xlCenter
  End With
  Range("Z76:AB76").Select
  Selection.Borders(xlBottom).LineStyle = xlContinuous
  Range("AA25:AB76").Select
  Selection.Borders(xlLeft).LineStyle = xlContinuous
  Range("Z27:AB76").Select
  With Selection.Interior
    .ColorIndex = 36
    .Pattern = xlSolid
  End With
  Range("AF76:AH76").Select
  Selection.Borders(xlBottom).LineStyle = xlContinuous
  Range("AC24:AC76").Select
  With Selection.Borders(xlEdgeLeft)
    .LineStyle = xlContinuous
    .Weight = xlMedium
  End With
  With Selection.Borders(xlEdgeTop)
    .LineStyle = xlContinuous
    .Weight = xlThin
  End With
  With Selection.Borders(xlEdgeBottom)
    .LineStyle = xlContinuous
    .Weight = xlThin
  End With
  Range("AF24:AF76").Select
  With Selection.Borders(xlEdgeLeft)
    .LineStyle = xlContinuous
    .Weight = xlMedium
  End With
  With Selection.Borders(xlEdgeTop)
    .LineStyle = xlContinuous
    .Weight = xlThin
  End With
  With Selection.Borders(xlEdgeBottom)
    .LineStyle = xlContinuous
    .Weight = xlThin
  End With
  Range("AF22:AH22").Select
  Selection.NumberFormat = "$#,##0  "

  Range("AC22:AE22").Select
  Selection.Borders(xlLeft).LineStyle = xlContinuous
  Selection.Borders(xlRight).LineStyle = xlContinuous
  Selection.Borders(xlTop).LineStyle = xlContinuous
  Selection.Borders(xlBottom).LineStyle = xlContinuous

  Range("AF22:AH22").Select
  Selection.Borders(xlLeft).LineStyle = xlContinuous
  Selection.Borders(xlRight).LineStyle = xlContinuous
  Selection.Borders(xlTop).LineStyle = xlContinuous
  Selection.Borders(xlBottom).LineStyle = xlContinuous

  Range("AI22").Select
  Selection.Borders(xlLeft).LineStyle = xlContinuous
  Selection.Borders(xlRight).LineStyle = xlContinuous
  Selection.Borders(xlTop).LineStyle = xlContinuous
  Selection.Borders(xlBottom).LineStyle = xlContinuous

  Range("AB22").Select
  Selection.Borders(xlLeft).LineStyle = xlContinuous
  Selection.Borders(xlRight).LineStyle = xlContinuous
  Selection.Borders(xlTop).LineStyle = xlContinuous
  Selection.Borders(xlBottom).LineStyle = xlContinuous

  Range("AB22:AI22").Select
  With Selection.Interior
    .ColorIndex = 35
    .Pattern = xlSolid
  End With

  Range("AI24:AI76").Select
  With Selection.Borders(xlEdgeLeft)
    .LineStyle = xlContinuous
    .Weight = xlMedium
  End With
  With Selection.Borders(xlEdgeTop)
    .LineStyle = xlContinuous
    .Weight = xlThin
  End With
  With Selection.Borders(xlEdgeBottom)
    .LineStyle = xlContinuous
    .Weight = xlThin
  End With
  With Selection.Borders(xlEdgeRight)
    .LineStyle = xlContinuous
    .Weight = xlThin
  End With

  Range("AI24:AI76").Select
  With Selection.Borders(xlEdgeLeft)
    .LineStyle = xlContinuous
    .Weight = xlMedium
  End With
  With Selection.Borders(xlEdgeTop)
    .LineStyle = xlContinuous
    .Weight = xlThin
  End With
  With Selection.Borders(xlEdgeBottom)
    .LineStyle = xlContinuous
    .Weight = xlThin
```

```
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlMedium
   End With

   Range("AE21").Select
   Selection.Font.Bold = True
   With Selection.Font
      .Name = "Geneva"
      .Size = 9
   End With
   ActiveCell.FormulaR1C1 = "=SUM(R[1]C[-2]:R[1]C)"

   Range("AD21").Select
   ActiveCell.FormulaR1C1 = "Total Volume ="
   With Selection
      .HorizontalAlignment = xlRight
   End With
   Selection.Font.Bold = True
   With Selection.Font
      .Name = "Geneva"
      .Size = 9
   End With

   Range("AB22:AI22").Select
   With Selection
      .VerticalAlignment = xlCenter
   End With

   Range("AK76").Select
   Selection.Borders(xlBottom).LineStyle = xlNone

   Range("X4:X6").Select
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlNone
   End With

   Columns("AL:AN").ColumnWidth = 0#
   Range("AJ10:AK10").Select
   Selection.AutoFill Destination:=Range("AJ10:AK19"),
Type:=xlFillDefault
   Range("AJ10:AK19").Select
   Range("AG9").Select

   Range("W9").Select

GoTo ENDBYTREESIZE:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDBYTREESIZE:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

End Sub


Sub BackToOutput()
'
' BackToOutput Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'
   Range("W1").Select
   Sheets("OUTPUT PARAMETERS").Select
   Range("Y1").Select
End Sub


Sub BlanksOutDuplicates()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   Sheets("DATABASE").Select

   Range("DL38:DM20038").Select
   Selection.Sort Key1:=Range("DM38"),
Order1:=xlAscending, Header:=xlGuess _
      , OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

   'Blanks out duplicates (1st of a pair of duplicates).  For
A = 1 (trust me)
   Range("DM38").Select
   For A = 1 To 20000
         If Cells(A + 37, 117) = Cells(A + 38, 117) Then
         ActiveCell.FormulaR1C1 = ""
         ActiveCell.Offset(0, -1).Select
         ActiveCell.FormulaR1C1 = ""
         ActiveCell.Offset(0, 1).Select
         End If
         If Cells(A + 37 + 1, 117) = 0 Then
         ActiveCell.FormulaR1C1 = ""
         ActiveCell.Offset(0, -1).Select
         ActiveCell.FormulaR1C1 = ""
```

```
        ActiveCell.Offset(0, 1).Select
        A = 20000
        End If
        ActiveCell.Offset(1, 0).Select
    Next A

    'Sorts Random Number to get rid of blanks
    Range("DL38:DM20038").Select
    Selection.Sort Key1:=Range("DM38"),
Order1:=xlAscending, Header:=xlGuess _
      , OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

    'Sorts by Item Number to put in random order again
    Application.Calculate
      Range("DL38") = A
    For A = 1 To Range("DM36")
        ActiveCell = A
        ActiveCell.Offset(1, 0).Select
    Next A


GoTo ENDBLANKSDUPS:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDBLANKSDUPS:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub



Sub CHANGErecord()

' CHANGErecord Macro
' Macro recorded 8/22/2002 by Norman Pillsbury

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

If Range("X23") = "" Or Range("X23") > Range("O48")
Then
    RndMsg = MsgBox("You MUST include a valid Se-
quence Number.")
    Sheets("DATABASE").Select
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
Exit Sub
End If


If Range("X25") = "" And Range("Y25") = "" Or
Range("X25") = "" And Range("Y25") < 1 Then
    RndMsg = MsgBox("You MUST include a valid Spe-
cies Code.")
    Sheets("DATABASE").Select
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
Exit Sub
End If


If Range("X28") = "" And Range("Y28") = "" Or
Range("X28") < 1 And Range("Y28") < 1 Then
    RndMsg = MsgBox("You MUST include a valid Tree
Dbh.")
    Sheets("DATABASE").Select
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
Exit Sub
End If



Sheets("DATABASE").Select
If Range("CU23") <> "Sort Tree Seq Number" Then
    Application.Run "SortTREESEQNUMBER"
End If

    'Species Code
    If Range("Y25") <> "" Then
      Range("Y25").Select
      Selection.Copy

    '***Cell range must be from O53 (not O54) so that
changes can be made to Seq No. 1
      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
        LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
        MatchCase:=False).Activate
      ActiveCell.Offset(0, 2).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select   'Copies format from
```

```
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
      Application.CutCopyMode = False
   End If

   'Tree Dbh
   If Range("Y28") <> "" Then
      Range("Y28").Select
      Selection.Copy

      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate
      ActiveCell.Offset(0, 6).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select   'Copies format from
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
      Application.CutCopyMode = False
   End If

   'Tree Height
   If Range("Y29") <> "" Then
      Range("Y29").Select
      Selection.Copy

      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate
      ActiveCell.Offset(0, 7).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select   'Copies format from
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
      Application.CutCopyMode = False
   End If

   '====================================

   'User Defined Variables 1 through 10

   If Range("Y30") <> "" Then
      Range("Y30").Select
      Selection.Copy

      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate
      ActiveCell.Offset(0, 17).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select   'Copies format from
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
      Application.CutCopyMode = False
   End If

   If Range("Y31") <> "" Then
      Range("Y31").Select
      Selection.Copy

      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate
      ActiveCell.Offset(0, 18).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select   'Copies format from
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
      Application.CutCopyMode = False
   End If

   If Range("Y32") <> "" Then
      Range("Y32").Select
      Selection.Copy
```

```
      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate
      ActiveCell.Offset(0, 19).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select  'Copies format from
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
      Application.CutCopyMode = False
   End If

   If Range("Y33") <> "" Then
      Range("Y33").Select
      Selection.Copy

      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate
      ActiveCell.Offset(0, 20).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select  'Copies format from
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
      Application.CutCopyMode = False
   End If

   If Range("Y34") <> "" Then
      Range("Y34").Select
      Selection.Copy

      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate
      ActiveCell.Offset(0, 21).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select  'Copies format from
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
      Application.CutCopyMode = False
   End If

   If Range("Y35") <> "" Then
      Range("Y35").Select
      Selection.Copy

      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate
      ActiveCell.Offset(0, 22).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select  'Copies format from
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
      Application.CutCopyMode = False
   End If

   If Range("Y36") <> "" Then
      Range("Y36").Select
      Selection.Copy

      Range("O53:O50053").Select
      Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate
      ActiveCell.Offset(0, 23).Select
      ActiveSheet.Paste
      Application.CutCopyMode = False

      ActiveCell.Offset(1, 0).Select  'Copies format from
next cell down
      Selection.Copy
      ActiveCell.Offset(-1, 0).Select
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
         False, Transpose:=False
```

```
    Application.CutCopyMode = False
  End If

  If Range("Y37") <> "" Then
    Range("Y37").Select
    Selection.Copy

    Range("O53:O50053").Select
    Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
      LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
      MatchCase:=False).Activate
    ActiveCell.Offset(0, 24).Select
    ActiveSheet.Paste
    Application.CutCopyMode = False

    ActiveCell.Offset(1, 0).Select   'Copies format from
next cell down
    Selection.Copy
    ActiveCell.Offset(-1, 0).Select
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False
  End If

  If Range("Y38") <> "" Then
    Range("Y38").Select
    Selection.Copy

    Range("O53:O50053").Select
    Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
      LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
      MatchCase:=False).Activate
    ActiveCell.Offset(0, 25).Select
    ActiveSheet.Paste
    Application.CutCopyMode = False

    ActiveCell.Offset(1, 0).Select   'Copies format from
next cell down
    Selection.Copy
    ActiveCell.Offset(-1, 0).Select
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False
  End If

  If Range("Y39") <> "" Then
    Range("Y39").Select
    Selection.Copy

    Range("O53:O50053").Select
    Selection.Find(What:=Range("X23"),
After:=ActiveCell, LookIn:=xlValues, _
      LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
      MatchCase:=False).Activate
    ActiveCell.Offset(0, 26).Select
    ActiveSheet.Paste
    Application.CutCopyMode = False

    ActiveCell.Offset(1, 0).Select   'Copies format from
next cell down
    Selection.Copy
    ActiveCell.Offset(-1, 0).Select
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False
  End If

  '====================================

  Range("X23").Select
  Selection.ClearContents
  Range("Y25:Y39").Select
  Selection.ClearContents

  Range("CU23").Select
  ActiveCell.FormulaR1C1 = "Sort Tree Seq Number"

  GoTo ENDCHANGE:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err) & vbCr &
vbCr & "Check New Tree data carefully." & vbCr & "And
try again."

ENDCHANGE:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

  With Application
    .Calculation = xlAutomatic
    .MaxChange = 0.001
  End With

  Cells(8, 13) = ""
  Range("X23").Select
End Sub
```

```
Sub ClearChoicesA()
'
' ClearChoicesA Macro
' Macro recorded 11/1/2002 by Norman Pillsbury
'
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Q26:Q40").Select    'Clears Column Q -- Actual
No. of Trees Removed
   Selection.ClearContents

   Range("L26:L40").Select    'Clears Column L -- No of
Trees in Dbh class
   Selection.ClearContents
   Range("H22").Select

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub
```

```
Sub ClearChoicesB()
'
' ClearChoicesB Macro
' Macro recorded 11/1/2002 by Norman Pillsbury

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Sheets("REMOVAL").Select
   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False
   Range("Q26:Q40").Select    'Clears Column Q -- Actual
No. of Trees Removed
   Selection.ClearContents

   Range("L26:L40").Select    'Clears Column L -- No of
Trees in Dbh class
   Selection.ClearContents
   Range("H22").Select

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub
```

```
Sub ClearColQ()
'
' ClearColQ Macro
' Macro recorded 11/8/2002 by Norman Pillsbury
'
   Range("Q26:Q40").Select
   Selection.ClearContents
   Range("H23").Select
End Sub
```

```
Sub ClearCopy()

   Sheets("DATABASE").Select

   Application.Run "SortDBH"

   Sheets("DATABASE").Select
   Range("DM38:DM50038").Select
   Selection.Clear

   Range("DK38:DK30038").Select
   Selection.Copy
   Range("DL38:DL30038").Select
   Selection.PasteSpecial Paste:=xlValues
   Application.CutCopyMode = False

Range("O1").Select

End Sub
```

```
Sub ClearDM()

   Sheets("DATABASE").Select

   Range("DM38:DM50038").Select
   Selection.Clear

Range("O1").Select

End Sub
```

```
Sub ClearExcluded()
'
' ClearExcluded Macro
' Macro recorded 6/6/02 by NRM
'

'
   Range("AL15:AL64").Select
   Selection.ClearContents

   AllowFormattingCells = True
   Range("AL15:AL64").Interior.Color = RGB(255, 255,
153)
   AllowFormattingCells = False


   Range("AK4").Select
End Sub
```

```
Sub ClearIncluded()
'
' ClearIncluded Macro
' Macro recorded 6/6/02 by NRM
'
   Range("AP15:AP64").Select
   Selection.ClearContents

   AllowFormattingCells = True
   Range("AP15:AP64").Interior.Color = RGB(255, 255,
153)
   AllowFormattingCells = False


   Range("AK4").Select
End Sub
```

```
Sub ClearStandStock()
'
' ClearStandStock Macro
' Macro recorded 11/8/2002 by Norman Pillsbury
'
Sheets("MGT OPTIONS").Select

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Clears Stand Array
   Range("U34:BS154").Select   'Don't clear Col S
   Selection.ClearContents
   Range("S160:BT288").Select
   Selection.ClearContents
   Range("S20").Select

'Clears Stock Array
   Range("CB34:DZ154").Select
   Selection.ClearContents
   Range("BZ160:DZ288").Select
   Selection.ClearContents

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Range("R1").Select

End Sub
```

```
Sub ClearStatTable()
'
' ClearStatTable Macro
'
   Range("Q23:R41").Select
   Selection.ClearContents

   Range("V21:Z540").Select
   Selection.ClearContents
   Range("S13").Select
End Sub
```

```
Sub ClearStndStckOnOutputParametersSheet()
'
' ClearStndStckOnOutputParametersSheet Macro
```

```
‘
Sheets(“OUTPUT PARAMETERS”).Select

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Range(“BA15:CX64”).Select
   Selection.ClearContents
   Range(“BA78:CX127”).Select
   Selection.ClearContents

   Range(“BA14:CX14”).Select
   Selection.ClearContents
   Range(“BA77:CX77”).Select
   Selection.ClearContents
   Range(“BB5”).Select

   ‘Clears Removal Column on Database sheet
   Sheets(“DATABASE”).Select
   Range(“R54:R50053”).Select
   Selection.Clear

‘Formats Removal Column on Database sheet
   Range(“R54:R50053”).Select
   With Selection
      .HorizontalAlignment = xlCenter
   End With

   Range(“R52:R53”).Select
   Selection.Borders(xlBottom).LineStyle = xlContinuous

   Range(“R54:R50052”).Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   Selection.Borders(xlEdgeLeft).LineStyle = xlNone
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
   Selection.Borders(xlEdgeRight).LineStyle = xlNone
   Selection.Borders(xlInsideVertical).LineStyle = xlNone
   With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
   Range(“R54:R50052”).Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   Selection.Borders(xlInsideVertical).LineStyle = xlNone
   With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With

   Range(“R45”).Select

   Sheets(“OUTPUT PARAMETERS”).Select

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub CreateCurrentSpeciesList()
‘
‘ CreateCurrentSpeciesList Macro
‘ Macro recorded 7/11/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False
```

```
Range("P23:P531").Select
Selection.Clear

Range("Y22:AK76").Select
Selection.Clear

Range("X22").Select
Selection.Clear

Range("Q22:R531").Select
Selection.Clear
Range("S24").Select
Selection.AutoFill Destination:=Range("S24:S27"),
Type:=xlFillDefault
Range("S24:S27").Select
Range("P21").Select
Selection.AutoFill Destination:=Range("P21:P22"),
Type:=xlFillDefault
Range("P21:P22").Select
Range("P20").Select

    Sheets("SETUP").Select
    Range("B34:D540").Select
    Selection.Copy
    Sheets("VALUATION").Select
    Range("M25").Select
    ActiveSheet.Paste
    Range("N21").Select
    Columns("N:N").ColumnWidth = 19

    Sheets("SETUP").Select
    Range("I35:I84").Select
    Selection.Copy
    Sheets("VALUATION").Select
    Range("W27").Select
    ActiveSheet.Paste
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Selection.Interior.ColorIndex = xlNone
    Sheets("SETUP").Select
    Range("H35:H84").Select
    Selection.Copy
    Sheets("VALUATION").Select
    Range("X27").Select
    ActiveSheet.Paste
    Selection.Interior.ColorIndex = xlNone
    Range("U36").Select
    Sheets("SETUP").Select
    Range("I32:I33").Select
    Selection.Copy
    Sheets("VALUATION").Select
    Range("W25").Select
    ActiveSheet.Paste
    Sheets("SETUP").Select
    Range("H32:H33").Select
    Application.CutCopyMode = False
    Selection.Copy
    Sheets("VALUATION").Select
    Range("X25").Select
    ActiveSheet.Paste
    Application.CutCopyMode = False
    Range("W25:Y76").Select
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Range("U40").Select

    Columns("W:W").ColumnWidth = 7.5
    Columns("X:X").EntireColumn.AutoFit
    Range("W25:W76").Select
    Selection.Borders(xlLeft).LineStyle = xlContinuous

    Columns("M:M").ColumnWidth = 12.5
    Range("M25:N26").Select
    With Selection.Font
        .Name = "Geneva"
        .Size = 9
    End With
    Selection.Font.Bold = True
    Range("S23").Select

    Range("R10").Select
    ActiveCell.FormulaR1C1 = "=FALSE"

    Range("M25:M26").Select
    Selection.Interior.ColorIndex = 36
    Range("M23").Select

Range("X25:X26").Select
Selection.Borders(xlRight).LineStyle = xlContinuous

Range("X27:X76").Select
Selection.Borders(xlRight).LineStyle = xlContinuous

    Range("M25").Select
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("M26").Select
    Selection.Borders(xlRight).LineStyle = xlContinuous

    Range("AI77").Select
    Selection.Borders(xlTop).LineStyle = xlNone
    Columns("AD:AE").ColumnWidth = 0
    Columns("AC:AC").ColumnWidth = 0
    Range("M8").Select

    Range("R10") = False
    Range("X8") = False
    Range("X9") = False
    Range("X10") = False
    Range("X132") = 1
```

```
   'Colors cells white
   AllowFormattingCells = True
   Range("A5:B8").Interior.Color = RGB(255, 255, 255)
   AllowFormattingCells = False

Range("P20:R20").Select
Selection.AutoFill Destination:=Range("P20:R546"),
Type:=xlFillDefault
Range("Y20:AK20").Select
Selection.AutoFill Destination:=Range("Y20:AK86"),
Type:=xlFillDefault

Range("M23").Select

GoTo ENDSPLIST:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSPLIST:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub




Sub DefaultReset()
'
' DefaultReset Macro
'
'Variable Declaration
Dim OutPut As Integer
'Example of vbYesNo
OutPut = MsgBox("Continuing Will RESET the Pro-
gram to the Default State." & vbCr & "ALL Data Will Be
Deleted." & vbCr & "This Cannot Be Undone." & vbCr &
vbCr & "Do You Want To Continue?", vbYesNo, "YOU
ARE RESETTING THIS PROGRAM")
If OutPut = 7 Then    'Output = 7(No)
   Exit Sub
   Else
   'Output = 6(Yes)
End If

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("ID").Select
   Range("L12:L19,L22:L24,L27:L29").Select
   Range("L27").Activate
   Selection.ClearContents
   Range("S15").Activate
   Selection.ClearContents
   Range("P4").Select

Sheets("SETUP").Select
   Range("B36:D540").Select
   Selection.ClearContents
   Range("H54").Select
   Range("H54:H83").Select
   Selection.ClearContents
   Range("J54:T83").Select
   Selection.ClearContents
   Range("Y52").Select
   Range("B14:D14").Select
   Selection.ClearContents
   Range("B23").Select
   Selection.ClearContents

Sheets("DATABASE").Select
   Range("R25:R37").Select
   Selection.ClearContents

   Range("P28:Q28").Select
   ActiveCell.FormulaR1C1 = "=RC[-5]"
   Range("P28:Q28").Select
   Selection.AutoFill Destination:=Range("P28:Q37"),
Type:=xlFillDefault
   Range("P28:Q37").Select
   Range("S18").Select

   Range("X23,Y25,Y28:Y39").Select
   Range("Y28").Activate
   Selection.ClearContents
   Range("AC23:AD23").Select
   Selection.ClearContents
   Range("O54:AO50053").Select
   Selection.ClearContents
   Range("AG2").Select
   Range("I7").Select
   ActiveCell.FormulaR1C1 = "1"
   Application.Run ("LoadDBeqn")  'Loads eqns into S54,
T54 and W54.

Sheets("STATS").Select
   Application.Run ("ClearStatTable")

Sheets("OUTPUT PARAMETERS").Select
   Application.Run ("ResetMaxMin")
   Application.Run ("ClearExcluded")
   Application.Run ("ClearIncluded")

Sheets("REMOVAL").Select
   Range("AA5").Select
   ActiveCell.FormulaR1C1 = "FALSE"
```

```
   Range("I69").Select
   ActiveCell.FormulaR1C1 = "1"
   Application.Run ("GenerateNoTreesinDBHclassA")
   Range("Z5").Select
   ActiveCell.FormulaR1C1 = "FALSE"
   Application.Run ("ClearChoicesA")
   Application.Run ("ClearChoicesB")
   Range("AD26:AD40").Select
   Selection.ClearContents
   Application.Run ("GenerateNoTreesinDBHclassB")

Sheets("MGT OPTIONS").Select
   Application.Run ("ClearStandStock")

Sheets("VALUATION").Select
   Application.Run ("VIIIaMethodOne")    'Clears
Method 1
   Range("R10").Select
   ActiveCell.FormulaR1C1 = "FALSE"
   Range("AU129").Select
   ActiveCell.FormulaR1C1 = "1"
   Range("AY126").Select
   ActiveCell.FormulaR1C1 = "1"
   Range("X8").Select
   ActiveCell.FormulaR1C1 = "FALSE"
   Range("X9").Select
   ActiveCell.FormulaR1C1 = "FALSE"
   Range("X10").Select
   ActiveCell.FormulaR1C1 = "FALSE"

Sheets("ID").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

MsgBox ("          Done..." & vbCr & "Program Has Been
Reset")

End Sub




Sub DELETErecord()

' DELETErecord Macro
' Macro recorded 8/22/2002 by Norman Pillsbury

On Error GoTo Errhandler

 If Cells(23, 31) = "Error--Not in List!!" Then
   Cells(23, 29) = ""
   Range("AC23").Select
   Exit Sub
End If

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

If Range("AC23") = "" Or Range("AC23") < 1 Then
   RndMsg = MsgBox("You MUST include a valid Se-
quence Number.")
   Sheets("DATABASE").Select
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
Exit Sub
End If

Sheets("DATABASE").Select
If Range("CU23") <> "Sort Tree Seq Number" Then
   Application.Run "SortTREESEQNUMBER"
End If


   '***Cell range must be from O53 (not O54) so that
changes can be made to Seq No. 1
   'Seq Number; Range must be O to O.
   If Range("AE23") <> "" Then
      Range("O53:O50053").Select
      Selection.Find(What:=Range("AC23"),
After:=ActiveCell, LookIn:=xlValues, _
         LookAt:=xlPart, SearchOrder:=xlByColumns,
SearchDirection:=xlNext, _
         MatchCase:=False).Activate

      'Tree Seq No.
      ActiveCell.Offset(0, 0).Select
      Selection.ClearContents

      'Tree Record
      ActiveCell.Offset(0, 1).Select
      Selection.ClearContents

      ActiveCell.Offset(0, 1).Select
      Selection.ClearContents

      ActiveCell.Offset(0, 1).Select
      Selection.ClearContents

      ActiveCell.Offset(0, 1).Select
      Selection.ClearContents

      ActiveCell.Offset(0, 1).Select
      Selection.ClearContents

      ActiveCell.Offset(0, 1).Select
      Selection.ClearContents
```

```
'ActiveCell.Offset(0, 3).Select
'Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

'Delete User Defined Variables 1-10
ActiveCell.Offset(0, 2).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
Selection.ClearContents

ActiveCell.Offset(0, 1).Select
```

```
    Selection.ClearContents

    ActiveCell.Offset(0, 1).Select
    Selection.ClearContents

End If
  Range("CU23").Select
  ActiveCell.FormulaR1C1 = "Sort Tree Seq Number"

  Range("AC23").Select
  Selection.ClearContents

GoTo ENDDELETE:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err) & vbCr &
vbCr & "Check New Tree data carefully." & vbCr & "And
try again."

ENDDELETE:
Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub
```

```
Sub DeleteSpecies()
'
' DeleteSpecies Macro
' Macro recorded 6/5/02 by NRM

Sheets("SETUP").Select

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

If Cells(23, 3) <> "" Then

For A = 1 To 500
  If Cells(A + 35, 2) = Cells(23, 2) Then
    Cells(A + 35, 2) = ""
    Cells(A + 35, 3) = ""
    Cells(A + 35, 4) = ""
  End If
   If Cells(23, 3) = "" Then
    A = 500
   End If
Next A
```

```
      Range("B36:D540").Select
      Selection.Sort Key1:=Range("B36"),
Order1:=xlAscending, Key2:=Range("C36" _
         ), Order2:=xlAscending, Key3:=Range("D36"),
Order3:=xlAscending, Header _
         :=xlNo, OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

      Range("B23").Select
      Selection.ClearContents
      Range("B23").Select

Else
   CodeMsg = MsgBox("Sorry, that Species Code doesn't
exist.")
End If

   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic

End Sub




Sub DELETEupdaterecord()


' DELETEupdaterecord Macro
' Macro recorded 8/22/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select
If Range("CU23") <> "Sort Tree Seq Number" Then
   Application.Run "SortTREESEQNUMBER"
End If

Range("O54:AO50053").Select
   Selection.Sort Key1:=Range("O54"),
Order1:=xlAscending, Header:=xlNo, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

   Range("N54:N50053").Select
   Worksheets("DATABASE").Columns(14).Calculate

   Selection.Copy
   Range("O54").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Application.CutCopyMode = False

Range("CU23").Select
ActiveCell.FormulaR1C1 = "Sort Tree Seq Number"

GoTo ENDUP:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDUP:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
  With Application
     .Calculation = xlAutomatic
     .MaxChange = 0.001
   End With

   Beep

End Sub



Sub DiaCheck2()

If Range("AB82") = 1 Then
      'Range("AB82") = 1
      'Range("AC82") = 1
      'RndMsg = MsgBox("Set minimum diameter first.")
Else

   If Range("AC82") < Range("AB82") Then
    Range("AB82") = 1
    Range("AC82") = 1
    RndMsg = MsgBox("Max Diameter must be larger
than minimum diameter.")
   End If
End If


End Sub



Sub DiaCheck2()

If Range("AB82") = 1 Then
      'Range("AB82") = 1
```

```
    'Range("AC82") = 1
    'RndMsg = MsgBox("Set minimum diameter first.")
Else

  If Range("AC82") < Range("AB82") Then
   Range("AB82") = 1
   Range("AC82") = 1
   RndMsg = MsgBox("Max Diameter must be larger
than minimum diameter.")
  End If
End If



End Sub




Sub DoIt()

  Application.ScreenUpdating = True
  'FOR FORMATTING DATABASE ON DATABASE
PAGE.  Also used when RESETTING the program on ID
sheet.
  With Sheet2.Shapes("Rectangle 22")
    .Visible = msoTrue = (Not Sheet2.Shapes("Rectangle
22").Visible)
    '.Visible = msoTrue = (Not .Visible)
  End With
  'Forces TextBox to show while code is running
  'Toggling sheets Forces Rectangle 1 to show while code
is running
  Sheet8.Select
  'Application.Wait (Now + #12:00:01 AM#)
  Sheet2.Select
  Application.Wait (Now + #12:00:01 AM#)
  End Sub




Sub DoItBRANCHES()

  'Sheet10 is the ID sheet
  'Sheet8 is a blank sheet added to the end and is called
"Sheet1"
  Application.ScreenUpdating = True
  With Sheet2.Shapes("Rectangle 28")
    .Visible = msoTrue = (Not Sheet2.Shapes("Rectangle
28").Visible)
  End With
  'Forces TextBox to show while code is running
  'Toggling sheets Forces Rectangle 1 to show while code
is running
  Sheet8.Select
  Sheet2.Select
  Application.Wait (Now + #12:00:01 AM#)
  End Sub



Sub DoItEXPORT()

  'Sheet10 is the ID sheet
  'Sheet8 is a blank sheet added to the end and is called
"Sheet1"
  Application.ScreenUpdating = True
  With Sheet10.Shapes("Rectangle 12")
    .Visible = msoTrue = (Not Sheet10.
Shapes("Rectangle 12").Visible)
  End With
  'Forces TextBox to show while code is running
  'Toggling sheets Forces Rectangle 1 to show while code
is running
  Sheet8.Select
  Sheet10.Select
  Application.Wait (Now + #12:00:01 AM#)
  End Sub




Sub DoItFormat()

  Application.ScreenUpdating = True
  'FOR FORMATTING DATABASE ON DATABASE
PAGE.  Also used when RESETTING the program on ID
sheet.
  With Sheet2.Shapes("Rectangle 26")
    .Visible = msoTrue = (Not Sheet2.Shapes("Rectangle
26").Visible)
  End With
  'Forces TextBox to show while code is running
  'Toggling sheets Forces Rectangle 1 to show while code
is running
  Sheet8.Select
  Sheet2.Select
  Application.Wait (Now + #12:00:01 AM#)
  End Sub



Sub DoItIMPORT()

  Application.ScreenUpdating = True
  'Sheets("ID").Select
  With Sheet10.Shapes("Rectangle 11")
    .Visible = msoTrue = (Not Sheet10.
Shapes("Rectangle 11").Visible)
    '.Visible = msoTrue = (Not .Visible)
  End With
  'Forces TextBox to show while code is running
```

```
   'Toggling sheets Forces Rectangle 1 to show while code
is running
   Sheet8.Select
   'Application.Wait (Now + #12:00:01 AM#)
   Sheet10.Select
   Application.Wait (Now + #12:00:01 AM#)
   End Sub



   Sub DoItEXPORT()

   'Sheet10 is the ID sheet
   'Sheet8 is a blank sheet added to the end and is called
"Sheet1"
   Application.ScreenUpdating = True
   With Sheet10.Shapes("Rectangle 12")
      .Visible = msoTrue = (Not Sheet10.
Shapes("Rectangle 12").Visible)
   End With
   'Forces TextBox to show while code is running
   'Toggling sheets Forces Rectangle 1 to show while code
is running
   Sheet8.Select
   Sheet10.Select
   Application.Wait (Now + #12:00:01 AM#)
   End Sub




Sub DoItRandom()

   Application.ScreenUpdating = True
   'Sheets("DATABASE").Select
   With Sheet5.Shapes("Rectangle 8")
      .Visible = msoTrue = (Not Sheet5.Shapes("Rectangle
8").Visible)
      '.Visible = msoTrue = (Not .Visible)
   End With
   'Forces TextBox to show while code is running
   'Toggling sheets Forces Rectangle 1 to show while code
is running
   Sheet8.Select
   'Application.Wait (Now + #12:00:01 AM#)
   Sheet5.Select
   Application.Wait (Now + #12:00:01 AM#)
   End Sub



Sub Export()


On Error GoTo Errhandler

'Variable Declaration
Dim OutPut As Integer
'Example of vbYesNo
OutPut = MsgBox("The Backup File Will Have a New
Name." & vbCr & vbCr & "Do You Want To Continue?",
vbYesNo, "YOU ARE EXPORTING CURRENT DATA
AS A BACKUP FILE")
If OutPut = 7 Then    'Output = 7(No)
   Exit Sub
   Else
   'Output = 6(Yes)
End If

Application.Run "DoItEXPORT"

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("AC6").Select
   Selection.Copy
   Range("T6").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   Dim Template As String
   Dim NewCUFIMName As String  'What the backup
file will be named at end of program
   Dim CUFIMProgram As String    'Name of main
program file
   Dim CurrentDir As String
   Template = "Export Template.xlsx"    'Name of Tem-
plate that becomes the NewCUFIMName
   NewCUFIMName = Range("S28") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsx"
   CUFIMProgram = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim DBBackup As String
   DBBackup = Range("S26")
   Dim ProgBackup As String
```

```
   ProgBackup = Range("S17")

   'BaseDir ==> Points to Desktop where CUFIM folder
resides
   'BaseDir & CUFIMFolder==> Points to inside of CU-
FIM folder
   'BaseDir & CUFIMFolder & DBBackup==> Points to
inside Database Backup Folder
   'BaseDir & CUFIMFolder & ProgBackup==> Points to
inside Program Backup Folder

   'Sheets("ID").Select
   CurrentDir = (BaseDir & CUFIMFolder)
   ChDir CurrentDir
   Workbooks.Open Template, ReadOnly:=True

   'Exports Name of Dataset
   Windows(CUFIMProgram).Activate
   Sheets("ID").Select
   Range("S15").Select
   Selection.Copy
   Windows("Export Template.xlsx").Activate
   Range("C1").Select
   ActiveSheet.Paste
   Range("A1").Select

   'Exports Species Code, Species List and Species Group
   Windows(CUFIMProgram).Activate
   Range("S30").Select
   Sheets("SETUP").Select
   Range("B36:D540").Select
   Selection.Copy
   Windows("Export Template.xlsx").Activate
   Range("A5").Select
   ActiveSheet.Paste
   Range("D9").Select
   Windows(CUFIMProgram).Activate
   Application.CutCopyMode = False
   Range("G12").Select

   'Exports Seq No., Tree Record No., Species Code
   Windows(CUFIMProgram).Activate
   Sheets("DATABASE").Select
   Range("O54:Q50053").Select
   Application.CutCopyMode = False
   Selection.Copy
   Windows("Export Template.xlsx").Activate
   Sheets("DATABASE").Select
   Range("D5").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:=False _
      , Transpose:=False
   'ActiveSheet.Paste

   'Exports Species Name
   Windows(CUFIMProgram).Activate
   Sheets("DATABASE").Select
   Range("S54:S50053").Select
   Application.CutCopyMode = False
   Selection.Copy
   Windows("Export Template.xlsx").Activate
   Sheets("DATABASE").Select
   Range("G5").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:=False _
      , Transpose:=False

   'Exports Dbh and Height
   Windows(CUFIMProgram).Activate
   Range("U54:V50053").Select
   Application.CutCopyMode = False
   Selection.Copy
   Windows("Export Template.xlsx").Activate
   Sheets("DATABASE").Select
   Range("H5").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False

   'Exports Variables 1-10
   Windows(CUFIMProgram).Activate
   Range("AF54:AO50053").Select
   Application.CutCopyMode = False
   Selection.Copy
   Windows("Export Template.xlsx").Activate
   Sheets("DATABASE").Select
   Range("K5").Select
   ActiveSheet.Paste

   Windows(CUFIMProgram).Activate
   Range("M28:M37").Select
   Selection.Copy
   Application.CutCopyMode = False
   Windows("Export Template.xlsx").Activate
   Range("K4").Select

Windows(CUFIMProgram).Activate
   Range("M28:M37").Select
   Selection.Copy
   Windows("Export Template.xlsx").Activate
   Range("K4").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:=False _
      , Transpose:=True
   Range("K2").Select
   Columns("K:K").EntireColumn.AutoFit
   Columns("L:L").EntireColumn.AutoFit
   Columns("M:M").EntireColumn.AutoFit
   Columns("N:N").EntireColumn.AutoFit
   Columns("O:O").EntireColumn.AutoFit
   Columns("P:P").EntireColumn.AutoFit
   Columns("Q:Q").EntireColumn.AutoFit
   Columns("R:R").EntireColumn.AutoFit
   Columns("S:S").EntireColumn.AutoFit
   Columns("T:T").EntireColumn.AutoFit
```

```
    Range("K1").Select
    Application.CutCopyMode = False
   Range("A1").Select

    'SAVE AS with correct name
    'BaseDir ==> Points to Desktop where CUFIM folder
resides
    'BaseDir & CUFIMFolder==> Points to inside of CU-
FIM folder
    'BaseDir & CUFIMFolder & DBBackup==> Points to
inside Database Backup Folder
    'BaseDir & CUFIMFolder & ProgBackup==> Points to
inside Program Backup Folder

    Windows(CUFIMProgram).Activate
    Sheets("ID").Select
    NewCUFIMName = Range("S28") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsx"
    Windows("Export Template.xlsx").Activate
    Sheets("DATABASE").Select

    ChDir (BaseDir & CUFIMFolder & DBBackup)
    ActiveWorkbook.SaveAs NewCUFIMName
    ActiveWorkbook.Close

    Windows(CUFIMProgram).Activate
    Sheets("ID").Select
    Range("A1").Select


GoTo ENDREMOVAL:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDREMOVAL:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Application.Run "DoItEXPORT"

Beep

End Sub




Sub FormatDatabase()
On Error GoTo Errhandler
'
' Formats all cells in database
'------------------
'Application.StatusBar = "Please wait"

'then place this at the end of your code
'Application.StatusBar = ""

'------------------

Application.Run "DoIt"

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

    Range("R25").Select

    Range("O54:R50053").Select
    With Selection
      .HorizontalAlignment = xlCenter
      .VerticalAlignment = xlBottom
    End With
    Selection.NumberFormat = "0"
    Range("S54:S50053").Select
    With Selection
      .HorizontalAlignment = xlLeft
      .VerticalAlignment = xlBottom
    End With
    Range("T54:U50053").Select
    With Selection
      .HorizontalAlignment = xlCenter
    End With
    Selection.NumberFormat = "0"
    Range("U54:V50053").Select
    Selection.NumberFormat = "0.0"
    With Selection
      .HorizontalAlignment = xlCenter
    End With
    Range("W54:X54").Select
    Selection.Copy
    Range("W55:X50053").Select
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False
    Range("Y54").Select
    Selection.Copy
    Range("Y55:Y50053").Select
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False
    Range("Z54:AC54").Select
    Selection.Copy
    Range("Z55:AC50053").Select
    Selection.PasteSpecial Paste:=xlFormats,
```

```
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("AF54:AO54").Select
   Application.CutCopyMode = False
   Selection.Copy
   Range("AF55:AO50053").Select
   Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Application.CutCopyMode = False
   Range("O55:AO50053").Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With

   Range("AE52:AE50055").Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   Selection.Borders(xlEdgeTop).LineStyle = xlNone
   Selection.Borders(xlEdgeBottom).LineStyle = xlNone
   Selection.Borders(xlInsideVertical).LineStyle = xlNone
   Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone
   Range("O50054:AD50054").Select
   Selection.Borders(xlTop).LineStyle = xlContinuous
   Range("AF50054:AO50054").Select
   Selection.Borders(xlTop).LineStyle = xlContinuous

   Sheets("DATABASE").Select
   Range("O54:AO50053").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With

      AllowFormattingCells = True
      Range("O54:AO50053").Interior.Color = RGB(255,
255, 255)
      AllowFormattingCells = False

   Sheets("SETUP").Select
   Range("B36:F540").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
       AllowFormattingCells = True
      Range("B36:F540").Interior.Color = RGB(255, 255,
255)
      AllowFormattingCells = False
   End With
   Range("B33").Select
   Sheets("DATABASE").Select

GoTo ENDFMTDB:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDFMTDB:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Application.Run "DoIt"
Beep

   Range("R25").Select
   End Sub



Sub GenerateNoTreesinDBHclassA()

On Error GoTo Errhandler

'Clears Stand Arrays BOX A and Actual No. Col (Brown
column)
   Range("L26:L40").Select
   Selection.ClearContents

   Range("L20").Select
'==========

Dim calcMode, updateMode
```

```
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("REMOVAL").Select
Dim StandRandom#(15, 1)     'Data Matrix for Stand
Table [8 dbh classes, 40 groups]
Dim B#(15, 1)               'Array for upper dbh class values

B(1, 1) = Range("K26")   'Upper diameter for dbh class 1
B(2, 1) = Range("K27")   'Upper diameter for dbh class 2
B(3, 1) = Range("K28")   'Upper diameter for dbh class 3
B(4, 1) = Range("K29")   'Upper diameter for dbh class 4
B(5, 1) = Range("K30")   'Upper diameter for dbh class 5
B(6, 1) = Range("K31")   'Upper diameter for dbh class 6
B(7, 1) = Range("K32")   'Upper diameter for dbh class 7
B(8, 1) = Range("K33")   'Upper diameter for dbh class 8
B(9, 1) = Range("K34")   'Upper diameter for dbh class 9
B(10, 1) = Range("K35")   'Upper diameter for dbh class
10
B(11, 1) = Range("K36")   'Upper diameter for dbh class
11
B(12, 1) = Range("K37")   'Upper diameter for dbh class
12
B(13, 1) = Range("K38")   'Upper diameter for dbh class
13
B(14, 1) = Range("K39")   'Upper diameter for dbh class
14
B(15, 1) = Range("K40")   'Upper diameter for dbh class
15

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("DATABASE").Select
'Application.Run "SortDBH"

'====================================
   'For A = ;  searches total database
   'Dim B array is number of dbh classes
   'For C =;   is number of dbh classes
   '54 Selects first possible tree in database
   'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                    'in C loop dbh class, then....
'====================================

   For A = 54 To MaxNo + 54
      If Cells(A, 15) <> "" Then          'A tree exists in the
database
         For C = 1 To 15
            If Cells(A, 21) < B(C, 1) Then
               StandRandom(C, 1) = StandRandom(C, 1) +
1: C = 15
            End If
         Next C
      End If
   Next A

Sheets("REMOVAL").Select

'===========

   'Outputs Stand Table data BOX A
   Range("L26").Select
   For C = 1 To 15
      Range(Cells(C + 25, 12), Cells(C + 25, 12)).Select
      If StandRandom(C, 1) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = StandRandom(C, 1)
      End If
   Next C
'===========
Range("L20").Select

GoTo ENDREMOVAL:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDREMOVAL:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub GenerateNoTreesinDBHclassB()

On Error GoTo Errhandler

'This sub must be built into Random sub.  Also is used
with "Click to Update this column", in Col AD.
'Clears Stand Arrays BOX B
   Range("AC26:AC40").Select
   Selection.ClearContents
   Range("AB22").Select
'===========

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("REMOVAL").Select
```

```
Dim StandRandom#(15, 1)     'Data Matrix for Stand
Table [8 dbh classes, 40 groups]
Dim B#(15, 1)               'Array for upper dbh class values

B(1, 1) = Range("K26")   'Upper diameter for dbh class 1
B(2, 1) = Range("K27")   'Upper diameter for dbh class 2
B(3, 1) = Range("K28")   'Upper diameter for dbh class 3
B(4, 1) = Range("K29")   'Upper diameter for dbh class 4
B(5, 1) = Range("K30")   'Upper diameter for dbh class 5
B(6, 1) = Range("K31")   'Upper diameter for dbh class 6
B(7, 1) = Range("K32")   'Upper diameter for dbh class 7
B(8, 1) = Range("K33")   'Upper diameter for dbh class 8
B(9, 1) = Range("K34")   'Upper diameter for dbh class 9
B(10, 1) = Range("K35")   'Upper diameter for dbh class
10
B(11, 1) = Range("K36")   'Upper diameter for dbh class
11
B(12, 1) = Range("K37")   'Upper diameter for dbh class
12
B(13, 1) = Range("K38")   'Upper diameter for dbh class
13
B(14, 1) = Range("K39")   'Upper diameter for dbh class
14
B(15, 1) = Range("K40")   'Upper diameter for dbh class
15

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("DATABASE").Select
'Application.Run "SortDBH"

'====================================
   'For A = ;  searches total database
   'Dim B array is number of dbh classes
   'For C =;   is number of dbh classes
   '54 Selects first possible tree in database
   'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                    'in C loop dbh class, then....
'====================================

   For A = 54 To MaxNo + 54
      If Cells(A, 15) <> "" Then            'A tree exists in the
database
         For C = 1 To 15
            If Cells(A, 21) < B(C, 1) Then
               StandRandom(C, 1) = StandRandom(C, 1) +
1: C = 15
            End If
         Next C
      End If
   Next A

Sheets("REMOVAL").Select
'==========
   'Outputs Stand Table data BOX B
   Range(" AC26").Select
   For C = 1 To 15
      Range(Cells(C + 25, 29), Cells(C + 25, 29)).Select
      If StandRandom(C, 1) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = StandRandom(C, 1)
      End If
   Next C
'==========

Range("AB22").Select

GoTo ENDNOTREESCLASSB:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDNOTREESCLASSB:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub

Sub GeneratePercentGraphBoxB()

' GeneratePercentGraphBoxB Macro
' Macro recorded 6/9/02 by Norm

On Error GoTo Errhandler
'
'This macro not used.  Graph for part B is done directly
from column 5

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   Range("Z26:Z40").Select
   Selection.Copy
   Range("CO26").Select

   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
```

```
   Application.CutCopyMode = False

Range("CO26").Select

For A = 1 To 15
   If Not IsNumeric(Cells(A + 25, 93)) Then
      ActiveCell.Select
      Selection.ClearContents
   End If
   ActiveCell.Offset(1, 0).Select
Next A
Range("AB22").Select

GoTo ENDPERCENTGRAPHB:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDPERCENTGRAPHB:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub




Sub GeneratePercentGraphBoxA()
'
' GeneratePercentGraphBoxA Macro
' Macro recorded 6/9/02 by NRM

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   Sheets("STATS").Select
   Application.Run "StatsStandTable"
   Sheets("REMOVAL").Select

   Range("I26:I40").Select
   Selection.Copy
   Range("CM26").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Application.CutCopyMode = False

   Range("N26:N40").Select
   Selection.Copy
   Range("CL26").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Application.CutCopyMode = False

   Range("Q26:Q40").Select
   Selection.Copy
   Range("CN26").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Application.CutCopyMode = False

Range("CL26").Select
For A = 26 To 40
   If Not IsNumeric(Cells(A, 89)) Then
      ActiveCell.Select
      Selection.ClearContents
   End If
   ActiveCell.Offset(1, 0).Select
Next A

Range("CM26").Select
For A = 26 To 40
   If Not IsNumeric(Cells(A, 91)) Then
      ActiveCell.Select
      Selection.ClearContents
   End If
   ActiveCell.Offset(1, 0).Select
Next A

Range("CN26").Select
For B = 26 To 40
   If Not IsNumeric(Cells(B, 92)) Then
      ActiveCell.Select
      Selection.ClearContents
   End If
   ActiveCell.Offset(1, 0).Select
Next B

Application.Run "GenerateNoTreesinDBHclassA"

Range("K22").Select

GoTo ENDPERCENTGRAPHA

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDPERCENTGRAPHA:

Application.ScreenUpdating = updateMode
```

```
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub



Sub GenerateSortedStockTableB1()

On Error GoTo Errhandler

Sheets("MGT OPTIONS").Select
If Range("S29") = 0 Then
   RndMsg = MsgBox("First you must Generate the Stock
Table (A1 or A2).")
   Exit Sub
End If

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Sheets("VALUATION").Select
   Range("M24:P531").Select
   Selection.Clear

   Range("Q24:W80").Select
   Selection.Clear
   Range("M19").Select

   Sheets("MGT OPTIONS").Select
   Range("CB22").Select

Application.Run "GenerateStockTableA1"

   Range("BZ26:DZ154").Select
   Selection.Copy
   Range("BZ160").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Application.CutCopyMode = False

   Range("CC161").Select
   Range("CC163:DZ288").Select
   Selection.Sort Key1:=Range("CC163"),
Order1:=xlDescending, Header:=xlGuess _
      , OrderCustom:=1, MatchCase:=False,
Orientation:=xlLeftToRight
   Range("CC166:EA166").Select
   With Selection
      .HorizontalAlignment = xlLeft
   End With


GoTo ENDSTKB1:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSTKB1:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Range("CC164").Select
End Sub



Sub GenerateSortedStockTableB2()

On Error GoTo Errhandler

Sheets("MGT OPTIONS").Select
If Range("BZ29") = 0 Then
   RndMsg = MsgBox("First you must Generate the Stock
Table (A1 or A2).")
   Exit Sub
End If

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Sheets("VALUATION").Select
   Range("M24:P531").Select
   Selection.Clear

   Range("Q24:W80").Select
   Selection.Clear
   Range("M19").Select

   Sheets("MGT OPTIONS").Select
   Range("CB22").Select

Application.Run "GenerateStockTableA2"
```

```
   Range("BZ26:DZ154").Select
   Selection.Copy
   Range("BZ160").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Application.CutCopyMode = False

   Range("CC161").Select
   Range("CC163:DZ288").Select
   Selection.Sort Key1:=Range("CC163"),
Order1:=xlDescending, Header:=xlGuess _
      , OrderCustom:=1, MatchCase:=False,
Orientation:=xlLeftToRight

   Range("CC166:EA166").Select
   With Selection
      .HorizontalAlignment = xlLeft
   End With


GoTo ENDSTKB2:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSTKB2:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Range("CC164").Select
End Sub




Sub GenerateStockTableA1()
'
' GenerateStockTableA1 Macro
' Macro recorded 7/11/2002 by Norman Pillsbury

On Error GoTo Errhandler
'
Sheets("MGT OPTIONS").Select
If Range("S29") = 0 Then
   RndMsg = MsgBox("First you must Generate the
Stand Table (A1 or A2).")
   Exit Sub
End If

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Dim Stand#(120, 50)      'Data Matrix for Stock Table [8
dbh classes, 40 groups]
Dim B(120, 2)            'Array for upper dbh class values

For A = 1 To 120
   If Cells(34, 137) > 0 Then
      B(A, 1) = Cells(33 + A, 139)       'Upper dia for dbh
classes
      B(A, 2) = Cells(33 + A, 137)       'dbh class
   Else
      A = 120
   End If
Next A

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("database").Select
'Application.Run "SortDBH"

'===========
   'For A = ;  searches total database
   'B array is number of dbh classes
   'For C =;   is number of dbh classes
   'For D =;   is number of species groups
   'R54 Selects first possible removal tree in Removal Col.
   'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                        'in C loop dbh class, then....
'===========

   For A = 54 To MaxNo + 54
      If Cells(A, 18) <> "" Then            'Checks to see if
Removal Code means tree to be removed.
      For C = 1 To 120
         If Cells(A, 21) < B(C, 1) Then     'Checks to see if
dbh is < upper limit of 1st dbh class; then 2nd, etc.
         For D = 1 To 50
            If Cells(A, 20) = D Then        'Checks  to see
which Species Group tree is in.
               Stand(C, D) = Stand(C, D) + Cells(A, 23): C
= 120: D = 50    'Adds tree volume to each species group
dbh class
            End If
         Next D
         End If
      Next C
      End If
   Next A
```

```
Sheets("MGT OPTIONS").Select

'Clears Stock Array
   Range("CB34:DZ154").Select
   Selection.ClearContents
   Range("BZ160:DZ288").Select
   Selection.ClearContents

'==========
   'Outputs Stand Table data
   Range("CC34").Select
   For C = 1 To 120
      Range(Cells(C + 33, 80), Cells(C + 33, 80)).Select
      ActiveCell.FormulaR1C1 = B(C, 2)
      For D = 1 To 50
      Range(Cells(C + 33, D + 80), Cells(C + 33, D + 80)).
Select
      If Stand(C, D) > 0 Then
         'MsgBox (Stand(C, D))
         ActiveCell.FormulaR1C1 = Stand(C, D)
      End If
      Next D
      If Cells(C + 33, 80) = "" Then
         D = 50: C = 120
      End If
   Next C
'==========

Range("BZ26").Select
ActiveCell.FormulaR1C1 = "Volume of Trees to be Re-
moved (by Dbh Class and by Species Group)"

GoTo ENDSTKA1:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSTKA1:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub




Sub GenerateStockTableA2()
'
' GenerateStockTableA2 Macro
' Macro recorded 7/11/2002 by Norman Pillsbury
'
Sheets("MGT OPTIONS").Select
If Range("S29") = 0 Then
   RndMsg = MsgBox("First you must Generate the
Stand Table (A1 or A2).")
   Exit Sub
End If

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Dim Stand#(120, 50)      'Data Matrix for Stand Table [8
dbh classes, 40 groups]
Dim B(120, 2)            'Array for upper dbh class values

For A = 1 To 120
   If Cells(34, 137) > 0 Then
      B(A, 1) = Cells(33 + A, 139)      'Upper dia for dbh
classes
      B(A, 2) = Cells(33 + A, 137)      'dbh class
   Else
      A = 120
   End If
Next A

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("database").Select
'Application.Run "SortDBH"

'==========
   'For A = ;  searches total database
   'B array is number of dbh classes
   'For C =;   is number of dbh classes
   'For D =;   is number of species groups
   'R54 Selects first possible removal tree in Removal Col.
   'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                        'in C loop dbh class, then....
'==========

   For A = 54 To MaxNo + 54
      For C = 1 To 120
         If Cells(A, 21) < B(C, 1) Then     'Checks to see if
dbh is < upper limit of 1st dbh class; then 2nd, etc.
         For D = 1 To 50
            If Cells(A, 20) = D Then          'Checks  to see
which Species Group tree is in.
               Stand(C, D) = Stand(C, D) + Cells(A, 23): C
= 120: D = 50    'Adds tree volume to each species group
dbh class
            End If
```

```
        Next D
        End If
      Next C
   Next A

Sheets("MGT OPTIONS").Select

'Clears Stand Array
   Range("CB34:DZ154").Select
   Selection.ClearContents
   Range("BZ160:DZ288").Select
   Selection.ClearContents

'==========
   'Outputs Stand Table data
   Range("CC34").Select
   For C = 1 To 120
      Range(Cells(C + 33, 80), Cells(C + 33, 80)).Select
      ActiveCell.FormulaR1C1 = B(C, 2)
      For D = 1 To 50
      Range(Cells(C + 33, D + 80), Cells(C + 33, D + 80)).
Select
      If Stand(C, D) > 0 Then
         'MsgBox (Stand(C, D))
         ActiveCell.FormulaR1C1 = Stand(C, D)
      End If
      Next D
      If Cells(C + 33, 80) = "" Then
         D = 50: C = 120
      End If
   Next C
'==========
Range("BZ26").Select
ActiveCell.FormulaR1C1 = "Volume of Trees in Inven-
tory (by Dbh Class and by Species Group)"


GoTo ENDSTKA2:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSTKA2:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub
```

```
' GoToDatabaseSheet Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'
   Range("A1").Select
   Sheets("DATABASE").Select
   Range("N1").Select
End Sub
```

```
Sub GoToStatsSheet()
'
' GoToStatsSheet Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'
   Range("A1").Select
   Sheets("STATS").Select
   Range("A1").Select
End Sub
```

```
Sub GoToOutputSheet()
'
' GoToOutputSheet Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'

   Range("A1").Select
   Sheets("OUTPUT PARAMETERS").Select
   Range("A1").Select
End Sub
```

```
Sub GoToRemovalSheet()
'
' GoToRemovalSheet Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'
   Range("A1").Select
   Sheets("REMOVAL").Select
   Range("Y1").Select
End Sub
```

```
Sub GoToDatabaseSheet()
'
```

```
Sub GoToMgtOptionsSheet()
'
```

```
' GoToMgtOptionsSheet Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'
   Range("A1").Select
   Sheets("MGT OPTIONS").Select
   Range("R1").Select
End Sub


Sub BackToOutput()
'
' BackToOutput Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'
   Range("W1").Select
   Sheets("OUTPUT PARAMETERS").Select
   Range("Y1").Select
End Sub


Sub GoToValuation()
'
' GoToValuation Macro
' Macro recorded 7/11/2002 by Norman Pillsbury

   Range("A1").Select
   Sheets("VALUATION").Select
   Range("A1").Select
End Sub


Sub GoToIDsheet()
'
' GoToIDsheet Macro
'

'
   Sheets("ID").Select
   Range("A1").Select
End Sub


Sub GoToImportExport()
'
With Application
   With ActiveWindow
         .Top = 1
         .Left = 1
         ' .Height = 1900
         '  .Width = 1640
        .Height = Application.UsableHeight - 50
        .Width = Application.UsableWidth - 50
      End With

         '.WindowState = xlMaximized
   End With

   Range("A1").Select
End Sub


Sub GoToParameters()


If Range("I81") = True And Cells(46, 20) <> 2 Then
   RndMsg = MsgBox("Go to OUTPUT PARAMETERS
sheet and Reset Max and Min to None.")
   Range("I81") = False
   Exit Sub
End If

End Sub


Sub GoToSetupSheet()
'
' GoToSetupSheet Macro
' Macro recorded 11/2/2002 by Norman Pillsbury


With Application
   With ActiveWindow
         .Top = 1
         .Left = 1
          .Height = 1900
          .Width = 1640
        .Height = Application.UsableHeight - 50
        .Width = Application.UsableWidth - 50
      End With

         '.WindowState = xlMaximized
   End With

   Range("A1").Select
   Sheets("SETUP").Select
   Range("A1").Select
End Sub
```

```
Sub Import()


On Error GoTo Errhandler

'Variable Declaration
Dim OutPut As Integer
'Example of vbYesNo
OutPut = MsgBox("Before Importing Database, all Existing Data Will be Erased." & vbCr & "This CANNOT be Undone." & vbCr & "Importing Data May Take Several Minutes." & vbCr & vbCr & "Do You Want To Continue?", vbYesNo, "YOU ARE IMPORTING NEW DATA INTO THIS PROGRAM")
If OutPut = 7 Then    'Output = 7(No)
   Exit Sub
   Else
   'Output = 6(Yes)
End If

   'Sets Import sheet error code to none
   Sheets("ID").Select
   Range("AI3").Select
   ActiveCell.Formula = ""
   Range("A1").Select

Application.Run "DoItIMPORT"
Application.Run "DefaultReset"

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames permanently for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues, Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   'This is name of main program: CUFIM-PRO date/time
   Dim CUFIMName As String
   CUFIMName = Range("R6")

   Dim CUFIMProgram As String    'Name of main program file
   Dim CurrentDir As String
   Template = "Export Template.xlsx"    'Name of Template that becomes the NewCUFIMName
   NewCUFIMName = Range("S28") & "  " & Format(Date, "mmm dd, yyyy") & "--Time is" & Format(Time, " hh mm ss") & ".xlsx"
   CUFIMProgram = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim DBBackup As String
   DBBackup = Range("S26")
   Dim ProgBackup As String
   ProgBackup = Range("S17")

   'BaseDir ==> Points to Desktop where CUFIM folder resides
   'BaseDir & CUFIMFolder==> Points to inside of CUFIM folder
   'BaseDir & CUFIMFolder & DBBackup==> Points to inside Database Backup Folder
   'BaseDir & CUFIMFolder & ProgBackup==> Points to inside Program Backup Folder

   'This sets directory to CUFIM folder -- but doesn't work----------
   ChDir (BaseDir & CUFIMFolder)

   'User selects file to open
   Dim DBName As String
   DBName = Application.GetOpenFilename    'This is database to be imported
   Workbooks.Open DBName, UpdateLinks:=0
   Range("A1").Select
   'NEED STATEMENT HERE IN CASE CANCEL IS SELECTED....


   'Getting import file name as variable
   Application.Calculate
   Windows(CUFIMName).Activate
   Sheets("ID").Select

   Range("X6").Select
   Selection.Copy
   Range("R6").Select
   Selection.PasteSpecial Paste:=xlValues, Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Application.CutCopyMode = False

   Dim myImport As String
   myImport = Range("R6")
   Dim R
   R = 0

   'Imports Database Name from Import File
   Windows(myImport).Activate
```

```
   Range("C1").Select
   Selection.Copy
   Windows(CUFIMName).Activate
   Range("S15").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
   With Selection
      .HorizontalAlignment = xlGeneral
      .VerticalAlignment = xlCenter
   End With

   Range("S18").Select
   Selection.Copy
   Range("S15").Select
   Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Application.CutCopyMode = False
   Range("S10").Select

   'Numbers Seq. Number and Tree Record Number from
1-n.
   'Consecutive no's for Seq No. and Tree Record columns
   Windows(myImport).Activate
   Range("F50050").End(xlUp).Select     'Species Code
column in Import Template file
   RowNumberF = ActiveCell.Row
    R = RowNumberF

   For A = 5 To R
      Cells(A, 4) = A - 4
      Cells(A, 5) = A - 4
   Next A

   'Imports Species Code, Species List and Species Group
   Windows(myImport).Activate
   Range("A5:C504").Select
   Application.CutCopyMode = False
   Selection.Copy
   Windows(CUFIMName).Activate
   Sheets("SETUP").Select
   Range("B36").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False

   'Imports Seq No., Tree Record No., Species Code
   Windows(myImport).Activate
   Range("D5:F50004").Select
   Application.CutCopyMode = False
   Selection.Copy
   Windows(CUFIMName).Activate
   Sheets("DATABASE").Select
   Range("O54").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False

   'Not importing Species Name; CUFIM automatically
generates that name from SETUP sheet.

   'Imports Dbh and Height
   Windows(myImport).Activate
   Range("H5:I50004").Select
   Application.CutCopyMode = False
   Selection.Copy
   Windows(CUFIMName).Activate
   Sheets("DATABASE").Select
   Range("U54").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False

   'Imports Variables 1-10
   Windows(myImport).Activate
   Range("K5:T50004").Select
   Application.CutCopyMode = False
   Selection.Copy
   Windows(CUFIMName).Activate
   Sheets("DATABASE").Select
   Range("AF54").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False

   Windows(myImport).Activate
   Range("K4:T4").Select
   Selection.Copy
   Windows(CUFIMName).Activate
   Sheets("DATABASE").Select
   Range("M28").Select
   Selection.PasteSpecial Paste:=xlAll,
Operation:=xlNone, SkipBlanks:=False _
      , Transpose:=True
   Range("P28:Q28").Select
      ActiveCell.FormulaR1C1 = "=IF(RC[-3]="""",RC[-
5],RC[-3])"
   Selection.AutoFill Destination:=Range("P28:Q37"),
Type:=xlFillDefault
   Range("P28:Q37").Select
   Range("N37").Select
   Application.CutCopyMode = False

   'Closes Import file
   Windows(myImport).Activate
   Application.CutCopyMode = False
   Range("Q20").Select
   Windows(myImport).Close savechanges:=False
   'ActiveWorkbook.Close savechanges:=False

   Windows(CUFIMName).Activate
   Sheets("SETUP").Select
   Range("B35:D35").Select
   Selection.Borders(xlBottom).LineStyle = xlContinuous

   Range("B36:D540").Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
```

```
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlInsideVertical)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    Application.CutCopyMode = False
    Range("E33").Select

    Range("B532:D532").Select
    Selection.AutoFill Destination:=Range("B532:D538"),
Type:=xlFillDefault
    Range("B532:D538").Select

    Range("B36:B540").Select
    With Selection
        .HorizontalAlignment = xlCenter
    End With
    Range("D36:D540").Select
    With Selection
        .HorizontalAlignment = xlCenter
    End With
    Application.CutCopyMode = False
    Range("E10").Select

    'Copies equations down.....to.....
    Sheets("DATABASE").Select
    'Range("O50053").End(xlUp).Select
    'RowNumber = ActiveCell.Row
    'MsgBox RowNumber

    'Puts 1, 2, 3 4 in first 4 rows, otherwise the next steps
wipe out S, T, W 52:54.
    Range("O54").Select
    ActiveCell.Formula = 1
    Range("O55").Select
    ActiveCell.Formula = 2
    Range("O56").Select
    ActiveCell.Formula = 3
    Range("O57").Select
    ActiveCell.Formula = 4

    Range("S54").Select
    Selection.AutoFill Destination:=Range("S54:S" &
Cells(Rows.Count, "O").End(xlUp).Row)
    Range("T54").Select
    Selection.AutoFill Destination:=Range("T54:T" &
Cells(Rows.Count, "O").End(xlUp).Row)
    Range("W54").Select
    Selection.AutoFill Destination:=Range("W54:W" &
Cells(Rows.Count, "O").End(xlUp).Row)
    'Selection.AutoFill Destination:=Range("A2:A" &
Cells(Rows.Count, "B").End(xlUp).Row)

    'This undoes the 1, 2, 3, 4 input above.
    If Cells(54, 16) > 0 Then
    GoTo NoClear:
    End If
    Range("O54:O57").Select
    Selection.ClearContents
    Range("O45").Select
NoClear:


    Application.Run "FormatDatabase"

    Sheets("ID").Select
    Dim CUFIMProgramName As String
    CUFIMProgramName = Range("S18") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsm"

    'BaseDir ==> Points to Desktop where CUFIM folder
resides
    'BaseDir & CUFIMFolder==> Points to inside of CU-
FIM folder
    'BaseDir & CUFIMFolder & DBBackup==> Points to
inside Database Backup Folder
    'BaseDir & CUFIMFolder & ProgBackup==> Points to
inside Program Backup Folder

    'SAVE CUFIM Program File with new name (date/
time).....
    ChDir (BaseDir & CUFIMFolder & ProgBackup)
    ActiveWorkbook.SaveAs CUFIMProgramName

    CurrentDir = Range("T6")
    ChDir CurrentDir
```

```
   Dim CurrentCUFIMPRO As String
   CurrentCUFIMPRO = Range("X6")
   Windows(CurrentCUFIMPRO).Activate
   Sheets("ID").Select


GoTo ENDIMPORT:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDIMPORT:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Application.Run "DoItIMPORT"

Beep

Range("A1").Select

End Sub




Sub LoadDBeqn()
'
' LoadDBeqn Macro
'

'
   Range("J54:K54").Select
   Selection.Copy
   Range("S54").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
   Range("L54").Select
   Selection.Copy
   Range("W54").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
   Range("P45").Select
End Sub




Sub MarkAllTrees()

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("STATS").Select
MaxNo = Range("I22")       'Max number of trees in
database
Sheets("DATABASE").Select

Dim F(50, 1)     'Data Matrix for trees included in OUT-
PUT PARAMETERS stand table

'Clears Removal Column on Database sheet
   Range("R54:R50053").Select
   Selection.Clear
   Range("R45").Select

'Reads in Sp Codes from Inclusion list into F[*] from
DATABASE sheet
For A = 1 To 50
   If Cells(75, 55) > 0 Then
      F(A, 1) = Cells(24 + A, 55)
   Else
      A = 120
   End If
Next A

 'Places a "1" in Col 4 to indicate trees that are cut trees
   For A = 54 To MaxNo + 54              'Cks all trees in
database

  'This tree record based on restrictions in Step 7 for
<lower limit and >upper limit
    If Cells(A, 21) < Cells(20, 55) Or Cells(A, 21) >
Cells(21, 55) Then
     GoTo NextA
     End If

     If Cells(A, 17) <> "" Then             'Species code ex-
ists.
        For E = 1 To 50                        'Cks if Database
tree is on Inclusion list
          If (Cells(A, 17)) = Cells(E + 24, 55) Then
            Cells(A, 18) = 1
           End If
           If Cells(E + 24, 55) = "" Then
           E = 50
           End If
         Next E
      End If
NextA:   Next A

'Formats Removal Column on Database sheet
   Range("R54:R50053").Select
   With Selection
      .HorizontalAlignment = xlCenter
   End With
```

```
    Range("R52:R53").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous

    Range("R54:R50052").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Selection.Borders(xlEdgeLeft).LineStyle = xlNone
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    Selection.Borders(xlEdgeRight).LineStyle = xlNone
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With

    Range("R54:R50052").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    Range("O45").Select

    'Sorts database for Removal Codes
    Sheets("DATABASE").Select
    Range("P54:AO50053").Select
        Selection.Sort Key1:=Range("R54"),
Order1:=xlAscending, Key2:=Range("Q54" _
        ), Order2:=xlAscending, Header:=xlNo, Order-
Custom:=1, MatchCase:= _
        False, Orientation:=xlTopToBottom
        Range("CU23").Select
        ActiveCell.FormulaR1C1 = ""
    Sheets("OUTPUT PARAMETERS").Select

    Beep

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub Method1Check()
'
' Method1Check Macro
' Macro recorded 7/22/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

    Range("AY126").Select
    ActiveCell.FormulaR1C1 = 1

    Range("AU129").Select
    ActiveCell.FormulaR1C1 = 1

    Range("AY129:AY142").Select
    Selection.Clear

If Range("R10") = False Then
    Range("P22:R531").Select
    Selection.Clear

    'Replace lines with white interior
    Range("P20:R20").Select
    Selection.AutoFill Destination:=Range("P20:R546"),
Type:=xlFillDefault
```

```
   Range("O25:O26").Select
   Selection.Borders(xlRight).LineStyle = xlContinuous

   Range("Q22").Select

Exit Sub
End If

   Range("P26").Select
   Selection.Font.Bold = True

   Range("P27:P531").Select
   Selection.NumberFormat = "$#,##0.00     "
   With Selection
      .HorizontalAlignment = xlRight
   End With

   Range("P27:P531").Select
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
   End With
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlThin
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
   End With

   Range("P24:P26").Select
   Selection.ClearContents
   Selection.Borders(xlLeft).LineStyle = xlNone
   Selection.Borders(xlRight).LineStyle = xlNone
   Selection.Borders(xlTop).LineStyle = xlNone
   Selection.Borders(xlBottom).LineStyle = xlNone
   Range("P24").Select
   ActiveCell.FormulaR1C1 = "Same Value"
   Range("P25").Select
   ActiveCell.FormulaR1C1 = "Method"
   Range("P26").Select
   ActiveCell.FormulaR1C1 = "($/cf)"
   Range("P26").Select
   Selection.Font.Bold = False
   Range("P24:P26").Select
   Selection.Font.Bold = True
   With Selection
      .HorizontalAlignment = xlCenter
   End With
   With Selection.Font
      .Name = "Geneva"
      .Size = 9
   End With
   Range("P24:P26").Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone
   Range("R29").Select

   Range("P24:P26").Select
   Selection.AutoFill Destination:=Range("P24:Q26"),
Type:=xlFillDefault
   Range("P24:Q26").Select
   Range("Q24").Select
   ActiveCell.FormulaR1C1 = "Volume"
   Range("Q25").Select
   ActiveCell.FormulaR1C1 = "by Species"
   Range("Q26").Select
   ActiveCell.FormulaR1C1 = "(cf)"
   Range("Q27:Q531").Select
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
   End With
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
   End With
   Selection.Borders(xlEdgeBottom).LineStyle = xlNone
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
   End With
   Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone
   Selection.NumberFormat = "#,##0  "
   Range("Q22").Select
```

```
    Selection.NumberFormat = "#,##0  "
    Range("Q24:Q26").Select
    Selection.AutoFill Destination:=Range("Q24:R26"),
Type:=xlFillDefault
    Range("Q24:R26").Select

    Range("R24").Select
    ActiveCell.FormulaR1C1 = "Value"
    Range("R26").Select
    ActiveCell.FormulaR1C1 = "($)"
    Range("R27").Select
    ActiveCell.FormulaR1C1 = "=IF(RC[-1]>0,RC[-
1]*RC[-2],"""")"
    Range("R27:R531").Select
    Selection.FillDown
    Range("Q22").Select
    Selection.AutoFill Destination:=Range("Q22:R22"),
Type:=xlFillDefault
    Range("Q22:R22").Select
    Range("R22").Select
    Selection.NumberFormat = "$#,##0  "
    Range("R27:R531").Select
    Selection.NumberFormat = "$#,##0  "
    Range("P22").Select
    ActiveCell.FormulaR1C1 = "Totals ="
    Range("P22").Select
    With Selection
        .HorizontalAlignment = xlRight
        .VerticalAlignment = xlBottom
    End With
    Range("P22:R22").Select
    Selection.Font.Bold = True
    With Selection.Font
        .Name = "Arial"
        .Size = 11
        .Underline = xlUnderlineStyleNone
        .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Range("R27:R531").Select
    Selection.Borders(xlRight).LineStyle = xlContinuous

    Range("R20").Select
    Range("R24:R26").Select
    With Selection.Interior
        .ColorIndex = 36
        .Pattern = xlSolid
    End With

    Range("P25:R25").Select
    With Selection
        .HorizontalAlignment = xlCenter
        .VerticalAlignment = xlTop
        .Orientation = 0
    End With

    Range("N25").Select
    Selection.Copy
    Range("P24:Q26").Select
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Selection.Borders(xlTop).LineStyle = xlContinuous
    Selection.Borders(xlTop).LineStyle = xlContinuous
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Range("T22").Select

    Range("P25:Q25").Select
    Selection.Borders(xlTop).LineStyle = xlNone
    Range("P24:Q24").Select
    Selection.Borders(xlBottom).LineStyle = xlNone
    Range("P25:Q25").Select
    Selection.Borders(xlBottom).LineStyle = xlNone
    Range("P26:Q26").Select
    Selection.Borders(xlTop).LineStyle = xlNone
    Range("P25:Q25").Select
    With Selection
        .HorizontalAlignment = xlCenter
        .VerticalAlignment = xlTop
    End With

    Range("P532").Select
    Selection.Borders(xlTop).LineStyle = xlContinuous
    Range("R532").Select
    Selection.Borders(xlTop).LineStyle = xlContinuous
    Range("R27:R530").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeTop)
       .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    Range("R532:R546").Select
    Selection.AutoFill Destination:=Range("L532:R546"),
Type:=xlFillDefault

Columns("N:N").EntireColumn.AutoFit
Range("L1").Select
```

```
   Range("R24").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With
   Range("R25").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With

   Range("M27:R531").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With

Range("L1").Select

GoTo ENDMETH1CK:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDMETH1CK:


Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

End Sub




Sub Method1SortSPCode()
'
' Method1Sort Macro
' Macro recorded 1/1/1970 by Norman Pillsbury
'
   ActiveWindow.Panes(1).Activate
   Range("M27:R531").Select
   Selection.Sort Key1:=Range("M27"),
Order1:=xlAscending, Header:=xlNo, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
   Range("M24").Select
End Sub
```

```
Sub Method1SortValue()
'
' Method1SortValue Macro
' Macro recorded 1/2/1970 by Norman Pillsbury
'
Sheets("VALUATION").Select

   Range("M27:R531").Select
   Selection.Sort Key1:=Range("Q27"),
Order1:=xlDescending, Header:=xlNo, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
   Range("R20").Select
End SubL



Sub Method2Check()
'
' Method2Check Macro
' Macro recorded 7/22/2002 by Norman Pillsbury
'
On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

If Range("X10") = False Then
   Range("Y21:AK76").Select
   Selection.Clear

   Range("AL21:AL76").Select
   Selection.AutoFill Destination:=Range("Y21:AL76"),
Type:=xlFillDefault
   Range("Y21:AL76").Select
   Range("AM21").Select

   Range("X9") = False
   Range("X8") = False
   Range("Y22").Select
   Selection.Clear

   Columns("Z:AK").ColumnWidth = 0
   Columns("AL:AN").ColumnWidth = 9.5

   Range("Y25:Y26").Select
   Selection.Borders(xlLeft).LineStyle = xlContinuous

   Range("W11:AN19").Select
   With Selection.Interior
```

```
      .ColorIndex = 37
      .Pattern = xlSolid
   End With
   Range("AL11:AL19").Select
   Selection.Borders(xlRight).LineStyle = xlNone
   Range("W11:AN11").Select
   Selection.Borders(xlTop).LineStyle = xlContinuous
   Range("AN11:AN19").Select
   Selection.Borders(xlRight).LineStyle = xlContinuous
   Range("AM19:AN19").Select
   Range("AN19").Activate
   Selection.Borders(xlBottom).LineStyle = xlContinuous
   Range("W21").Select

   Range("AL19").Select
   Selection.Borders(xlBottom).LineStyle = xlContinuous

Range("Q22").Select

Exit Sub
End If

If Range("X10") = True Then
   Range("W11:AN19").Select
   Selection.Interior.ColorIndex = 37
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone
   Range("W21").Select
End If

   Columns("Y:Y").ColumnWidth = 9.5
   Range("Y25:Y76").Select
   Range("Y76").Activate
   Selection.Borders(xlRight).LineStyle = xlContinuous
   Range("Y25").Select
   ActiveCell.FormulaR1C1 = "Value"
   Range("Y26").Select
   ActiveCell.FormulaR1C1 = "($/cf)"
   Range("Y25:Y26").Select
   With Selection
      .HorizontalAlignment = xlCenter
   End With

   Range("W25:Y26").Select
   With Selection.Font
      .Name = "Geneva"
      .Size = 9
   End With
   Selection.Font.Bold = True
   Range("V30").Select

   Columns("X:X").ColumnWidth = 16.17

   Range("Y27:Y76").Select
   With Selection.Interior
      .ColorIndex = 36
      .Pattern = xlSolid
   End With

   Range("Y27:Y76").Select
   Selection.NumberFormat = "$#,##0.00  "
   With Selection
      .HorizontalAlignment = xlRight
   End With

   Range("Y25").Select
   Selection.Borders(xlTop).LineStyle = xlContinuous

   Range("Y27:Y76").Select
   Selection.Borders(xlLeft).LineStyle = xlContinuous
   Range("Y76").Select
   Selection.Borders(xlBottom).LineStyle = xlContinuous
   Range("W1").Select

   Range("Y25:Y26").Select
   Selection.AutoFill Destination:=Range("Y25:AK26"),
Type:=xlFillDefault
   Range("Y25:AK26").Select
   Range("AJ25").Select
   ActiveCell.FormulaR1C1 = "Volume"
   Range("AJ26").Select
   ActiveCell.FormulaR1C1 = "(cf)"
   Range("AK25").Select
   ActiveCell.FormulaR1C1 = "Total Value"
   Columns("AK:AK").ColumnWidth = 12
   Range("AK26").Select
   ActiveCell.FormulaR1C1 = "($)"
   Range("AJ27:AK76").Select
   Selection.Borders(xlRight).LineStyle = xlContinuous
   Range("Y76:AK76").Select
   Range("AK76").Activate
```

```
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Range("AK23").Select
    Columns("AJ:AJ").ColumnWidth = 12
    Columns("AN:AN").ColumnWidth = 12.5

    Range("Y26:AK26").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous

    Columns("Z:AI").ColumnWidth = 0
    Range("W9").Select

    Range("X4:X6").Select
    Selection.Borders(xlRight).LineStyle = xlNone

GoTo ENDMETH2CK:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDMETH2CK:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
    With Application
        .Calculation = xlAutomatic
        .MaxChange = 0.001
    End With

Range("Y22").Select

End Sub
```

```
ub NoDiaLimitsBoxA1()
'
' NoDiaLimitsBoxA1 Macro
' Macro recorded 10/31/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("OUTPUT PARAMETERS").Select

If Range("AD11") <> "None" And Range("AD12") <>
"None" Then
    Sheets("REMOVAL").Select
    Range("I81").Select
    ActiveCell.FormulaR1C1 = False
    RndMsg = MsgBox("For this Choice you must set
the MIN and MAX Diameter limits on the OUTPUT
PARAMETERS sheet to NONE.")

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Exit Sub
End If

If Range("AD11") = "None" And Range("AD12") =
"None" Then
    Sheets("REMOVAL").Select
    Range("I81").Select
    ActiveCell.FormulaR1C1 = True
    Range("J81").Select
    ActiveCell.FormulaR1C1 = False
    Range("K81").Select
    ActiveCell.FormulaR1C1 = False

Else
    Sheets("REMOVAL").Select
    Range("I81").Select
    ActiveCell.FormulaR1C1 = False
    Range("J81").Select
    ActiveCell.FormulaR1C1 = False
    Range("K81").Select
    ActiveCell.FormulaR1C1 = False

End If

Sheets("REMOVAL").Select
Range("H16").Select


If (Range("I81") = True Or Range("J81") = True Or
Range("K81") = True) And Range("AA5") = False Then

    Range("Z5").Select
    ActiveCell.FormulaR1C1 = False

    Range("AA5").Select
    ActiveCell.FormulaR1C1 = False

    Range("I69").Select
    ActiveCell.FormulaR1C1 = 1

    Sheets("REMOVAL").Select
    Range("I81").Select
    ActiveCell.FormulaR1C1 = False
    Range("J81").Select
    ActiveCell.FormulaR1C1 = False
    Range("K81").Select
    ActiveCell.FormulaR1C1 = False
```

```
   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Q26:Q40").Select
   Selection.ClearContents

   Range("AH26:AH40").Select
   Selection.ClearContents

   Range("L26:L40").Select
   Selection.ClearContents

   Range("AC26:AC40").Select
   Selection.ClearContents

   Range("Q20").Select
   RndMsg = MsgBox("First -- Select Box A or B.")
End If

Application.Run "GenerateNoTreesinDBHclassA"


GoTo ENDNODIAA1:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDNODIAA1:


Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

If (Range("I69") = 1 And Range("AA5")) = True Then
   RndMsg = MsgBox("Select Percent in Part B.")
End If


End Sub




Sub NoDiaLimitsBoxA2()
'
' NoDiaLimitsBoxA2 Macro
' Macro recorded 10/31/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("OUTPUT PARAMETERS").Select
If Range("AD11") = "None" And Range("AD12") =
"None" Then
Sheets("REMOVAL").Select
   RndMsg = MsgBox("For this Choice you must first set
Diameter limits on the OUTPUT PARAMETERS sheet.")

   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

   Exit Sub
End If

Sheets("REMOVAL").Select
Range("H16").Select

If (Range("I81") = True Or Range("J81") = True Or
Range("K81") = True) And Range("AA5") = False Then

   Range("Z5").Select
   ActiveCell.FormulaR1C1 = False

   Range("AA5").Select
   ActiveCell.FormulaR1C1 = False

   Range("I69").Select
   ActiveCell.FormulaR1C1 = 1

   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
```

```
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Q26:Q40").Select
   Selection.ClearContents

   Range("AH26:AH40").Select
   Selection.ClearContents

   Range("L26:L40").Select
   Selection.ClearContents

   Range("AC26:AC40").Select
   Selection.ClearContents

   Range("Q20").Select
   RndMsg = MsgBox("First -- Select Box A or B.")
End If

   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = True
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

Application.Run "GenerateNoTreesinDBHclassA"

   Range("Q26:Q40").Select
   Selection.ClearContents

GoTo ENDNODIAA2:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDNODIAA2:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

If (Range("I69") = 1 And Range("AA5")) = True Then
   RndMsg = MsgBox("Select Percent in Part B.")
End If

   Range("Q19").Select

End Sub
```

```
Sub NoDiaLimitsBoxA3()
'
' NoDiaLimitsBoxA3 Macro
' Macro recorded 10/31/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("OUTPUT PARAMETERS").Select
If Range("AD11") = "None" And Range("AD12") =
"None" Then
Sheets("REMOVAL").Select
   RndMsg = MsgBox("For this Choice you must first set
Diameter limits on the OUTPUT PARAMETERS sheet.")
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

   Exit Sub
End If

Sheets("REMOVAL").Select
Range("H16").Select

If (Range("I81") = True Or Range("J81") = True Or
Range("K81") = True) And Range("AA5") = False Then

   Range("Z5").Select
   ActiveCell.FormulaR1C1 = False

   Range("AA5").Select
   ActiveCell.FormulaR1C1 = False

   Range("I69").Select
   ActiveCell.FormulaR1C1 = 1

   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False
```

```
   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Q26:Q40").Select
   Selection.ClearContents

   Range("AH26:AH40").Select
   Selection.ClearContents

   Range("L26:L40").Select
   Selection.ClearContents

   Range("AC26:AC40").Select
   Selection.ClearContents

   Range("Q20").Select
   RndMsg = MsgBox("First -- Select Box A or B.")
End If

   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = True

Application.Run "GenerateNoTreesinDBHclassA"

   Range("Q26:Q40").Select
   Selection.ClearContents

GoTo ENDNODIAA3:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDNODIAA3:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

If (Range("I69") = 1 And Range("AA5")) = True Then
   RndMsg = MsgBox("Select Percent in Part B.")
End If

   Range("Q19").Select

End Sub
```

```
Sub NoDiaLimitsBoxB1()
'
' NoDiaLimitsBoxB1 Macro
' Macro recorded 10/31/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("REMOVAL").Select

If Range("Z5") = False Then
   RndMsg = MsgBox("You must select Box A or Box B.")
   Application.Run "ClearChoicesB"
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic
   Exit Sub
End If
Sheets("REMOVAL").Select

Sheets("OUTPUT PARAMETERS").Select

If Range("AD11") <> "None" And Range("AD12") <>
"None" Then
   RndMsg = MsgBox("For this Choice you must set
the MIN and MAX Diameter limits on the OUTPUT
PARAMETERS sheet to NONE.")
   Sheets("REMOVAL").Select
   Application.Run "ClearChoicesB"
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
Exit Sub
End If
   Sheets("REMOVAL").Select

If Range("AD41") = 0 Then
   RndMsg = MsgBox("You must enter percent values in
Col 5 before continuing.")
   Application.Run "ClearChoicesB"
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic
Exit Sub
End If
   Sheets("REMOVAL").Select
```

```
Sheets("OUTPUT PARAMETERS").Select
If Range("AD11") = "None" And Range("AD12") =
"None" Then
   Sheets("REMOVAL").Select
   Range("Z81").Select
   ActiveCell.FormulaR1C1 = True
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False
Else
   Sheets("REMOVAL").Select
   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False
End If

Sheets("REMOVAL").Select
Range("Z16").Select

If (Range("Z81") = True Or Range("AA81") = True Or
Range("AB81") = True) And Range("Z5") = False Then

   Range("Z5").Select
   ActiveCell.FormulaR1C1 = False

   Range("AA5").Select
   ActiveCell.FormulaR1C1 = False

   Range("I69").Select
   ActiveCell.FormulaR1C1 = 1

   Sheets("REMOVAL").Select
   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False

   Sheets("REMOVAL").Select
   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Q26:Q40").Select
   Selection.ClearContents

   Range("AH26:AH40").Select
   Selection.ClearContents

   Range("L26:L40").Select
   Selection.ClearContents

   Range("AC26:AC40").Select
   Selection.ClearContents

   Range("Q20").Select
   RndMsg = MsgBox("First -- Select Box A or B.")
End If

Application.Run "GenerateNoTreesinDBHclassA"

GoTo ENDNODIABOXB1:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDNODIABOXB1:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

'If (Range("I69") = 1 And Range("Z5")) = True Then
'    RndMsg = MsgBox("Select Percent in Part B.")
'End If

End Sub
```

```
Sub NoDiaLimitsBoxB2()
'
' NoDiaLimitsBoxB2 Macro
' Macro recorded 10/31/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("REMOVAL").Select

If Range("Z5") = False Then
   RndMsg = MsgBox("You must select Box A or Box B.")
   Application.Run "ClearChoicesB"
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
```

```
   Application.Calculation = xlAutomatic
   Exit Sub
End If

Sheets("OUTPUT PARAMETERS").Select

If Range("AD11") = "None" And Range("AD12") =
"None" Then
Sheets("REMOVAL").Select
   RndMsg = MsgBox("For this Choice you must first set
Diameter limits on the OUTPUT PARAMETERS sheet.")
   Application.Run "ClearChoicesB"
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
   Exit Sub
End If

Sheets("REMOVAL").Select

If Range("AD41") = 0 Then
   RndMsg = MsgBox("You must enter percent values in
Col 5 before continuing.")
   Application.Run "ClearChoicesB"
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic
   Exit Sub
End If

Sheets("REMOVAL").Select
Range("H16").Select

If (Range("Z81") = True Or Range("AA81") = True Or
Range("AB81") = True) And Range("Z5") = False Then

   Range("Z5").Select
   ActiveCell.FormulaR1C1 = False

   Range("AA5").Select
   ActiveCell.FormulaR1C1 = False

   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False

   Sheets("REMOVAL").Select
   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False

   Sheets("REMOVAL").Select
   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Q26:Q40").Select
   Selection.ClearContents

   Range("AH26:AH40").Select
   Selection.ClearContents

   Range("L26:L40").Select
   Selection.ClearContents

   Range("AC26:AC40").Select
   Selection.ClearContents

   Range("Q20").Select
   RndMsg = MsgBox("First -- Select Box A or B.")
End If

   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = True
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False

Application.Run "GenerateNoTreesinDBHclassA"

   Range("AH26:AH40").Select
   Selection.ClearContents

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

If (Range("Z69") = 1 And Range("Z5")) = True Then
   RndMsg = MsgBox("Select Percent in Part B.")
End If

   Range("Q19").Select

Sheets("REMOVAL").Select
Range("X3").Select


GoTo ENDNODIABOXB2:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDNODIABOXB2:
```

```
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub NoDiaLimitsBoxB3()
‘
‘ NoDiaLimitsBoxB3 Macro
‘ Macro recorded 10/31/2002 by Norman Pillsbury

On Error GoTo Errhandler

Sheets(“REMOVAL”).Select

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

If Range(“Z5”) = False Then
   RndMsg = MsgBox(“You must select Box A or Box B.”)
   Range(“AB81”) = False
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic
   Exit Sub
End If

Sheets(“OUTPUT PARAMETERS”).Select

If Range(“AD11”) = “None” And Range(“AD12”) =
“None” Then
Sheets(“REMOVAL”).Select
   RndMsg = MsgBox(“For this Choice you must first set
Diameter limits on the OUTPUT PARAMETERS sheet.”)
   Range(“Z81”).Select
   ActiveCell.FormulaR1C1 = False
   Range(“AA81”).Select
   ActiveCell.FormulaR1C1 = False
   Range(“AB81”).Select
   ActiveCell.FormulaR1C1 = False
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
   Exit Sub
End If
Sheets(“REMOVAL”).Select

If Range(“AD41”) = 0 Then
   RndMsg = MsgBox(“You must enter percent values in
Col 5 before continuing.”)
   Range(“AB81”) = False
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic
   Exit Sub
End If

Sheets(“REMOVAL”).Select
Range(“Z16”).Select

If (Range(“Z81”) = True Or Range(“AA81”) = True Or
Range(“AB81”) = True) And Range(“Z5”) = False Then

   Range(“Z5”).Select
   ActiveCell.FormulaR1C1 = False

   Range(“AA5”).Select
   ActiveCell.FormulaR1C1 = False

   Range(“Z81”).Select
   ActiveCell.FormulaR1C1 = 1

   Sheets(“REMOVAL”).Select
   Range(“Z81”).Select
   ActiveCell.FormulaR1C1 = False
   Range(“AA81”).Select
   ActiveCell.FormulaR1C1 = False
   Range(“AB81”).Select
   ActiveCell.FormulaR1C1 = False

   Sheets(“REMOVAL”).Select
   Range(“Z81”).Select
   ActiveCell.FormulaR1C1 = False
   Range(“AA81”).Select
   ActiveCell.FormulaR1C1 = False
   Range(“AB81”).Select
   ActiveCell.FormulaR1C1 = False

   Range(“Q26:Q40”).Select
   Selection.ClearContents

   Range(“AH26:AH40”).Select
   Selection.ClearContents

   Range(“L26:L40”).Select
   Selection.ClearContents

   Range(“AC26:AC40”).Select
   Selection.ClearContents

   Range(“Q20”).Select
   RndMsg = MsgBox(“First -- Select Box A or B.”)
```

```
End If

   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = True

Application.Run "GenerateNoTreesinDBHclassA"

   Range("AH26:AH40").Select
   Selection.ClearContents

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Range("X12").Select

GoTo ENDNODIABOXB3:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDNODIABOXB3:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub NoOfSpeciesOnSetupSheet()

'Calculates number of species per and their volume in
database

   Sheets("SETUP").Select
   Range("E37:F540").Select
   Selection.Clear

   Range("E36").Select
   Selection.AutoFill Destination:=Range("E36:E" &
Cells(Rows.Count, "B").End(xlUp).Row)
   Range("F36").Select
   Selection.AutoFill Destination:=Range("F36:F" &
Cells(Rows.Count, "B").End(xlUp).Row)
   Application.Calculate

   Range("E36:F540").Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   Selection.Borders(xlInsideVertical).LineStyle = xlNone
   With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
      With Selection.Borders(xlInsideVertical)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With

   Range("E33").Select
   Application.CutCopyMode = False

End Sub



Sub OutputDiaBoxesA()
'
' NoDiaLimitsBoxA3 Macro
' Macro recorded 10/31/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
```

```
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("OUTPUT PARAMETERS").Select

If Range("AD11") = "None" Or Range("AD12") = "None"
Then
   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False

End If

If Range("AD11") > 0 Or Range("AD12") > 0 Then
   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Z81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AA81").Select
   ActiveCell.FormulaR1C1 = False
   Range("AB81").Select
   ActiveCell.FormulaR1C1 = False

End If

Sheets("OUTPUT PARAMETERS").Select
Range("AG9").Select

GoTo ENDOUTPUTBOXA:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDOUTPUTBOXA:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub PrintDatabaseOptions()
'
' PrintDatabaseOptions Macro
' Macro recorded 8/29/2002 by Norman Pillsbury

On Error GoTo Errhandler

'This macro selects data from DATABASE and pastes and
formats it onto
'Scratch Workspace sheet and prints it from there in
order to retain
'all headings and summary rows.

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   Sheets("DATABASE").Select
   If Range("I7") < 3 Then
   Exit Sub
   End If

'==========================
'
' This section prints first 50 records on one sheet.
'First sheet (columns 1-16)

If Range("I7") = 3 Then

   Range("O46:AC103").Select
   ActiveSheet.PageSetup.PrintArea = ""
   With ActiveSheet.PageSetup
      .CenterHeader = "&36&K000000DATABASE (first
50 trees)"
      .RightFooter = "&K000000Page 1"
      .PrintComments = xlPrintNoComments
      .PrintQuality = -4
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .PaperSize = xlPaperLetter
      .FirstPageNumber = xlAutomatic
      .Order = xlDownThenOver
      .BlackAndWhite = False
      .Zoom = False
      .FitToPagesWide = 1
      .FitToPagesTall = 1
```

```
    End With
    Selection.PrintOut Copies:=1

    End If

    'Second sheet (columns 17-26)

If Range("I7") = 4 Then

    Range("AF46:AO103").Select
    ActiveSheet.PageSetup.PrintArea = ""
    With ActiveSheet.PageSetup
      .CenterHeader = "&36&K000000DATABASE (first
50 trees)"
      .RightFooter = "&K000000Page 2"
      .PrintComments = xlPrintNoComments
      .PrintQuality = -4
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .PaperSize = xlPaperLetter
      .FirstPageNumber = xlAutomatic
      .Order = xlDownThenOver
      .BlackAndWhite = False
      .Zoom = False
      .FitToPagesWide = 1
      .FitToPagesTall = 1
    End With
    Selection.PrintOut Copies:=1
    Range("AT67").Select

End If

'============================
'BEGINNING OF PRINT OPTION 3 (Species & Volume
Data, Columns 1-15 ALL pages).

If Range("I7") = 6 Then

    Sheets("Scratch Workspace").Select
    Range("CA212:CO265").Select
    Selection.Clear
    Range("CA211").Select

    'Prints Page X of Y (uses formats in CN&CO208)
    Range("CN208").Select
    ActiveCell.FormulaR1C1 = 1

    'Select and Paste Count and Volume Sums
    Sheets("DATABASE").Select
    Range("O46:AC48").Select
    Selection.Copy
    Sheets("Scratch Workspace").Select

    Range("CA212").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False

    'Copies 1st half of column headings over
    Sheets("DATABASE").Select
    Range("O52:W53").Select
    Application.CutCopyMode = False
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CA216").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False

    'Copies Vol by dia class
    Sheets("DATABASE").Select
    Range("X53:AC53").Select
    Application.CutCopyMode = False
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CJ217").Select
    ActiveSheet.Paste
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False

    'Copies Vol Header
    Sheets("DATABASE").Select
    Range("X52").Select
    Application.CutCopyMode = False
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CJ216").Select
    ActiveSheet.Paste

    Sheets("DATABASE").Select
    Range(Cells(54, 15), Cells(103, 29)).Select
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CA218").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False
```

```
    Range("CA200:CO267").Select
    With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.5)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(0.27)
      .BottomMargin = Application.InchesToPoints(0.56)
      .HeaderMargin = Application.InchesToPoints(0.17)
      .FooterMargin = Application.InchesToPoints(0.56)
      .PrintQuality = -4
     .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = 1
    End With
    Selection.PrintOut Copies:=1, Preview:=False
    Application.CutCopyMode = False

    'Clears Scratch Print Area
    Range("CA213:CO215").Select
    Selection.Delete Shift:=xlUp
    Range("CA210").Select

    Range("CA271:CO271").Select
    Selection.AutoFill Destination:=Range("CA215:
CO271"), Type:=xlFillDefault
    Range("CA215:CO271").Select
    Range("CA210").Select

    Sheets("DATABASE").Select

    For A = 51 To Int(Range("P48")) + 1 Step 50
      Range(Cells(A + 53, 15), Cells(A + 102, 29)).Select
      Selection.Copy
      Sheets("Scratch Workspace").Select

      Range("CN208").Select
      ActiveCell.FormulaR1C1 = Int(A / 50) + 1

      Range("CA215").Select
      Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
      Application.CutCopyMode = False
      Range("CA200:CO264").Select

      Selection.PrintOut Copies:=1, Preview:=False
      Application.CutCopyMode = False

      'Clears Scratch Print Area
      Range("CA215:CO265").Select
      Selection.ClearContents
      Range("CA211").Select

    Sheets("DATABASE").Select
    Next A

'============================
'END OF PRINT OPTION 3


'============================
'BEGINNING OF PRINT OPTION 4 (Species & Volume
Data, Columns 17-26 ALL pages).
ElseIf Range("I7") = 7 Then

    Sheets("Scratch Workspace").Select
    Range("CS212:DG265").Select
    Selection.Clear
    Range("CS211").Select

    'Prints Page X of Y (uses formats in CN&CO208)
    Range("DD208").Select
    ActiveCell.FormulaR1C1 = 1

    'Total number of pages

    'Select and Paste Count and Volume Sums for Cols 1,
2 & 3
    Sheets("DATABASE").Select
    Range("O46:Q48").Select
    Selection.Copy
    Sheets("Scratch Workspace").Select

    Range("CS212").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False

    'Select and Paste Count and Volume Sums for Cols
17-26
    Sheets("DATABASE").Select
    Range("AF46:AO48").Select
    Selection.Copy
    Sheets("Scratch Workspace").Select

    Range("CV212").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False

    'Copies 1st 3 column headings over
```

```
    Sheets("DATABASE").Select
    Range("O52:Q103").Select
    Application.CutCopyMode = False
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CS216").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False

    'Copies User Variables Headings Over
    Sheets("DATABASE").Select
    Range("AF52:AO103").Select
    Application.CutCopyMode = False
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CV216").Select
    ActiveSheet.Paste
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Application.CutCopyMode = False

    Sheets("Scratch Workspace").Select
    Columns("CS:CS").ColumnWidth = 11.67
    Columns("CT:CT").ColumnWidth = 10.83
    Columns("CU:CU").ColumnWidth = 12.17
    Columns("CV:CV").EntireColumn.AutoFit
    Columns("CW:CW").EntireColumn.AutoFit
    Columns("CX:CX").EntireColumn.AutoFit
    Columns("CY:CY").EntireColumn.AutoFit
    Columns("CZ:CZ").EntireColumn.AutoFit
    Columns("DA:DA").EntireColumn.AutoFit
    Columns("DB:DB").EntireColumn.AutoFit
    Columns("DC:DC").EntireColumn.AutoFit
    Columns("DD:DD").ColumnWidth = 14.33
    Columns("DE:DE").ColumnWidth = 14.33

    Range("CS200:DE267").Select
    With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.5)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(0.27)
      .BottomMargin = Application.InchesToPoints(0.56)
      .HeaderMargin = Application.InchesToPoints(0.17)
      .FooterMargin = Application.InchesToPoints(0.56)
      .PrintQuality = -4
      .Orientation = xlLandscape
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = 1
    End With
    Selection.PrintOut Copies:=1, Preview:=False
    Application.CutCopyMode = False

    'Cuts out Count headings
    Range("CS212:DE215").Select
    Selection.Delete Shift:=xlUp
    Range("CS210").Select

    'Clears Scratch Print Area
    Range("CS214:DE271").Select
    Selection.ClearContents
    Range("CS210").Select

    Sheets("DATABASE").Select

    For A = 51 To Int(Range("P48")) + 1 Step 50

      Range(Cells(A + 53, 15), Cells(A + 102, 17)).Select
      Selection.Copy
      Sheets("Scratch Workspace").Select
      Range("DD208").Select
      ActiveCell.FormulaR1C1 = Int(A / 50) + 1

      Range("CS215").Select
      Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
          False, Transpose:=False
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
          False, Transpose:=False
      Application.CutCopyMode = False

      Sheets("DATABASE").Select
      Range(Cells(A + 53, 32), Cells(A + 102, 41)).Select
      Selection.Copy

      Sheets("Scratch Workspace").Select
      Range("CV215").Select
      Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
          False, Transpose:=False
      Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
          False, Transpose:=False
      Application.CutCopyMode = False

      Sheets("Scratch Workspace").Select
      Columns("CS:CS").ColumnWidth = 11.67
      Columns("CT:CT").ColumnWidth = 10.83
      Columns("CU:CU").ColumnWidth = 12.17
      Columns("CV:CV").EntireColumn.AutoFit
      Columns("CW:CW").EntireColumn.AutoFit
      Columns("CX:CX").EntireColumn.AutoFit
      Columns("CY:CY").EntireColumn.AutoFit
      Columns("CZ:CZ").EntireColumn.AutoFit
      Columns("DA:DA").EntireColumn.AutoFit
```

```
    Columns("DB:DB").EntireColumn.AutoFit
    Columns("DC:DC").EntireColumn.AutoFit
    Columns("DD:DD").ColumnWidth = 14.33
    Columns("DE:DE").ColumnWidth = 14.33

    Range("CS200:DE264").Select
    Selection.PrintOut Copies:=1, Preview:=False
    Application.CutCopyMode = False

    'Clears Scratch Print Area
    Range("CS215:DE265").Select
    Selection.ClearContents
    Range("CS211").Select

    Sheets("DATABASE").Select
  Next A

End If

'End of Print Options 4
'====================================

Sheets("DATABASE").Select
Range("I7").Select
ActiveCell.FormulaR1C1 = "1"


GoTo ENDPRTDB:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDPRTDB:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

  With Application
    .Calculation = xlAutomatic
    .MaxChange = 0.001
  End With

  Range("AA8").Select
End Sub




Sub PrintIDpage()
'
' PrintIDpage Macro
' Macro recorded 11/3/2002 by Norman Pillsbury

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

  Range("B1:N38").Select
  With ActiveSheet.PageSetup
    .LeftMargin = Application.InchesToPoints(0.5)
    .RightMargin = Application.InchesToPoints(0#)
    .TopMargin = Application.InchesToPoints(0.75)
    .BottomMargin = Application.InchesToPoints(0.5)
    .HeaderMargin = Application.InchesToPoints(0)
    .FooterMargin = Application.InchesToPoints(0)
    .CenterHorizontally = True
    .Orientation = xlLandscape
    .Zoom = 45
  End With
  Selection.PrintOut Copies:=1, Preview:=False
  Range("A1").Select

  Application.ScreenUpdating = updateMode
  Application.Calculate
  Application.Calculation = calcMode
  Application.Calculation = xlAutomatic

End Sub


Sub PrintOutputPage()
'
' PrintOutputPage Macro
' Macro recorded 11/9/2002 by Norman Pillsbury

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

  Range("Z2:AS65").Select
  ActiveSheet.PageSetup.PrintArea = ""
  With ActiveSheet.PageSetup
    .LeftMargin = Application.InchesToPoints(0.5)
    .RightMargin = Application.InchesToPoints(0.5)
    .TopMargin = Application.InchesToPoints(1.5)
    .BottomMargin = Application.InchesToPoints(0)
    .HeaderMargin = Application.InchesToPoints(0.5)
    .FooterMargin = Application.InchesToPoints(0)
    .CenterHorizontally = True
    .CenterVertically = False
    .Orientation = xlLandscape
    .PaperSize = xlPaperLetter
    .FirstPageNumber = xlAutomatic
    .FitToPagesWide = 1
    .FitToPagesTall = 1
  End With
```

```
    Selection.PrintOut From:=1, To:=1, Copies:=1
    Range("AJ5").Select

Sheets("OUTPUT PARAMETERS").Select
    Range("Z2:AS65").Select
    With ActiveSheet.PageSetup
        .LeftMargin = Application.InchesToPoints(0.5)
        .RightMargin = Application.InchesToPoints(0.5)
        .TopMargin = Application.InchesToPoints(1.5)
        .BottomMargin = Application.InchesToPoints(0)
        .HeaderMargin = Application.InchesToPoints(0.5)
        .FooterMargin = Application.InchesToPoints(0)
        .CenterHorizontally = True
        .Orientation = xlLandscape
        .FitToPagesWide = 1
        .FitToPagesTall = 1
    End With
    Selection.PrintOut Copies:=1, Preview:=False
    Range("Y1").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub




Sub PrintRemovalTreeList()
'
' PrintRemovalTreeList Macro
' Macro recorded 8/29/2002 by Norman Pillsbury

'This macro selects all Removal Trees from DATABASE
and pastes and formats it onto
'Scratch Workspace sheet and prints it from there in
order to retain
'all headings and summary rows.

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

    Sheets("OUTPUT PARAMETERS").Select
    If Range("I7") < 3 Then

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
    With Application
        .Calculation = xlAutomatic
        .MaxChange = 0.001
    End With

    Sheets("OUTPUT PARAMETERS").Select
    Exit Sub
    End If

'==========================
'BEGINNING OF PRINT OPTION 1 (Species & Volume
Data, Columns 1-15 ALL pages).

If Range("I7") = 3 Then

    Sheets("Scratch Workspace").Select
    Range("CA212:CO265").Select
    Selection.Clear
    Range("CA211").Select

    'Prints Page X of Y (uses formats in CN&CO208)
    Range("CN208").Select
    ActiveCell.FormulaR1C1 = 1

    'Select and Paste Count and Volume Sums
    Sheets("DATABASE").Select
    Range("P46:AC48").Select
    Selection.Copy
    Sheets("Scratch Workspace").Select

    Range("CA212").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Application.CutCopyMode = False

    'Copies 1st half of column headings over
    Sheets("DATABASE").Select
    Range("P52:W53").Select
    Application.CutCopyMode = False
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CA216").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False

    'Copies Vol by dia class
    Sheets("DATABASE").Select
    Range("X53:AC53").Select
    Application.CutCopyMode = False
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CI217").Select
```

```
    ActiveSheet.Paste
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False

    'Copies Branch Vol Header
    Sheets("DATABASE").Select
    Range("X52").Select
    Application.CutCopyMode = False
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CI216").Select
    ActiveSheet.Paste

    Sheets("DATABASE").Select
    Range(Cells(54, 16), Cells(103, 29)).Select
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("CA218").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=False
    Application.CutCopyMode = False

        'FIX
        Sheets("Scratch Workspace").Select
        Columns("CA:CA").EntireColumn.AutoFit
        Columns("CB:CB").EntireColumn.AutoFit
        Columns("CC:CC").EntireColumn.AutoFit
        Columns("CD:CD").EntireColumn.AutoFit
        Columns("CE:CE").EntireColumn.AutoFit
        Columns("CF:CF").EntireColumn.AutoFit
        Columns("CG:CG").EntireColumn.AutoFit
        Columns("CH:CH").EntireColumn.AutoFit
        Columns("CI:CI").EntireColumn.AutoFit
        Columns("CJ:CJ").ColumnWidth = 8.43
        Columns("CK:CK").EntireColumn.AutoFit
        Columns("CL:CL").EntireColumn.AutoFit
        Columns("CM:CM").EntireColumn.AutoFit
        Columns("CN:CN").EntireColumn.AutoFit


    Range("CA200:CN267").Select
    With ActiveSheet.PageSetup
        .CenterHeader = "&36&K000000Removal Tree List"
        .LeftMargin = Application.InchesToPoints(0.5)
        .RightMargin = Application.InchesToPoints(0.5)
        .TopMargin = Application.InchesToPoints(0.27)
        .BottomMargin = Application.InchesToPoints(0.56)
        .HeaderMargin = Application.InchesToPoints(0.17)
        .FooterMargin = Application.InchesToPoints(0.56)
        .PrintQuality = -4
       .CenterHorizontally = True
        .CenterVertically = True
        .Orientation = xlLandscape
        .FirstPageNumber = xlAutomatic
        .FitToPagesWide = 1
        .FitToPagesTall = 1
    End With
    Selection.PrintOut Copies:=1, Preview:=False
    Application.CutCopyMode = False

    'Clears Scratch Print Area
    Range("CA213:CN215").Select
    Selection.Delete Shift:=xlUp
    Range("CA210").Select

    Range("CA271:CN271").Select
    Selection.AutoFill Destination:=Range("CA215:
CN271"), Type:=xlFillDefault
    Range("CA215:CN271").Select
    Range("CA210").Select

    Sheets("DATABASE").Select

    For A = 51 To Int(Range("R48")) + 1 Step 50
    Sheets("DATABASE").Select
        Range(Cells(A + 53, 16), Cells(A + 102, 29)).Select
        Selection.Copy
        Sheets("Scratch Workspace").Select

        Range("CN208").Select
        ActiveCell.FormulaR1C1 = Int(A / 50) + 1

        Range("CA215").Select
        Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
            False, Transpose:=False
        Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
            False, Transpose:=False
        Application.CutCopyMode = False
        Range("CA200:CN264").Select

        'FIX
        Sheets("Scratch Workspace").Select
        Columns("CA:CA").EntireColumn.AutoFit
        Columns("CB:CB").EntireColumn.AutoFit
        Columns("CC:CC").EntireColumn.AutoFit
        Columns("CD:CD").EntireColumn.AutoFit
        Columns("CE:CE").EntireColumn.AutoFit
        Columns("CF:CF").EntireColumn.AutoFit
        Columns("CG:CG").EntireColumn.AutoFit
        Columns("CH:CH").EntireColumn.AutoFit
        Columns("CI:CI").EntireColumn.AutoFit
        Columns("CJ:CJ").EntireColumn.AutoFit
        Columns("CK:CK").EntireColumn.AutoFit
        Columns("CL:CL").EntireColumn.AutoFit
```

```
    Columns("CM:CM").EntireColumn.AutoFit
    Columns("CN:CN").EntireColumn.AutoFit

    Selection.PrintOut Copies:=1, Preview:=False
    Application.CutCopyMode = False

    'Clears Scratch Print Area
    Range("CA215:CN265").Select
    Selection.ClearContents
    Range("CA211").Select

  Sheets("OUTPUT PARAMETERS").Select
  Next A


'============================
'END OF PRINT OPTION 1


'============================
'BEGINNING OF PRINT OPTION 2 (Species & Volume
Data, Columns 17-26 ALL pages).
ElseIf Range("I7") = 4 Then

  Sheets("Scratch Workspace").Select
  Range("CS212:DG265").Select
  Selection.Clear
  Range("CS211").Select

  'Prints Page X of Y (uses formats in CN&CO208)
  Range("DD208").Select
  ActiveCell.FormulaR1C1 = 1

  'Total number of pages

  'Select and Paste Count and Volume Sums for Cols 1,
2 & 3
  Sheets("DATABASE").Select
  Range("P46:Q48").Select
  Selection.Copy
  Sheets("Scratch Workspace").Select

  Range("CS212").Select
  Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
    False, Transpose:=False
  Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
    False, Transpose:=False
  Application.CutCopyMode = False

  'Select and Paste Count and Volume Sums for Cols
17-26
  Sheets("DATABASE").Select
  Range("AF46:AO48").Select
  Selection.Copy
  Sheets("Scratch Workspace").Select
  Range("CU212").Select
  Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
    False, Transpose:=False
  Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
    False, Transpose:=False
  Application.CutCopyMode = False

  'Copies 1st 3 column headings over
  Sheets("DATABASE").Select
  Range("P52:Q103").Select
  Application.CutCopyMode = False
  Selection.Copy
  Sheets("Scratch Workspace").Select
  Range("CS216").Select
  Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
    False, Transpose:=False
  Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
    False, Transpose:=False

  'Copies User Variables Headings Over
  Sheets("DATABASE").Select
  Range("AF52:AO103").Select
  Application.CutCopyMode = False
  Selection.Copy
  Sheets("Scratch Workspace").Select
  Range("CU216").Select
  ActiveSheet.Paste
  Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
    False, Transpose:=False
  Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
    False, Transpose:=False
  Application.CutCopyMode = False

  Sheets("Scratch Workspace").Select
  Columns("CS:CS").ColumnWidth = 11.67
  Columns("CT:CT").ColumnWidth = 10.83
  Columns("CU:CU").EntireColumn.AutoFit
  Columns("CV:CV").EntireColumn.AutoFit
  Columns("CW:CW").EntireColumn.AutoFit
  Columns("CX:CX").EntireColumn.AutoFit
  Columns("CY:CY").EntireColumn.AutoFit
  Columns("CZ:CZ").EntireColumn.AutoFit
  Columns("DA:DA").EntireColumn.AutoFit
  Columns("DB:DB").EntireColumn.AutoFit
  Columns("DC:DC").EntireColumn.AutoFit
  Columns("DD:DD").EntireColumn.AutoFit

  Range("CS200:DD263").Select
  With ActiveSheet.PageSetup
    .CenterHeader = "&36&K000000Removal Tree List"
```

```
    .LeftMargin = Application.InchesToPoints(0.5)
    .RightMargin = Application.InchesToPoints(0.5)
    .TopMargin = Application.InchesToPoints(0.27)
    .BottomMargin = Application.InchesToPoints(0.56)
    .HeaderMargin = Application.InchesToPoints(0.17)
    .FooterMargin = Application.InchesToPoints(0.56)
    .PrintQuality = -4
    .Orientation = xlLandscape
    .FirstPageNumber = xlAutomatic
    .FitToPagesWide = 1
    .FitToPagesTall = 1
  End With
  Selection.PrintOut Copies:=1, Preview:=False
  Application.CutCopyMode = False

  'Cuts out Count headings
  Range("CS212:DD215").Select
  Selection.Delete Shift:=xlUp
  Range("CS210").Select

  'Clears Scratch Print Area
  Range("CS214:DE271").Select
  Selection.ClearContents
  Range("CS210").Select

  Sheets("DATABASE").Select

  For A = 51 To Int(Range("R48")) + 1 Step 50
    Sheets("DATABASE").Select
    Range(Cells(A + 53, 16), Cells(A + 102, 17)).Select
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("DD208").Select
    ActiveCell.FormulaR1C1 = Int(A / 50) + 1

    Range("CS214").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False

    Sheets("DATABASE").Select
    Range(Cells(A + 53, 32), Cells(A + 102, 41)).Select
    Selection.Copy

    Sheets("Scratch Workspace").Select
    Range("CU215").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
    Application.CutCopyMode = False

    Sheets("Scratch Workspace").Select
    Columns("CS:CS").ColumnWidth = 11.67
    Columns("CT:CT").ColumnWidth = 10.83
    Columns("CU:CU").ColumnWidth = 12.17
    Columns("CV:CV").EntireColumn.AutoFit
    Columns("CW:CW").EntireColumn.AutoFit
    Columns("CX:CX").EntireColumn.AutoFit
    Columns("CY:CY").EntireColumn.AutoFit
    Columns("CZ:CZ").EntireColumn.AutoFit
    Columns("DA:DA").EntireColumn.AutoFit
    Columns("DB:DB").EntireColumn.AutoFit
    Columns("DC:DC").EntireColumn.AutoFit
    Columns("DD:DD").ColumnWidth = 14.33
    Columns("DE:DE").ColumnWidth = 14.33

    Range("CS200:DD263").Select
    Selection.PrintOut Copies:=1, Preview:=False
    Application.CutCopyMode = False

    'Clears Scratch Print Area
    Range("CS215:DE265").Select
    Selection.ClearContents
    Range("CS211").Select

  Sheets("OUTPUT PARAMETERS").Select
  Next A

End If

'End of Print Options 2
'====================================

Sheets("OUTPUT PARAMETERS").Select
Range("I7").Select
ActiveCell.FormulaR1C1 = "1"

Range("AX3").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

  With Application
    .Calculation = xlAutomatic
    .MaxChange = 0.001
  End With



End Sub
```

```
Sub PrintSortedStockChart()
‘
‘ PrintSortedStockChart Macro
‘ Macro recorded 7/11/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   Range(“CF293:CW332”).Select
   ActiveSheet.PageSetup.PrintArea = “”
   With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(1.35)
      .RightMargin = Application.InchesToPoints(0.75)
      .TopMargin = Application.InchesToPoints(0.75)
      .BottomMargin = Application.InchesToPoints(0.75)
      .HeaderMargin = Application.InchesToPoints(0#)
      .FooterMargin = Application.InchesToPoints(0#)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1
   Range(“CG13”).Select


GoTo ENDPRTSTKCHART:

‘What an error reports via the msgbox
Errhandler:
MsgBox “Error#” & Err & “ : “ & Error(Err)

ENDPRTSTKCHART:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub
```

```
Sub PrintsBoxAColumns1to9()
‘
‘ PrintsBoxAColumns1to9 Macro-- Method 1A
‘
   Range(“H17:Q62”).Select
   ActiveSheet.PageSetup.PrintArea = “”
   With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.5)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(0.5)
      .BottomMargin = Application.InchesToPoints(0.5)
      .HeaderMargin = Application.InchesToPoints(0)
      .FooterMargin = Application.InchesToPoints(0)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .PaperSize = xlPaperLetter
      .BlackAndWhite = False
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1
   Range(“S17”).Select
End Sub
```

```
Sub PrintsBoxBColumns1to9()
‘
‘ PrintsBoxBColumns1to9 Macro -- Method 1B
‘
   Range(“Z17:AH69”).Select
   ActiveSheet.PageSetup.PrintArea = “”
   With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.5)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(0.5)
      .BottomMargin = Application.InchesToPoints(0.5)
      .HeaderMargin = Application.InchesToPoints(0)
      .FooterMargin = Application.InchesToPoints(0)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .PaperSize = xlPaperLetter
      .BlackAndWhite = False
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1
   Range(“AJ42”).Select
End Sub
```

```
Sub PrintSortedStandChart()
‘
‘ PrintSortedStandChart Macro

Dim calcMode, updateMode
calcMode = Application.Calculation
```

```
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Range("V297:AT340").Select
   Selection.PrintOut From:=1, To:=1, Copies:=1
   Range("AE15").Select

   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic

End Sub



Sub PrintSortedStockChart()
'
' PrintSortedStockChart Macro
' Macro recorded 7/11/2002 by Norman Pillsbury

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   Range("CF293:CW332").Select
   ActiveSheet.PageSetup.PrintArea = ""
   With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(1.35)
      .RightMargin = Application.InchesToPoints(0.75)
      .TopMargin = Application.InchesToPoints(0.75)
      .BottomMargin = Application.InchesToPoints(0.75)
      .HeaderMargin = Application.InchesToPoints(0#)
      .FooterMargin = Application.InchesToPoints(0#)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1
   Range("CG13").Select


GoTo ENDPRTSTKCHART:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDPRTSTKCHART:
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub PrintSortedStockTable()
'
' PrintSortedStockTable Macro
' Macro recorded 7/11/2002 by Norman Pillsbury


On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

If Range("CD310") = 0 Then
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic
   Standerror = MsgBox("First Click on Generate
SORTED STOCK TABLE.")
   Exit Sub
End If

   Sheets("Scratch Workspace").Select
   Cells.Select
   Selection.Delete Shift:=xlUp
   Range("F65").Select

   Sheets("MGT OPTIONS").Select
   Columns("BZ:DZ").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Columns("H:H").Select
   ActiveSheet.Paste

   Rows("1:155").Select
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Rows("1:17").Select
   Selection.Insert Shift:=xlDown
   Range("F32").Select
   Sheets("SETUP").Select
   Range("H32:I84").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
```

```
    Range("D28").Select
    ActiveSheet.Paste
    Range("D30:E30").Select
    Application.CutCopyMode = False
    Selection.Delete Shift:=xlUp
    Range("D28:E79").Select
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone

    Range("H22:BH29").Select
    Selection.Copy

    Range("H90:BH90").Select
    Selection.Insert Shift:=xlDown
    Rows("92:94").Select
    Application.CutCopyMode = False
    Selection.Delete Shift:=xlUp
    Range("H94").Select
    Selection.ClearContents
    Rows("90:91").Select
    Selection.Insert Shift:=xlDown
    Range("H96").Select

    Range("H22").Select
    ActiveCell.FormulaR1C1 = _
      "Volume of Trees to be Removed (Sorted by top 20
Species Groups)"
    With ActiveCell.Characters(Start:=1, Length:=0).Font
      .Name = "Arial"
      .FontStyle = "Bold"
      .Size = 12
    End With

    Range("H92").Select
    ActiveCell.FormulaR1C1 = _
      "STOCK TABLE.  Volume of Trees to be Removed
(Sorted by top 20 Species Groups) -- continued."
    With ActiveCell.Characters(Start:=1, Length:=0).Font
      .Name = "Arial"
      .FontStyle = "Bold"
      .Size = 12
    End With

    'Touch-ups
    Range("E28:E79").Select
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("J67").Select
    Range("H89:AD89").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Range("AD28:AD89").Select
    Range("AD89").Activate
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("AE92").Select
    Range("AD95:AD157").Select
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("AG117").Select
    Range("H17").Select

    'Prints out 2 pages of Sorted Stand Table
    Sheets("MGT OPTIONS").Select
    Range("CC305:CD311").Select
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("K6").Select
    ActiveSheet.Paste
    Range("K6").Select

    With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.3)
      .RightMargin = Application.InchesToPoints(0.3)
      .TopMargin = Application.InchesToPoints(0.5)
      .BottomMargin = Application.InchesToPoints(0.5)
      .HeaderMargin = Application.InchesToPoints(0#)
      .FooterMargin = Application.InchesToPoints(0#)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .FitToPagesWide = 1
      .FitToPagesTall = 1
    End With

    If Range("L11") <> 0 Then
      Range("D18:AD89").Select
      Selection.PrintOut Copies:=1
    End If

    If Range("L12") <> 0 Then
      Range("H92:AD157").Select
      Selection.PrintOut Copies:=1
    End If

    Range("AJ92").Select
    Sheets("MGT OPTIONS").Select
    Range("CC16").Select


GoTo ENDSTOCK:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSTOCK:

    Application.ScreenUpdating = updateMode
    Application.Calculate
    Application.Calculation = calcMode
    Application.Calculation = xlAutomatic
    End Sub
```

```
Sub PrintSpeciesList()
‘
‘ PrintSpeciesList Macro
‘ Macro recorded 6/5/02 by NRM


Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With

If Range(“B84”) = “” Then
   Range(“B34:E84”).Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range(“B20”).Select

ElseIf Range(“B134”) = “” Then
   Range(“B34:E134”).Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range(“B20”).Select

ElseIf Range(“B184”) = “” Then Range(“B34:E184”).
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range(“B20”).Select

ElseIf Range(“B234”) = “” Then Range(“B34:E234”).
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range(“B20”).Select

ElseIf Range(“B284”) = “” Then Range(“B34:E284”).
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range(“B20”).Select

ElseIf Range(“B334”) = “” Then Range(“B34:E334”).
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range(“B20”).Select

ElseIf Range(“B384”) = “” Then Range(“B34:E384”).
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range(“B20”).Select

ElseIf Range(“B434”) = “” Then Range(“B34:E434”).
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range(“B20”).Select

ElseIf Range(“B484”) = “” Then Range(“B34:E484”).
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
```

```
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("B20").Select

Else: Range("B34:E540").Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("B20").Select

End If


Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

   Range("B20").Select

End Sub




Sub PrintStandTable()
'
' Macro recorded 7/10/2002 by Norman Pillsbury


On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

If Range("S309") = 0 Then
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic
   Standerror1 = MsgBox("First Click on Generate
STAND TABLE.")
   Exit Sub
End If
   Sheets("Scratch Workspace").Select
   Cells.Select
   Range("A52").Activate
   Selection.Delete Shift:=xlUp
   Range("F65").Select

   Sheets("MGT OPTIONS").Select
   Columns("S:BS").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Columns("H:H").Select
   ActiveSheet.Paste

   Sheets("MGT OPTIONS").Select
   Range("U34:U154").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Range("J34").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("G28").Select

   Rows("2:24").Select
   Range("A24").Activate
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Rows("1:1").Select
   Selection.Delete Shift:=xlUp
   Rows("1:20").Select
   Selection.Insert Shift:=xlDown
   Range("F32").Select
   Sheets("SETUP").Select
   Range("H32:I84").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Range("D29").Select
   Range("D28").Select
   ActiveSheet.Paste
   Range("D30:E30").Select
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Range("D28:E79").Select
   Selection.Borders(xlRight).LineStyle = xlNone
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone

   Range("H22:BH29").Select
   Selection.Copy
   Range("H90:BH90").Select
   Selection.Insert Shift:=xlDown
   Rows("92:94").Select
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Range("H94").Select
   Selection.ClearContents
```

```
  Rows("90:91").Select
  Selection.Insert Shift:=xlDown
  Range("H96").Select

  Columns("J:J").Select
  Range("J23").Activate
  Selection.Copy

  Columns("AJ:AJ").Select
  Range("AJ23").Activate
  Selection.Insert Shift:=xlToRight
  Range("AJ30").Select

  Range("H22").Select
  Application.CutCopyMode = False
  Selection.Copy
  Range("AK22").Select
  ActiveSheet.Paste

  Application.CutCopyMode = False
  ActiveCell.FormulaR1C1 = _
    "STAND TABLE.  Number of Trees to be Removed
(by Dbh Class and by Species Group) -- continued."
  With ActiveCell.Characters(Start:=1, Length:=0).Font
    .Name = "Arial"
    .FontStyle = "Bold"
    .Size = 12
  End With
  With ActiveCell.Characters(Start:=13, Length:=0).Font
    .Name = "Arial"
    .FontStyle = "Regular"
    .Size = 12
  End With
  Range("AK22").Select
  Selection.Copy
  Range("H92").Select
  ActiveSheet.Paste
  Range("AK92").Select
  ActiveSheet.Paste
  Range("AI95").Select
  Application.CutCopyMode = False
  Selection.ClearContents

  Range("AI28").Select
  Selection.ClearContents

  'Touch-ups
  Range("AK92").Select
  Selection.Cut
  Range("AJ92").Select
  ActiveSheet.Paste
  Range("AJ91").Select

  Range("E28:E79").Select
  Selection.Borders(xlRight).LineStyle = xlContinuous
  Range("F27").Select
  Range("AK22").Select
  Selection.Cut
  Range("AJ22").Select
  ActiveSheet.Paste
  Range("AJ28:AJ89").Select
  Selection.Borders(xlLeft).LineStyle = xlContinuous
  Range("AJ89:BI89").Select
  Selection.Borders(xlBottom).LineStyle = xlContinuous
  Range("BI28:BI89").Select
  Range("BI89").Activate
  Selection.Borders(xlRight).LineStyle = xlContinuous
  Range("BI95:BI157").Select
  Selection.Borders(xlRight).LineStyle = xlContinuous
  Range("H19").Select

  Range("H30:AI89").Select
  Selection.Borders(xlLeft).LineStyle = xlNone
  Selection.Borders(xlRight).LineStyle = xlNone
  Selection.Borders(xlTop).LineStyle = xlNone
  Selection.Borders(xlBottom).LineStyle = xlNone

  Selection.Borders(xlEdgeLeft).LineStyle = xlContinu-
ous
  Selection.Borders(xlEdgeTop).LineStyle = xlContinu-
ous
  Selection.Borders(xlEdgeBottom).LineStyle = xlCon-
tinuous
  Selection.Borders(xlEdgeRight).LineStyle = xlContinu-
ous

  Range("AJ95:AJ157").Select
  Selection.Borders(xlLeft).LineStyle = xlContinuous
  Range("AJ93").Select

  Range("AI28:AI29").Select
  With Selection.Borders(xlEdgeRight)
    .LineStyle = xlContinuous
    .Weight = xlThin
  End With

  Range("H89:AI89").Select
  With Selection.Borders(xlEdgeBottom)
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = xlAutomatic
  End With
  Range("AE86").Select

  Sheets("MGT OPTIONS").Select
  Range("S22:S23").Select
  Selection.Copy
  Sheets("Scratch Workspace").Select
  Range("H18").Select
  ActiveSheet.Paste
  Range("H19").Select

  Range("H30:AI89").Select
```

```
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   Selection.Borders(xlInsideVertical).LineStyle = xlNone
   With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = xlAutomatic
   End With
   Range("J30:J89").Select
   Selection.Borders(xlLeft).LineStyle = xlContinuous
   Selection.Borders(xlRight).LineStyle = xlContinuous
   Range("L30:AI89").Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
   End With
   With Selection.Borders(xlInsideVertical)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = xlAutomatic
   End With
   Range("H16").Select

   Columns("D:D").EntireColumn.AutoFit

   'Prints out 4 pages of Stand Table

   Sheets("MGT OPTIONS").Select
   Range("R305:S308").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Range("K6").Select
   ActiveSheet.Paste
   Range("K6").Select

   With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.3)
      .RightMargin = Application.InchesToPoints(0.3)
      .TopMargin = Application.InchesToPoints(0.5)
      .BottomMargin = Application.InchesToPoints(0.5)
      .HeaderMargin = Application.InchesToPoints(0#)
      .FooterMargin = Application.InchesToPoints(0#)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With

   If Range("L6") <> 0 Then
      Range("D18:AI89").Select
      Selection.PrintOut Copies:=1
   End If

   If Range("L8") <> 0 Then
      Range("AJ18:BI89").Select
      Selection.PrintOut Copies:=1
   End If

   If Range("L7") <> 0 Then
      Range("H92:AI157").Select
      Selection.PrintOut Copies:=1
   End If

   If Range("L9") <> 0 Then
      Range("AJ92:BI157").Select
      Selection.PrintOut Copies:=1
   End If

   Range("AJ92").Select
   Sheets("MGT OPTIONS").Select
   Range("X16").Select
```

```
GoTo ENDPRTSTTB:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDPRTSTTB:

   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic

End Sub



Sub PrintStandTableSorted()
'
' Macro recorded 7/10/2002 by Norman Pillsbury


On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

If Range("S310") = 0 Then
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic
   Standerror = MsgBox("First Click on Generate
SORTED STAND TABLE.")
   Exit Sub
End If

   Sheets("Scratch Workspace").Select
   Cells.Select
   Range("A52").Activate
   Selection.Clear
   Range("F65").Select

   Sheets("MGT OPTIONS").Select
   Columns("S:AO").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Columns("H:H").Select
   ActiveSheet.Paste

   Rows("1:155").Select
   'Range("A24").Activate
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Rows("1:17").Select
   Selection.Insert Shift:=xlDown
   Range("F32").Select
   Sheets("SETUP").Select
   Range("H32:I84").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Range("D28").Select
   ActiveSheet.Paste
   Range("D30:E30").Select
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Range("D28:E79").Select
   Selection.Borders(xlRight).LineStyle = xlNone
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone

   Range("H22:BH29").Select
   Selection.Copy

   Range("H90:BH90").Select
   Selection.Insert Shift:=xlDown
   Rows("92:94").Select
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Range("H94").Select
   Selection.ClearContents
   Rows("90:91").Select
   Selection.Insert Shift:=xlDown
   Range("H96").Select

   Range("H22").Select
   ActiveCell.FormulaR1C1 = _
      "Number of Trees to be Removed (Sorted by top 20
Species Groups)"
   With ActiveCell.Characters(Start:=1, Length:=0).Font
      .Name = "Arial"
      .FontStyle = "Bold"
      .Size = 12
   End With

   Range("H92").Select
   ActiveCell.FormulaR1C1 = _
      "STAND TABLE.  Number of Trees to be Removed
(Sorted by top 20 Species Groups) -- continued."
   With ActiveCell.Characters(Start:=1, Length:=0).Font
      .Name = "Arial"
      .FontStyle = "Bold"
      .Size = 12
   End With

   'Touch-ups
   Range("E28:E79").Select
   Selection.Borders(xlRight).LineStyle = xlContinuous
```

```
    Range("J67").Select
    Range("H89:AD89").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Range("AD28:AD89").Select
    Range("AD89").Activate
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("AE92").Select
    Range("AD95:AD157").Select
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("AG117").Select
    Range("H17").Select

    'Prints out 2 pages of Sorted Stand Table
    Sheets("MGT OPTIONS").Select
    Range("R305:S311").Select
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("K6").Select
    ActiveSheet.Paste
    Range("K6").Select

    With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.3)
      .RightMargin = Application.InchesToPoints(0.3)
      .TopMargin = Application.InchesToPoints(0.5)
      .BottomMargin = Application.InchesToPoints(0.5)
      .HeaderMargin = Application.InchesToPoints(0#)
      .FooterMargin = Application.InchesToPoints(0#)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .FitToPagesWide = 1
      .FitToPagesTall = 1
    End With

    If Range("L11") <> 0 Then
      Range("D18:AD89").Select
      Selection.PrintOut Copies:=1
    End If

    If Range("L12") <> 0 Then
      Range("H92:AD157").Select
      Selection.PrintOut Copies:=1
    End If

    Range("AJ92").Select
    Sheets("MGT OPTIONS").Select
    Range("X16").Select


GoTo ENDPRTSTTBSRT:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDPRTSTTBSRT:
    Application.ScreenUpdating = updateMode
    Application.Calculate
    Application.Calculation = calcMode
    Application.Calculation = xlAutomatic

End Sub



Sub PrintStatistics()
'
' PrintStatistics Macro
' Macro recorded 6/8/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

    With ActiveSheet.PageSetup
      .CenterHeader = _
      "&""Arial,Bold Italic""&24Basic Statistics for Tree
Inventory" & vbLf & ""
      .RightHeader = "Page 1"
      .LeftMargin = Application.InchesToPoints(0.5)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(1#)
      .HeaderMargin = Application.InchesToPoints(0.75)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .FitToPagesWide = 1
      .FitToPagesTall = 1
    End With

    Range("H11:R55").Select
    Selection.PrintOut From:=1, To:=1, Copies:=1
    Range("H4").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

    With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
    End With

End Sub



Sub PrintStatSpeciesList()
```

```
‘
‘ PrintStatSpeciesList Macro
‘

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   ‘Create header
   Range(“AR19:AV20”).Select
   Selection.Copy
   Range(“AB19”).Select
   ActiveSheet.Paste
   Application.CutCopyMode = False

   ‘Page 1
   Range(“V71:Z120”).Select
   Selection.Copy
   Range(“AB21”).Select
   ActiveSheet.Paste
   Range(“AC16”).Select
   Columns(“AC:AC”).EntireColumn.AutoFit
   Application.CutCopyMode = False
   Range(“V19:AF70”).Select
   Range(“AF70”).Activate
   With ActiveSheet.PageSetup
      .LeftHeader = “&K000000&D   &T”
      .CenterHeader = “&24&K000000Species List” &
Chr(13) & “”
      .RightHeader = “Page 1”
      .LeftMargin = Application.InchesToPoints(0.75)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(1#)
      .BottomMargin = Application.InchesToPoints(1)
      .HeaderMargin = Application.InchesToPoints(0.75)
      .FooterMargin = Application.InchesToPoints(0.5)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlPortrait
      .PaperSize = xlPaperLetter
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1

   ‘Page 2

   If Range(“Z121”) = “” Then
   GoTo Page3
   End If

   Range(“AB19:AF20”).Select
   Selection.Copy
   Range(“AH19”).Select
   ActiveSheet.Paste
   Application.CutCopyMode = False

   Range(“V121:Z170”).Select
   Selection.Copy
   Range(“AB21”).Select
   ActiveSheet.Paste
   Range(“AG21”).Select
   Columns(“AC:AC”).EntireColumn.AutoFit
   Range(“V171:Z220”).Select
   Application.CutCopyMode = False
   Selection.Copy
   Range(“AH21”).Select
   ActiveSheet.Paste
   Range(“AI13”).Select
   Columns(“AI:AI”).EntireColumn.AutoFit
   Range(“AB19:AL70”).Select
   Application.CutCopyMode = False
   With ActiveSheet.PageSetup
      .LeftHeader = “&K000000&D   &T”
      .CenterHeader = “&24&K000000Species List” &
Chr(13) & “”
      .RightHeader = “Page 2”
      .LeftMargin = Application.InchesToPoints(0.75)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(1.5)
      .BottomMargin = Application.InchesToPoints(1)
      .HeaderMargin = Application.InchesToPoints(0.75)
      .FooterMargin = Application.InchesToPoints(0.5)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlPortrait
      .PaperSize = xlPaperLetter
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1

Page3:
   ‘Page 3

   If Range(“Z221”) = “” Then
   GoTo Page4
   End If

   Range(“AB21:AL70”).Select
   Selection.ClearContents
   Range(“AF72”).Select
   Range(“V221:Z270”).Select
   Selection.Copy
   Range(“AB21”).Select
   ActiveSheet.Paste
   Range(“AC14”).Select
   Columns(“AC:AC”).EntireColumn.AutoFit
   Range(“V271:Z320”).Select
   Application.CutCopyMode = False
   Selection.Copy
   Range(“AH21”).Select
```

```
    ActiveSheet.Paste
    Range("AH16").Select
    Columns("AI:AI").EntireColumn.AutoFit
    Range("AB19:AL70").Select
    Application.CutCopyMode = False
    With ActiveSheet.PageSetup
      .LeftHeader = "&K000000&D   &T"
      .CenterHeader = "&24&K000000Species List" &
Chr(13) & ""
      'RightHeader = "&K000000Page &P"
      .RightHeader = "Page 3"
      .LeftMargin = Application.InchesToPoints(0.75)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(1.5)
      .BottomMargin = Application.InchesToPoints(1)
      .HeaderMargin = Application.InchesToPoints(0.75)
      .FooterMargin = Application.InchesToPoints(0.5)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlPortrait
      .PaperSize = xlPaperLetter
      .FitToPagesWide = 1
      .FitToPagesTall = 1
    End With
    Selection.PrintOut From:=1, To:=1, Copies:=1

Page4:
    'Page 4

    If Range("Z321") = "" Then
    GoTo Page5
    End If

    Range("AB21:AL75").Select
    Selection.ClearContents
    Range("AF72").Select
    Range("V321:Z370").Select
    Selection.Copy
    Range("AB21").Select
    ActiveSheet.Paste
    Range("AD14").Select
    Columns("AC:AC").EntireColumn.AutoFit
    Range("V371:Z420").Select
    Application.CutCopyMode = False
    Selection.Copy
    Range("AH21").Select
    ActiveSheet.Paste
    Range("AJ16").Select
    Columns("AI:AI").EntireColumn.AutoFit
    Range("AB19:AL70").Select
    Application.CutCopyMode = False
    With ActiveSheet.PageSetup
      .LeftHeader = "&K000000&D   &T"
      .CenterHeader = "&24&K000000Species List" &
Chr(13) & ""
      'RightHeader = "&K000000Page &P"
      .RightHeader = "Page 4"
      .LeftMargin = Application.InchesToPoints(0.75)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(1.5)
      .BottomMargin = Application.InchesToPoints(1)
      .HeaderMargin = Application.InchesToPoints(0.75)
      .FooterMargin = Application.InchesToPoints(0.5)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlPortrait
      .PaperSize = xlPaperLetter
      .FitToPagesWide = 1
      .FitToPagesTall = 1
    End With
    Selection.PrintOut From:=1, To:=1, Copies:=1

Page5:
    'Page 5

    If Range("Z421") = "" Then
    GoTo EndPrint
    End If

    Range("AB21:AL75").Select
    Selection.ClearContents
    Range("AM68").Select
    Range("V421:Z470").Select
    Selection.Copy
    Range("AB21").Select
    ActiveSheet.Paste
    Range("AF18").Select
    Columns("AC:AC").EntireColumn.AutoFit
    Range("V471:Z525").Select
    Application.CutCopyMode = False
    Selection.Copy
    Range("AH21").Select
    ActiveSheet.Paste
    Range("AJ14").Select
    Columns("AI:AI").EntireColumn.AutoFit
    Range("AB21:AL75").Select
    Application.CutCopyMode = False
    With ActiveSheet.PageSetup
      .LeftHeader = "&K000000&D   &T"
      .CenterHeader = "&24&K000000Species List" &
Chr(13) & ""
      .RightHeader = "Page 5"
      .LeftMargin = Application.InchesToPoints(0.75)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(1.5)
      .BottomMargin = Application.InchesToPoints(1)
      .HeaderMargin = Application.InchesToPoints(0.75)
      .FooterMargin = Application.InchesToPoints(0.5)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlPortrait
      .PaperSize = xlPaperLetter
      .FitToPagesWide = 1
      .FitToPagesTall = 1
```

```
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1

EndPrint:

   Range("AB17:AL17").Select
   Selection.AutoFill Destination:=Range("AB17:AL88"),
Type:=xlFillDefault
   Range("AB17:AL88").Select
   Range("X11").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With
End Sub



Sub PrintStckTableOnOutputParametersSheet()
'
' PrintStckTableOnOutputParametersSheet Macro

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Range("AX70:BW127").Select
   ActiveSheet.PageSetup.PrintArea = ""
   With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.5)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(0.25)
      .BottomMargin = Application.InchesToPoints(0.25)
      .HeaderMargin = Application.InchesToPoints(0.5)
      .FooterMargin = Application.InchesToPoints(0)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1

   Range("BX70:CY127").Select
   Selection.PrintOut Copies:=1
   Range("AV6").Select

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub PrintStndTableOnOutputParametersSheet()
'
' PrintStndTableOnOutputParametersSheet Macro

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Range("AX7:BW64").Select
   ActiveSheet.PageSetup.PrintArea = ""
   With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.5)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(0.25)
      .BottomMargin = Application.InchesToPoints(0.25)
      .HeaderMargin = Application.InchesToPoints(0.5)
      .FooterMargin = Application.InchesToPoints(0)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlLandscape
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1

   Range("BX7:CY64").Select
   Selection.PrintOut Copies:=1
   Range("AV6").Select

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub PrintStockTable()
'
' PrintStockTable Macro
' Macro recorded 7/11/2002 by Norman Pillsbury


On Error GoTo Errhandler
```

```
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

If Range("CD309") = 0 Then
   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic
   Standerror1 = MsgBox("First Click on Generate
STOCK TABLE.")
   Exit Sub
End If

   Sheets("Scratch Workspace").Select
   Cells.Select
   Selection.Delete Shift:=xlUp
   Range("F65").Select

   Sheets("MGT OPTIONS").Select
   Columns("BZ:DZ").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Columns("H:H").Select
   ActiveSheet.Paste

   Sheets("MGT OPTIONS").Select
   Range("CB34:CB154").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Range("J34").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("G28").Select

   Rows("2:24").Select
   Range("A24").Activate
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Rows("1:1").Select
   Selection.Delete Shift:=xlUp
   Rows("1:20").Select
   Selection.Insert Shift:=xlDown
   Range("F32").Select
   Sheets("SETUP").Select
   Range("H32:I84").Select
   Selection.Copy
   Sheets("Scratch Workspace").Select
   Range("D28").Select
   ActiveSheet.Paste
   Range("D30:E30").Select
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Range("D28:E79").Select
   Selection.Borders(xlRight).LineStyle = xlNone
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone

   Range("H22:BH29").Select
   Selection.Copy
   Range("H90:BH90").Select
   Selection.Insert Shift:=xlDown
   Rows("92:94").Select
   Application.CutCopyMode = False
   Selection.Delete Shift:=xlUp
   Range("H94").Select
   Selection.ClearContents
   Rows("90:91").Select
   Selection.Insert Shift:=xlDown
   Range("H96").Select

   Columns("AJ:AJ").Select
   Range("AJ23").Activate
   Selection.Insert Shift:=xlToRight
   Range("AJ30").Select

   Range("H22").Select
   Application.CutCopyMode = False
   Selection.Copy
   Range("AK22").Select
   ActiveSheet.Paste

   Application.CutCopyMode = False
   ActiveCell.FormulaR1C1 = _
      "STOCK TABLE.  Volume of Trees to be Removed
(by Dbh Class and by Species Group) -- continued."
   With ActiveCell.Characters(Start:=1, Length:=0).Font
      .Name = "Arial"
      .FontStyle = "Bold"
      .Size = 12
   End With
   With ActiveCell.Characters(Start:=13, Length:=0).Font
      .Name = "Arial"
      .FontStyle = "Regular"
      .Size = 12
   End With
   Range("AK22").Select
   Selection.Copy
   Range("H92").Select
   ActiveSheet.Paste
   Range("AK92").Select
   ActiveSheet.Paste
   Range("AI95").Select
   Application.CutCopyMode = False
   Selection.ClearContents

   Range("AI28").Select
   Selection.ClearContents

   'Touch-ups
```

```
    Range("AK92").Select
    Selection.Cut
    Range("AJ92").Select
    ActiveSheet.Paste
    Range("AJ91").Select

    Range("E28:E79").Select
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("F27").Select

    Range("AK22").Select
    Selection.Cut
    Range("AJ22").Select
    ActiveSheet.Paste
    Range("AJ28:AJ89").Select
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Range("AJ89:BI89").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    Range("BI28:BI89").Select
    Range("BI89").Activate
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("BI95:BI157").Select
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("H19").Select

    Range("H30:AI89").Select
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone

    Selection.Borders(xlEdgeLeft).LineStyle = xlContinu-
ous
    Selection.Borders(xlEdgeTop).LineStyle = xlContinu-
ous
    Selection.Borders(xlEdgeBottom).LineStyle = xlCon-
tinuous
    Selection.Borders(xlEdgeRight).LineStyle = xlContinu-
ous

    Range("AJ95:AJ157").Select
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Range("AJ93").Select

    Range("AI28:AI29").Select
    With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
    End With

    Range("H89:AI89").Select
    With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
    End With
    Range("AE86").Select
    Sheets("MGT OPTIONS").Select
    Application.CutCopyMode = False
    Range("BZ22:BZ23").Select
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("H18").Select
    ActiveSheet.Paste
    Range("BL19").Select

    Range("H30:AI89").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeRight)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = xlAutomatic
    End With
    Range("J30:J89").Select
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Range("L30:AI88").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = xlAutomatic
    End With
```

```
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlInsideVertical)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = xlAutomatic
    End With
    Range("M18").Select

    Columns("F:G").Select
    Selection.ColumnWidth = 1
    Columns("D:D").EntireColumn.AutoFit

    Range("J28:J291").Select
    Selection.Copy
    Range("AJ28").Select
    ActiveSheet.Paste
    Range("AJ23:AJ291").Select
    Application.CutCopyMode = False
    Selection.EntireColumn.Insert
    Range("AJ7").Select
    Columns("AJ:AJ").ColumnWidth = 48#
    Range("AJ21").Select
    Selection.AutoFill Destination:=Range("AJ21:AJ293"),
Type:=xlFillDefault
    Range("AJ21:AJ293").Select
    Range("AJ24:AJ25").Select
    Selection.AutoFill Destination:=Range("AJ24:AK25"),
Type:=xlFillDefault
    Range("AJ24:AK25").Select
    Range("AJ22").Select

    'Prints out 4 pages of Stand Table

    Sheets("MGT OPTIONS").Select
    Range("CC305:CD308").Select
    Selection.Copy
    Sheets("Scratch Workspace").Select
    Range("K6").Select
    ActiveSheet.Paste
    Range("K6").Select

    With ActiveSheet.PageSetup
        .LeftMargin = Application.InchesToPoints(0.3)
        .RightMargin = Application.InchesToPoints(0.3)
        .TopMargin = Application.InchesToPoints(0.5)
        .BottomMargin = Application.InchesToPoints(0.5)
        .HeaderMargin = Application.InchesToPoints(0#)
        .FooterMargin = Application.InchesToPoints(0#)
        .CenterHorizontally = True
        .CenterVertically = True
        .Orientation = xlLandscape
        .FirstPageNumber = xlAutomatic
        .FitToPagesWide = 1
        .FitToPagesTall = 1
    End With

    If Range("L6") <> 0 Then
        Range("D18:AI89").Select
        Selection.PrintOut From:=1, To:=1, Copies:=1
        Range("D17").Select
    End If


    If Range("L8") <> 0 Then
        Range("AJ18:BI89").Select
        Selection.PrintOut Copies:=1
    End If

    If Range("L7") <> 0 Then
        Range("H92:AI157").Select
        Selection.PrintOut Copies:=1
    End If

    If Range("L9") <> 0 Then
        Range("AJ92:BI157").Select
        Selection.PrintOut Copies:=1
    End If

    Range("AJ92").Select
    Sheets("MGT OPTIONS").Select
    Range("CG12").Select


    GoTo ENDPRTSTKTABL:

    'What an error reports via the msgbox
    Errhandler:
    MsgBox "Error#" & Err & " : " & Error(Err)

    ENDPRTSTKTABL:

        Application.ScreenUpdating = updateMode
        Application.Calculate
        Application.Calculation = calcMode
        Application.Calculation = xlAutomatic
    End Sub



    Sub PrintValuesMeth1()
    '
    ' PrintValuesMeth1 Macro
    ' Macro recorded 8/1/2002 by Norman Pillsbury
```

```
On Error GoTo Errhandler

'Problably can delete all repeats in this macro

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With

If Range("M126") = "" Then
   Range("M20:R126").Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("O22").Select

ElseIf Range("M176") = "" Then
   Range("M20:R176").Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("O22").Select

ElseIf Range("M226") = "" Then Range("M20:R226").
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("O22").Select

ElseIf Range("M276") = "" Then Range("M20:R276").
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("O22").Select

ElseIf Range("M326") = "" Then Range("M20:R326").
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("O22").Select

ElseIf Range("M376") = "" Then Range("M20:R376").
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("O22").Select

ElseIf Range("M426") = "" Then Range("M20:R426").
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("O22").Select

ElseIf Range("M476") = "" Then Range("M20:R476").
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
   End With
   Range("O22").Select

ElseIf Range("M531") = "" Then Range("M20:R531").
Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlPortrait
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = False
```

```
   End With
   Range("O22").Select

End If


GoTo ENDPRTMETH1:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDPRTMETH1:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

End Sub



Sub PrintValuesMeth1a()
'
' PrintValuesMeth1 Macro
' Macro recorded 8/1/2002 by Norman Pillsbury

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   Range("M20:R66").Select
   ActiveSheet.PageSetup.PrintArea = ""
   With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.75)
      .RightMargin = Application.InchesToPoints(0.75)
      .TopMargin = Application.InchesToPoints(1)
      .BottomMargin = Application.InchesToPoints(1)
      .HeaderMargin = Application.InchesToPoints(0.5)
      .FooterMargin = Application.InchesToPoints(0.5)
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlPortrait
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut From:=1, To:=1, Copies:=1

   Range("L1").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

End Sub



Sub PrintValuesMeth2()
'
' PrintValuesMeth2 Macro
' Macro recorded 8/1/2002 by Norman Pillsbury
'
Sheets("VALUATION").Select

   With ActiveSheet.PageSetup
      .Orientation = xlLandscape
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

If Range("AI24") = "Total" Then
   Range("W20:AI76").Select

   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlLandscape
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With

Else: Range("W20:AM76").Select
   Selection.PrintOut Copies:=1, Preview:=False
   With ActiveSheet.PageSetup
      .Orientation = xlLandscape
      .FirstPageNumber = xlAutomatic
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
End If

Range("W9").Select
```

```
Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

End Sub



Sub PrintVolEqnList()
'
' PrintVolEqnList Macro
' Macro recorded 8/21/2002 by Norman Pillsbury

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   Range("H30:U84").Select
   With ActiveSheet.PageSetup
      .LeftMargin = Application.InchesToPoints(0.5)
      .RightMargin = Application.InchesToPoints(0.5)
      .TopMargin = Application.InchesToPoints(1)
      .BottomMargin = Application.InchesToPoints(1)
      .HeaderMargin = Application.InchesToPoints(0.5)
      .FooterMargin = Application.InchesToPoints(0.5)
      .PrintHeadings = False
      .PrintGridlines = False
      .PrintComments = xlPrintNoComments
      .CenterHorizontally = True
      .CenterVertically = True
      .Orientation = xlPortrait
      .PaperSize = xlPaperLetter
      .Zoom = False
      .FitToPagesWide = 1
      .FitToPagesTall = 1
   End With
   Selection.PrintOut Copies:=1
   Range("H26").Select

   Application.ScreenUpdating = updateMode
   Application.Calculate
   Application.Calculation = calcMode
   Application.Calculation = xlAutomatic

End Sub
```

```
Public Sub Random()
'
'Run the Error handler "ErrHandler" when an error oc-
curs
On Error GoTo Errhandler



Call GoToParameters
Call Seg1   '<<-----
Call Seg2   '<<-----
Application.Run ("DoItRandom")
Randomize    'Forces random number generator to start
with different seed each time it is run.

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Application.Run "ClearDM"

Sheets("REMOVAL").Select
D1S = Range("AX26"): D1E = Range("AY26")     'Tree
SEQ Num for Start and End of dbh class no. 1
D2S = Range("AX27"): D2E = Range("AY27")
D3S = Range("AX28"): D3E = Range("AY28")
D4S = Range("AX29"): D4E = Range("AY29")
D5S = Range("AX30"): D5E = Range("AY30")
D6S = Range("AX31"): D6E = Range("AY31")
D7S = Range("AX32"): D7E = Range("AY32")
D8S = Range("AX33"): D8E = Range("AY33")
D9S = Range("AX34"): D9E = Range("AY34")
D10S = Range("AX35"): D10E = Range("AY35")
D11S = Range("AX36"): D11E = Range("AY36")
D12S = Range("AX37"): D12E = Range("AY37")
D13S = Range("AX38"): D13E = Range("AY38")
D14S = Range("AX39"): D14E = Range("AY39")
D15S = Range("AX40"): D15E = Range("AY40")    'Tree
SEQ Num for Start and End of dbh class no. 15
'================================
R1S = Range("BF26"): R1E = Range("BG26")     'Start
and End for Removal of trees in dbh class no. 1
R2S = Range("BF27"): R2E = Range("BG27")
R3S = Range("BF28"): R3E = Range("BG28")
R4S = Range("BF29"): R4E = Range("BG29")
R5S = Range("BF30"): R5E = Range("BG30")
R6S = Range("BF31"): R6E = Range("BG31")
R7S = Range("BF32"): R7E = Range("BG32")
R8S = Range("BF33"): R8E = Range("BG33")
R9S = Range("BF34"): R9E = Range("BG34")
R10S = Range("BF35"): R10E = Range("BG35")
R11S = Range("BF36"): R11E = Range("BG36")
R12S = Range("BF37"): R12E = Range("BG37")
R13S = Range("BF38"): R13E = Range("BG38")
R14S = Range("BF39"): R14E = Range("BG39")
```

```
R15S = Range("BF40"): R15E = Range("BG40")    'Start and End for Removal of trees in dbh class no. 15
T = 3: S = 10000: R = 30000
'D9 = Range("BL53")       'Number of Trees to be Removed, based on Box A or Box B
Dim Data#(30000, 1)     'Data Matrix for Removal Trees (from Random number)--NEED TO MAKE A VARIABLE
Dim B#(10000, 1)          'Final Data array to be printed in Col DA. Initially dim to 5000
'===
  Sheets("DATABASE").Select
'====
  'Inputs random number into Data matrix for 6" trees to be removed
  'R1S =1, first number of removal
  'R1E = 109, last number of removal
  'D1S = 2, Tree SEQ Num for 1st Start for 6" dbh class
  'D1E = n, Tree SEQ Num for last tree for 6" dbh class
'====
'Inputs random number into Data matrix for trees to be removed in dbh class 1 (6")
'T * R1E provides 3 times the no. of random numbers from which to choose from

h = 0
If R1S <> 0 Then
  For A = R1S To T * R1E + R1S
    Data(A, 1) = Int(D1S + Rnd * (D1E - D1S + 1))
'takes a fraction 0.01 to 0.99 of total no of trees in 1st dbh class which is the tree seq number.
  'True means Box B is selected;  40,50 = "" means no % chance of removal is desired & random number is removed.
  'If Box A is checked, this section will always pass.
    If Cells(40, 51) = "True" And Cells(40, 50) = "" Then
      Data(A, 1) = 0
    End If
  'This removes random no's based on restrictions in Step 7 for <lower limit and >upper limit
    If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
      Data(A, 1) = 0
    End If
    '-----
    'Exclusions
    If Cells(75, 54) > 0 Then
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24, 54) Then
          If A > 1 Then
            A = A - 1: C = 50
          End If
        End If
      Next C
    End If
    '---- Inclusions
    If Cells(75, 55) > 0 Then
      G = 0
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24, 55) Then
          C = 50: G = 1
        End If
      Next C
        If G <> 1 Then
          A = A - 1
        End If
    End If
    '----
    h = h + 1
    If h = Cells(21, 50) Then      'Total no. of trees/dbh class
      A = T * R1E + R1S          'If all rnd no's are calculated in dbh class #1, sets A to max number so it'll quit.
    End If
  Next A

  Application.Run "ClearCopy"

  'Prints random trees in 1st dbh class
  Range("DM38").Select
  For A = R1S To T * R1E + R1S
    If Data(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = Data(A, 1)
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  'Blanks out duplicates (1st of a pair of duplicates).  For A = 1 (trust me)
  Application.Run "BlanksOutDuplicates"
  'Reads unsorted random numbers back into array
  Range("DM38").Select
  For A = 1 To R1E - R1S + 1
    If Cells(A + 38, 117) <> "" Then
      B(A, 1) = Cells(A + 38, 117)
    End If
    ActiveCell.Offset(1, 0).Select
  Next A
  'Prints out final list to be checked
  Range("DA36").Select
  Selection.End(xlDown).Select
  ActiveCell.Offset(1, 0).Select
  For A = 1 To R1E - R1S + 1
    If B(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = B(A, 1)
    End If
    If B(A, 1) = 0 Then
      A = R1E - R1S + 1
    End If
      ActiveCell.Offset(1, 0).Select
  Next A
```

```
   For A = 1 To R
      Data(A, 1) = 0
   Next A
   For A = 1 To S
      B(A, 1) = 0
   Next A
End If

'Inputs random number into Data matrix for trees to be
removed in dbh class 2 (12")
'Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0
If R2S <> 0 Then

   For A = R2S To T * R2E + R2S
      Data(A, 1) = Int(D2S + Rnd * (D2E - D2S + 1))
      If Cells(40, 51) = "True" And Cells(41, 50) = "" Then
         Data(A, 1) = 0
      End If
      If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
         Data(A, 1) = 0
      End If
      '-----
      If Cells(75, 54) > 0 Then
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
               If A > 1 Then
                  A = A - 1: C = 50
               End If
            End If
         Next C
      End If
      '-----
      If Cells(75, 55) > 0 Then
         G = 0
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
               C = 50: G = 1
            End If
         Next C
            If G <> 1 Then
               A = A - 1
            End If
      End If
      '---
      h = h + 1
      If h = Cells(22, 50) Then     'Cells 22,50 is number of
trees in dbh class
         A = T * R2E + R2S
      End If
   Next A

   Application.Run "ClearCopy"

   'Prints random trees in this dbh class
   Range("DM38").Select
   For A = R2S To T * R2E + R2S
      If Data(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = Data(A, 1)
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   Application.Run "BlanksOutDuplicates"
   'Reads unsorted random numbers back into array
   Range("DM38").Select
   For A = 1 To R2E - R2S + 1
      If Cells(A + 38, 117) <> "" Then
         B(A, 1) = Cells(A + 38, 117)
      End If
      ActiveCell.Offset(1, 0).Select
   Next A

   'Prints out final list to be checked
   Range("DA36").Select
   Selection.End(xlDown).Select
   ActiveCell.Offset(1, 0).Select
   For A = 1 To R2E - R2S + 1
      If B(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = B(A, 1)
      End If
      If B(A, 1) = 0 Then
         A = R2E - R2S + 1
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   For A = 1 To R
      Data(A, 1) = 0
   Next A
   For A = 1 To S
      B(A, 1) = 0
   Next A
End If

'----
'Inputs random number into Data matrix for trees to be
removed in dbh class 3 (18")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R3S <> 0 Then
   For A = R3S To T * R3E + R3S
      Data(A, 1) = Int(D3S + Rnd * (D3E - D3S + 1))
      If Cells(40, 51) = "True" And Cells(42, 50) = "" Then
         Data(A, 1) = 0
      End If
      If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
```

```
        Data(A, 1) = 0
      End If
      '-----
      If Cells(75, 54) > 0 Then
        For C = 1 To 50
          If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
            If A > 1 Then
              A = A - 1: C = 50
            End If
          End If
        Next C
      End If
      '-----
      If Cells(75, 55) > 0 Then
        G = 0
        For C = 1 To 50
          If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
            C = 50: G = 1
          End If
        Next C
          If G <> 1 Then
            A = A - 1
          End If
      End If
      '-------
      h = h + 1
      If h = Cells(23, 50) Then
        A = T * R3E + R3S
      End If
    Next A

    Application.Run "ClearCopy"

      'Prints random trees in this dbh class
    Range("DM38").Select
    For A = R3S To T * R3E + R3S
      If Data(A, 1) <> 0 Then
        ActiveCell.FormulaR1C1 = Data(A, 1)
      End If
        ActiveCell.Offset(1, 0).Select
    Next A

      Application.Run "BlanksOutDuplicates"
    'Reads unsorted random numbers back into array
    Range("DM38").Select
    For A = 1 To R3E - R3S + 1
      If Cells(A + 38, 117) <> "" Then
        B(A, 1) = Cells(A + 38, 117)
      End If
      ActiveCell.Offset(1, 0).Select
    Next A
    'Prints out final list to be checked
    Range("DA36").Select
    Selection.End(xlDown).Select
    ActiveCell.Offset(1, 0).Select
    For A = 1 To R3E - R3S + 1
      If B(A, 1) <> 0 Then
        ActiveCell.FormulaR1C1 = B(A, 1)
      End If
      If B(A, 1) = 0 Then
        A = R3E - R3S + 1
      End If
        ActiveCell.Offset(1, 0).Select
    Next A

    For A = 1 To R
      Data(A, 1) = 0
    Next A
    For A = 1 To S
      B(A, 1) = 0
    Next A
  End If
  '------
  'Inputs random number into Data matrix for trees to be
removed in dbh class 4 (24")
  Sheets("DATABASE").Select
  Application.Run "SortDBH"
  h = 0

  If R4S <> 0 Then
    For A = R4S To T * R4E + R4S
      Data(A, 1) = Int(D4S + Rnd * (D4E - D4S + 1))
      If Cells(40, 51) = "True" And Cells(43, 50) = "" Then
        Data(A, 1) = 0
      End If
      If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
        Data(A, 1) = 0
      End If
      '------
      If Cells(75, 54) > 0 Then
        For C = 1 To 50
          If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
            If A > 1 Then
              A = A - 1: C = 50
            End If
          End If
        Next C
      End If
      '------
      If Cells(75, 55) > 0 Then
        G = 0
        For C = 1 To 50
          If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
            C = 50: G = 1
          End If
        Next C
          If G <> 1 Then
            A = A - 1
          End If
```

```
    End If
    '-----
    h = h + 1
    If h = Cells(24, 50) Then
      A = T * R4E + R4S
    End If
  Next A

  Application.Run "ClearCopy"

  'Prints random trees in this dbh class
  Range("DM38").Select
  For A = R4S To T * R4E + R4S
    If Data(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = Data(A, 1)
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

    Application.Run "BlanksOutDuplicates"
  'Reads unsorted random numbers back into array
  Range("DM38").Select
  For A = 1 To R4E - R4S + 1
    If Cells(A + 38, 117) <> "" Then
      B(A, 1) = Cells(A + 38, 117)
    End If
    ActiveCell.Offset(1, 0).Select
  Next A
  'Prints out final list to be checked
  Range("DA36").Select
  Selection.End(xlDown).Select
  ActiveCell.Offset(1, 0).Select

  For A = 1 To R4E - R4S + 1
    If B(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = B(A, 1)
    End If
    If B(A, 1) = 0 Then
      A = R4E - R4S + 1
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  For A = 1 To R
    Data(A, 1) = 0
  Next A
  For A = 1 To S
    B(A, 1) = 0
  Next A
End If
'------
'Inputs random number into Data matrix for trees to be
removed in dbh class 5 (30")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0
If R5S <> 0 Then
  For A = R5S To T * R5E + R5S
    Data(A, 1) = Int(D5S + Rnd * (D5E - D5S + 1))
    If Cells(40, 51) = "True" And Cells(44, 50) = "" Then
      Data(A, 1) = 0
    End If
    If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
      Data(A, 1) = 0
    End If
    '-------------------------------------------------------
------------
    If Cells(75, 54) > 0 Then
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
          If A > 1 Then
            A = A - 1: C = 50
          End If
        End If
      Next C
    End If
    '-------------------------------------------------------
------------
    If Cells(75, 55) > 0 Then
      G = 0
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
          C = 50: G = 1
        End If
      Next C
        If G <> 1 Then
          A = A - 1
        End If
    End If
    '-------------------------------------------------------
------------
    h = h + 1
    If h = Cells(25, 50) Then
      A = T * R5E + R5S
    End If
  Next A

  Application.Run "ClearCopy"

  'Prints random trees in this dbh class
  Range("DM38").Select
  For A = R5S To T * R5E + R5S
    If Data(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = Data(A, 1)
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  Application.Run "BlanksOutDuplicates"
  'Reads unsorted random numbers back into array
```

```
   Range("DM38").Select
   For A = 1 To R5E - R5S + 1
      If Cells(A + 38, 117) <> "" Then
         B(A, 1) = Cells(A + 38, 117)
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
   'Prints out final list to be checked
   Range("DA36").Select
   Selection.End(xlDown).Select
   ActiveCell.Offset(1, 0).Select

   For A = 1 To R5E - R5S + 1
      If B(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = B(A, 1)
      End If
      If B(A, 1) = 0 Then
         A = R5E - R5S + 1
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   For A = 1 To R
      Data(A, 1) = 0
   Next A
   For A = 1 To S
      B(A, 1) = 0
   Next A
End If
'----
'Inputs random number into Data matrix for trees to be
removed in dbh class 6 (36")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R6S <> 0 Then
   For A = R6S To T * R6E + R6S
      Data(A, 1) = Int(D6S + Rnd * (D6E - D6S + 1))
      If Cells(40, 51) = "True" And Cells(45, 50) = "" Then
         Data(A, 1) = 0
      End If
      If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
         Data(A, 1) = 0
      End If
      '----
      If Cells(75, 54) > 0 Then
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
               If A > 1 Then
                  A = A - 1: C = 50
               End If
            End If
         Next C
      End If
      '----
      If Cells(75, 55) > 0 Then
         G = 0
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
               C = 50: G = 1
            End If
         Next C
            If G <> 1 Then
               A = A - 1
            End If
      End If
      '----
      h = h + 1
      If h = Cells(26, 50) Then
         A = T * R6E + R6S
      End If
   Next A

   Application.Run "ClearCopy"

   'Prints random trees in this dbh class
   Range("DM38").Select
   For A = R6S To T * R6E + R6S
      If Data(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = Data(A, 1)
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   Application.Run "BlanksOutDuplicates"
   'Reads unsorted random numbers back into array
   Range("DM38").Select
   For A = 1 To R6E - R6S + 1
      If Cells(A + 38, 117) <> "" Then
         B(A, 1) = Cells(A + 38, 117)
      End If
      ActiveCell.Offset(1, 0).Select
   Next A

   'Prints out final list to be checked
   Range("DA36").Select
   Selection.End(xlDown).Select
   ActiveCell.Offset(1, 0).Select

   For A = 1 To R6E - R6S + 1
      If B(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = B(A, 1)
      End If
      If B(A, 1) = 0 Then
         A = R6E - R6S + 1
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   For A = 1 To R
```

```
    Data(A, 1) = 0
  Next A
  For A = 1 To S
    B(A, 1) = 0
  Next A
End If
'-----
'Inputs random number into Data matrix for trees to be
removed in dbh class 7 (42").  T*10 from here on.
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R7S <> 0 Then
  For A = R7S To T * 10 * R7E + R7S
    Data(A, 1) = Int(D7S + Rnd * (D7E - D7S + 1))
    If Cells(40, 51) = "True" And Cells(46, 50) = "" Then
      Data(A, 1) = 0
    End If
    If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
      Data(A, 1) = 0
    End If
    '----
    If Cells(75, 54) > 0 Then
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
          If A > 1 Then
            A = A - 1: C = 50
          End If
        End If
      Next C
    End If
    '----
    If Cells(75, 55) > 0 Then
      G = 0
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
          C = 50: G = 1
        End If
      Next C
        If G <> 1 Then
          A = A - 1
        End If
    End If
    '----
    h = h + 1
    If h = Cells(27, 50) Then
      A = T * 10 * R7E + R7S
    End If
  Next A

  Application.Run "ClearCopy"

  'Prints random trees in this dbh class
  Range("DM38").Select
  For A = R7S To T * R7E + R7S
    If Data(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = Data(A, 1)
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  Application.Run "BlanksOutDuplicates"
  'Reads unsorted random numbers back into array
  Range("DM38").Select
  For A = 1 To R7E - R7S + 1
    If Cells(A + 38, 117) <> "" Then
      B(A, 1) = Cells(A + 38, 117)
    End If
    ActiveCell.Offset(1, 0).Select
  Next A

  'Prints out final list to be checked
  Range("DA36").Select
  Selection.End(xlDown).Select
  ActiveCell.Offset(1, 0).Select

  For A = 1 To R7E - R7S + 1
    If B(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = B(A, 1)
    End If
    If B(A, 1) = 0 Then
      A = R7E - R7S + 1
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  For A = 1 To R
    Data(A, 1) = 0
  Next A
  For A = 1 To S
    B(A, 1) = 0
  Next A
End If
'----
'Inputs random number into Data matrix for trees to be
removed in dbh class 8 (48")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R8S <> 0 Then
  For A = R8S To T * 10 * R8E + R8S
    Data(A, 1) = Int(D8S + Rnd * (D8E - D8S + 1))
    If Cells(40, 51) = "True" And Cells(47, 50) = "" Then
      Data(A, 1) = 0
    End If
    If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
      Data(A, 1) = 0
    End If
```

```
    '----
    If Cells(75, 54) > 0 Then
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
          If A > 1 Then
            A = A - 1: C = 50
          End If
        End If
      Next C
    End If
    '-----
    If Cells(75, 55) > 0 Then
      G = 0
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
          C = 50: G = 1
        End If
      Next C
        If G <> 1 Then
          A = A - 1
        End If
    End If
    '-----
    h = h + 1
    If h = Cells(28, 50) Then
      A = T * 10 * R8E + R8S
    End If
  Next A

  Application.Run "ClearCopy"

  'Prints random trees in this dbh class
  Range("DM38").Select
  For A = R8S To T * R8E + R8S
    If Data(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = Data(A, 1)
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  Application.Run "BlanksOutDuplicates"
  'Reads unsorted random numbers back into array
  Range("DM38").Select
  For A = 1 To R8E - R8S + 1
    If Cells(A + 38, 117) <> "" Then
      B(A, 1) = Cells(A + 38, 117)
    End If
    ActiveCell.Offset(1, 0).Select
  Next A
  'Prints out final list to be checked
  Range("DA36").Select
  Selection.End(xlDown).Select
  ActiveCell.Offset(1, 0).Select

  For A = 1 To R8E - R8S + 1
    If B(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = B(A, 1)
    End If
    If B(A, 1) = 0 Then
      A = R8E - R8S + 1
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  For A = 1 To R
    Data(A, 1) = 0
  Next A
  For A = 1 To S
    B(A, 1) = 0
  Next A
End If
'----
'Inputs random number into Data matrix for trees to be
removed in dbh class 9 (54")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R9S <> 0 Then
  For A = R9S To T * 10 * R9E + R9S
    Data(A, 1) = Int(D9S + Rnd * (D9E - D9S + 1))
    If Cells(40, 51) = "True" And Cells(48, 50) = "" Then
      Data(A, 1) = 0
    End If
    If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
      Data(A, 1) = 0
    End If
    '----
    If Cells(75, 54) > 0 Then
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
          If A > 1 Then
            A = A - 1: C = 50
          End If
        End If
      Next C
    End If
    '-----
    If Cells(75, 55) > 0 Then
      G = 0
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
          C = 50: G = 1
        End If
      Next C
        If G <> 1 Then
          A = A - 1
        End If
    End If
```

```
    '------
    h = h + 1
    If h = Cells(29, 50) Then
      A = T * 10 * R9E + R9S
    End If
  Next A

  Application.Run "ClearCopy"

  'Prints random trees in this dbh class
  Range("DM38").Select
  For A = R9S To T * R9E + R9S
    If Data(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = Data(A, 1)
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  Application.Run "BlanksOutDuplicates"
  'Reads unsorted random numbers back into array
  Range("DM38").Select
  For A = 1 To R9E - R9S + 1
    If Cells(A + 38, 117) <> "" Then
      B(A, 1) = Cells(A + 38, 117)
    End If
    ActiveCell.Offset(1, 0).Select
  Next A
  'Prints out final list to be checked
  Range("DA36").Select
  Selection.End(xlDown).Select
  ActiveCell.Offset(1, 0).Select

  For A = 1 To R9E - R9S + 1
    If B(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = B(A, 1)
    End If
    If B(A, 1) = 0 Then
      A = R9E - R9S + 1
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  For A = 1 To R
    Data(A, 1) = 0
  Next A
  For A = 1 To S
    B(A, 1) = 0
  Next A
End If
'----
'Inputs random number into Data matrix for trees to be
removed in dbh class 10 (60")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R10S <> 0 Then
  For A = R10S To T * 10 * R10E + R10S
    Data(A, 1) = Int(D10S + Rnd * (D10E - D10S + 1))
    If Cells(40, 51) = "True" And Cells(49, 50) = "" Then
      Data(A, 1) = 0
    End If
    If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
      Data(A, 1) = 0
    End If
    '-----
    If Cells(75, 54) > 0 Then
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
          If A > 1 Then
            A = A - 1: C = 50
          End If
        End If
      Next C
    End If
    '----
    If Cells(75, 55) > 0 Then
      G = 0
      For C = 1 To 50
        If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
          C = 50: G = 1
        End If
      Next C
        If G <> 1 Then
          A = A - 1
        End If
    End If
    '----
    h = h + 1
    If h = Cells(30, 50) Then
      A = T * 10 * R10E + R10S
    End If
  Next A

  Application.Run "ClearCopy"

  'Prints random trees in this dbh class
  Range("DM38").Select
  For A = R10S To T * R10E + R10S
    If Data(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = Data(A, 1)
    End If
      ActiveCell.Offset(1, 0).Select
  Next A

  Application.Run "BlanksOutDuplicates"
  'Reads unsorted random numbers back into array
  Range("DM38").Select
  For A = 1 To R10E - R10S + 1
    If Cells(A + 38, 117) <> "" Then
      B(A, 1) = Cells(A + 38, 117)
```

```
      End If
      ActiveCell.Offset(1, 0).Select
   Next A

   'Prints out final list to be checked
   Range("DA36").Select
   Selection.End(xlDown).Select
   ActiveCell.Offset(1, 0).Select

   For A = 1 To R10E - R10S + 1
      If B(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = B(A, 1)
      End If
      If B(A, 1) = 0 Then
         A = R10E - R10S + 1
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   For A = 1 To R
      Data(A, 1) = 0
   Next A
   For A = 1 To S
      B(A, 1) = 0
   Next A
End If
'----
'Inputs random number into Data matrix for trees to be
removed in dbh class 11 (66")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R11S <> 0 Then
   For A = R11S To T * 10 * R11E + R11S
      Data(A, 1) = Int(D11S + Rnd * (D11E - D11S + 1))
      If Cells(40, 51) = "True" And Cells(50, 50) = "" Then
         Data(A, 1) = 0
      End If
      If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
         Data(A, 1) = 0
      End If
      '----
      If Cells(75, 54) > 0 Then
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
               If A > 1 Then
                  A = A - 1: C = 50
               End If
            End If
         Next C
      End If
      '-----
      If Cells(75, 55) > 0 Then
         G = 0
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
               C = 50: G = 1
            End If
         Next C
         If G <> 1 Then
            A = A - 1
         End If
      End If
      '-----
      h = h + 1
      If h = Cells(31, 50) Then
         A = T * 10 * R11E + R11S
      End If
   Next A

   Application.Run "ClearCopy"

   'Prints random trees in this dbh class
   Range("DM38").Select
   For A = R11S To T * R11E + R11S
      If Data(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = Data(A, 1)
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   Application.Run "BlanksOutDuplicates"
   'Reads unsorted random numbers back into array
   Range("DM38").Select
   For A = 1 To R11E - R11S + 1
      If Cells(A + 38, 117) <> "" Then
         B(A, 1) = Cells(A + 38, 117)
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
   'Prints out final list to be checked
   Range("DA36").Select
   Selection.End(xlDown).Select
   ActiveCell.Offset(1, 0).Select

   For A = 1 To R11E - R11S + 1
      If B(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = B(A, 1)
      End If
      If B(A, 1) = 0 Then
         A = R11E - R11S + 1
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   For A = 1 To R
      Data(A, 1) = 0
   Next A
   For A = 1 To S
      B(A, 1) = 0
```

```
   Next A
End If

'----
'Inputs random number into Data matrix for trees to be
removed in dbh class 12 (72")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R12S <> 0 Then
   For A = R12S To T * 10 * R12E + R12S
      Data(A, 1) = Int(D12S + Rnd * (D12E - D12S + 1))
      If Cells(40, 51) = "True" And Cells(51, 50) = "" Then
         Data(A, 1) = 0
      End If
      If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
         Data(A, 1) = 0
      End If
      '-----
      If Cells(75, 54) > 0 Then
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
               If A > 1 Then
                  A = A - 1: C = 50
               End If
            End If
         Next C
      End If
      '----
      If Cells(75, 55) > 0 Then
         G = 0
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
               C = 50: G = 1
            End If
         Next C
            If G <> 1 Then
               A = A - 1
            End If
      End If
      '-----
      h = h + 1
      If h = Cells(32, 50) Then
         A = T * 10 * R12E + R12S
      End If
   Next A

   Application.Run "ClearCopy"

   'Prints random trees in this dbh class
   Range("DM38").Select
   For A = R12S To T * R12E + R12S
      If Data(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = Data(A, 1)
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   Application.Run "BlanksOutDuplicates"
   'Reads unsorted random numbers back into array
   Range("DM38").Select
   For A = 1 To R12E - R12S + 1
      If Cells(A + 38, 117) <> "" Then
         B(A, 1) = Cells(A + 38, 117)
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
   'Prints out final list to be checked
   Range("DA36").Select
   Selection.End(xlDown).Select
   ActiveCell.Offset(1, 0).Select

   For A = 1 To R12E - R12S + 1
      If B(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = B(A, 1)
      End If
      If B(A, 1) = 0 Then
         A = R12E - R12S + 1
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   For A = 1 To R
      Data(A, 1) = 0
   Next A
   For A = 1 To S
      B(A, 1) = 0
   Next A
End If
'---
'Inputs random number into Data matrix for trees to be
removed in dbh class 13 (78")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R13S <> 0 Then
   For A = R13S To T * 10 * R13E + R13S
      Data(A, 1) = Int(D13S + Rnd * (D13E - D13S + 1))
      If Cells(40, 51) = "True" And Cells(52, 50) = "" Then
         Data(A, 1) = 0
      End If
      If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
         Data(A, 1) = 0
      End If
      '-----
      If Cells(75, 54) > 0 Then
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
```

```
54) Then
            If A > 1 Then
               A = A - 1: C = 50
            End If
         End If
      Next C
   End If
   '----
   If Cells(75, 55) > 0 Then
      G = 0
      For C = 1 To 50
         If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
            C = 50: G = 1
         End If
      Next C
         If G <> 1 Then
            A = A - 1
         End If
   End If
   '----
   h = h + 1
   If h = Cells(33, 50) Then
      A = T * 10 * R13E + R13S
   End If
Next A

Application.Run "ClearCopy"

'Prints random trees in this dbh class
Range("DM38").Select
For A = R13S To T * R13E + R13S
   If Data(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = Data(A, 1)
   End If
      ActiveCell.Offset(1, 0).Select
Next A

Application.Run "BlanksOutDuplicates"
'Reads unsorted random numbers back into array
Range("DM38").Select
For A = 1 To R13E - R13S + 1
   If Cells(A + 38, 117) <> "" Then
      B(A, 1) = Cells(A + 38, 117)
   End If
   ActiveCell.Offset(1, 0).Select
Next A
'Prints out final list to be checked
Range("DA36").Select
Selection.End(xlDown).Select
ActiveCell.Offset(1, 0).Select

For A = 1 To R13E - R13S + 1
   If B(A, 1) <> 0 Then
      ActiveCell.FormulaR1C1 = B(A, 1)
   End If
   If B(A, 1) = 0 Then
      A = R13E - R13S + 1
   End If
      ActiveCell.Offset(1, 0).Select
Next A

For A = 1 To R
   Data(A, 1) = 0
Next A
For A = 1 To S
   B(A, 1) = 0
Next A
End If
'---
'Inputs random number into Data matrix for trees to be
removed in dbh class 14 (84")
Sheets("DATABASE").Select
Application.Run "SortDBH"
h = 0

If R14S <> 0 Then
   For A = R14S To T * 10 * R14E + R14S
      Data(A, 1) = Int(D14S + Rnd * (D14E - D14S + 1))
   'True means Box B is selected;   40,50 = "" means no
% chance of removal is desired & random number is
removed.
   'If Box A is checked, this section will always pass.
      If Cells(40, 51) = "True" And Cells(53, 50) = "" Then
         Data(A, 1) = 0
      End If
      If Cells(Data(A, 1) + 53, 21) < Cells(20, 55) Or
Cells(Data(A, 1) + 53, 21) > Cells(21, 55) Then
         Data(A, 1) = 0
      End If
      '---
      If Cells(75, 54) > 0 Then
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
54) Then
               If A > 1 Then
                  A = A - 1: C = 50
               End If
            End If
         Next C
      End If
      '----
      If Cells(75, 55) > 0 Then
         G = 0
         For C = 1 To 50
            If (Cells(Data(A, 1) + 53, 17)) = Cells(C + 24,
55) Then
               C = 50: G = 1
            End If
         Next C
            If G <> 1 Then
               A = A - 1
            End If
      End If
```

```
      '----
      h = h + 1
      If h = Cells(34, 50) Then
         A = T * 10 * R14E + R14S
      End If
   Next A

   Application.Run "ClearCopy"

   'Prints random trees in this dbh class
   Range("DM38").Select
   For A = R14S To T * R14E + R14S
      If Data(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = Data(A, 1)
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   'Blanks out duplicates (1st of a pair of duplicates).  For
A = 1 (trust me)
   Application.Run "BlanksOutDuplicates"
   'Reads unsorted random numbers back into array
   Range("DM38").Select
   For A = 1 To R14E - R14S + 1
      If Cells(A + 38, 117) <> "" Then
         B(A, 1) = Cells(A + 38, 117)
      End If
      ActiveCell.Offset(1, 0).Select
   Next A

   'Prints out final list to be checked
   Range("DA36").Select
   Selection.End(xlDown).Select
   ActiveCell.Offset(1, 0).Select

   For A = 1 To R14E - R14S + 1
      If B(A, 1) <> 0 Then
         ActiveCell.FormulaR1C1 = B(A, 1)
      End If
      If B(A, 1) = 0 Then
         A = R14E - R14S + 1
      End If
         ActiveCell.Offset(1, 0).Select
   Next A

   For A = 1 To R
      Data(A, 1) = 0
   Next A
   For A = 1 To S
      B(A, 1) = 0
   Next A
End If

'Keep this for the very end
'================
   Application.Run "SortDBH"
   'Sorts FINAL random numbers in DA38:DA20038
   Range("DA38:DA30038").Select
   Selection.Sort Key1:=Range("DA38"),
Order1:=xlAscending, Header:=xlNo _
      , OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
   Range("O46").Select
Sheets("DATABASE").Select
 'Places a "1" in Col 4 to indicate trees that are cut trees
      For A = 1 To R
         If Cells(A + 37, 105) <> 0 Then
            Cells(53 + Cells(A + 37, 105), 18) = 1
         End If
         If Cells(A + 37 + 1, 105) = "" Then
            A = R
         End If
      Next A

Application.Run "ClearCopy"
Application.Run "ActualNoTreesRemoved"
Application.Run "GeneratePercentGraphBoxA"

GoTo ENDREMOVAL:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err) & vbCr &
vbCr & "Over 5000 trees were selected for Removal." &
vbCr & "Select a more realistic number!"

ENDREMOVAL:
Sheets("REMOVAL").Select
Range("W20").Select
Beep
Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic
Application.Run ("DoItRandom")

End Sub



Sub Rectangle41_Click()

End Sub



Sub ResetBoxA()

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
```

```
Application.ScreenUpdating = False
Application.Calculation = xlManual

Application.Run "StatsStandTable"
Application.Run "StatsVolTable"
Application.Run "ClearStndStckOnOutputParam-
etersSheet"

   Sheets("REMOVAL").Select
   Range("Z5").Select
   ActiveCell.FormulaR1C1 = False

   Range("I69").Select
   ActiveCell.FormulaR1C1 = 1

   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Q26:Q40").Select
   Selection.ClearContents

   Range("AH26:AH40").Select
   Selection.ClearContents

   Range("L26:L40").Select
   Selection.ClearContents

   Range("AC26:AD40").Select
   Selection.ClearContents

   Application.Run "ClearChoicesB"
   Application.Run "GenerateNoTreesinDBHclassA"

   Range("Q20").Select

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub
```

```
Sub ResetBoxB()


Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Application.Run "StatsStandTable"
Application.Run "StatsVolTable"
Application.Run "ClearStndStckOnOutputParam-
etersSheet"

   Sheets("REMOVAL").Select
   Range("AA5").Select
   ActiveCell.FormulaR1C1 = False

   Range("I69").Select
   ActiveCell.FormulaR1C1 = 1

   Sheets("REMOVAL").Select
   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("I81").Select
   ActiveCell.FormulaR1C1 = False
   Range("J81").Select
   ActiveCell.FormulaR1C1 = False
   Range("K81").Select
   ActiveCell.FormulaR1C1 = False

   Range("Q26:Q40").Select
   Selection.ClearContents

   Range("AH26:AH40").Select
   Selection.ClearContents

   Range("L26:L40").Select
   Selection.ClearContents

   Range("AC26:AD40").Select
   Selection.ClearContents

   Application.Run "GenerateNoTreesinDBHclassB"
   Application.Run "ClearChoicesB"

   Range("AO12").Select

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic
```

```
End Sub


Sub ResetMaxMin()

Range("AB82") = 1
Range("AC82") = 1

End Sub


Sub SameValueCalculation()
'
' SameValueCalculation Macro
' Macro recorded 12/31/1969 by Norman Pillsbury


On Error GoTo Errhandler

'
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'
   Sheets("VALUATION").Select
   Dim C(5000, 1)      'Keep at 5000 in case some city
codes their 500 trees with large numbers
   'Best if codes are consecutive (1-500), but cannot
exceed 5000

   units = Range("AU129")
   BF = Cells(128, 40)
   CF = Cells(128, 41)
   DRY = Cells(128, 42)
   WET = Cells(128, 43)

      If units = 3 Then
      Range("P24") = "Finished"
      Range("P25") = "Lumber"
      Range("P26") = "($/bf)"
      Range("Q24") = "BF Volume"
      Range("Q25") = "by Species"
      Range("Q26") = "(bf)"
      End If

      If units = 4 Then
      Range("P24") = "Rough"
      Range("P25") = "Lumber"
      Range("P26") = "($/bf)"
      Range("Q24") = "BF Volume"
      Range("Q25") = "by Species"
      Range("Q26") = "(bf)"
      End If

      If units = 5 Then
      Range("P24") = "Standing or"
      Range("P25") = "Cut Trees/Logs"
      Range("P26") = "($/cf)"
      Range("Q24") = "Volume"
      Range("Q25") = "by Species"
      Range("Q26") = "(cf)"
      End If

      If units = 6 Then
      Range("P24") = "Firewood"
      Range("P25") = "Stacked"
      Range("P26") = "($/Std Cord)"
      Range("Q24") = "Volume "
      Range("Q25") = "by Species"
      Range("Q26") = "(Std Cords)"
      End If

      If units = 7 Then
      Range("P24") = ""
      Range("P25") = "Air Dry Weight"
      Range("P26") = "($/Ton)"
      Range("Q24") = "Weight"
      Range("Q25") = "by Species"
      Range("Q26") = "(Tons)"
      End If

      If units = 8 Then
      Range("P24") = ""
      Range("P25") = "Green Weight"
      Range("P26") = "($/Ton)"
      Range("Q24") = "Weight"
      Range("Q25") = "by Species"
      Range("Q26") = "(Tons)"
      End If

   Range("L1").Select

   Sheets("DATABASE").Select

   'sort by removal, then by species code
   'Range("R54:R55").Select

   If Range("CU23") <> "Sort Removal" Then
      Range("P54:AC50053").Select
      Selection.Sort Key1:=Range("R54"),
Order1:=xlAscending, Key2:=Range("Q54" _
         ), Order2:=xlAscending, Header:=xlNo, Order-
Custom:=1, MatchCase:= _
         False, Orientation:=xlTopToBottom
```

```
      Range("CU23").Select
      ActiveCell.FormulaR1C1 = "Sort Removal"
   End If

      'Range("P54:AC50053").Select
      'Selection.Sort Key1:=Range("R54"),
Order1:=xlAscending, Key2:=Range("Q54" _
      '   ), Order2:=xlAscending, Header:=xlNo, Order-
Custom:=1, MatchCase:= _
      '   False, Orientation:=xlTopToBottom

   Range("R54").Select
   For A = 1 To 50000

      If Cells(A + 53, 18) = 1 Then
         If units = 3 Or units = 4 Then
            C(Cells(A + 53, 17), 1) = C(Cells(A + 53, 17), 1)
+ Cells(A + 53, 23) * BF         '"7" converts 1 cf to 7 bf
         End If
         If units = 5 Then
            C(Cells(A + 53, 17), 1) = C(Cells(A + 53, 17), 1)
+ Cells(A + 53, 23)                  'Vol in cf. No conversion
needed
         End If
         If units = 6 Then
            C(Cells(A + 53, 17), 1) = C(Cells(A + 53, 17), 1)
+ Cells(A + 53, 23) / CF            '62.5 converts 62.5 cf to 1
cord
         End If
         If units = 7 Then
            C(Cells(A + 53, 17), 1) = C(Cells(A + 53, 17), 1)
+ Cells(A + 53, 23) * DRY / 2000  '26.5÷2000 converts 1
cubic ft of wood to Air Dry wt in tons
         End If
         If units = 8 Then
            C(Cells(A + 53, 17), 1) = C(Cells(A + 53, 17), 1)
+ Cells(A + 53, 23) * WET / 2000   '53.0/2000 converts 1
cubic ft of wood to Green wt in tons
         End If
      End If

      If Cells(A + 53 + 1, 18) = "" Then
         A = 50000
      End If

   Next A

   'sort by species code
   Sheets("VALUATION").Select
   Range("M26:O531").Select
   Selection.Sort Key1:=Range("M27"),
Order1:=xlAscending, Header:=xlGuess, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

   Range("Q27").Select
   For A = 1 To 50000
      Range(Cells(A + 26, 17), Cells(A + 26, 17)).Select
      ActiveCell.FormulaR1C1 = C(Cells(A + 26, 13), 1)
      If Cells(A + 26, 13) = "" Then
         A = 50000
      End If
   Next A

   Range("T25").Select
   Selection.AutoFill Destination:=Range("T25:T27"),
Type:=xlFillDefault
   Range("T25:T27").Select
   Range("Q22").Select
   ActiveCell.FormulaR1C1 = "=SUM(R[5]C:R[509]C)"
   Range("Q22").Select
   Selection.AutoFill Destination:=Range("Q22:R22"),
Type:=xlFillDefault
   Range("Q22:R22").Select
   Range("R22").Select
   Selection.NumberFormat = "$#,##0  "


GoTo ENDSMVL:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSMVL:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic


End Sub



Sub SameValueMethod()

On Error GoTo Errhandler

If Range("R10") = False Then
   Range("X132").Select
   ActiveCell.FormulaR1C1 = 1
   Range("P23").Select
Exit Sub
End If

If Range("X132") < 6 Or Range("X132") = 14 Or
Range("X132") = 15 Or Range("X132") = 16 Or
Range("X132") = 25 Then
   Range("X132").Select
   ActiveCell.FormulaR1C1 = 1
```

```
   Range("P27:Q531").Select
   Selection.ClearContents
   Range("P23").Select
Exit Sub
End If

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

'Flat means that there is a flat price regardless of species
or size.
'FlatNo is the current selection (ex: 19 is $100/cord)
'Flat6 is the value/cf of the first choice (Cell X138)
FlatNo = Range("X132")

Flat6 = Range("X138")
Flat7 = Range("X139")
Flat8 = Range("X140")
Flat9 = Range("X141")
Flat10 = Range("X142")
Flat11 = Range("X143")
Flat12 = Range("X144")
Flat13 = Range("X145")

Flat17 = Range("Z149")
Flat18 = Range("Z150")
Flat19 = Range("Z151")
Flat20 = Range("Z152")
Flat21 = Range("Z153")
Flat22 = Range("Z154")
Flat23 = Range("Z155")
Flat24 = Range("Z156")

Range("P27").Select

If FlatNo = 6 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat6
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 7 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat7
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 8 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat8
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 9 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat9
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 10 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat10
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 11 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat11
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 12 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat12
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 13 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat13
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 17 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
```

```
        ActiveCell.FormulaR1C1 = Flat17
      End If
      ActiveCell.Offset(1, 0).Select
    Next A
End If

If FlatNo = 18 Then
    For A = 27 To 531
      If Cells(A, 13) <> "" Then
        ActiveCell.FormulaR1C1 = Flat18
      End If
      ActiveCell.Offset(1, 0).Select
    Next A
End If

If FlatNo = 19 Then
    For A = 27 To 531
      If Cells(A, 13) <> "" Then
        ActiveCell.FormulaR1C1 = Flat19
      End If
      ActiveCell.Offset(1, 0).Select
    Next A
End If

If FlatNo = 20 Then
    For A = 27 To 531
      If Cells(A, 13) <> "" Then
        ActiveCell.FormulaR1C1 = Flat20
      End If
      ActiveCell.Offset(1, 0).Select
    Next A
End If

If FlatNo = 21 Then
    For A = 27 To 531
      If Cells(A, 13) <> "" Then
        ActiveCell.FormulaR1C1 = Flat21
      End If
      ActiveCell.Offset(1, 0).Select
    Next A
End If

If FlatNo = 22 Then
    For A = 27 To 531
      If Cells(A, 13) <> "" Then
        ActiveCell.FormulaR1C1 = Flat22
      End If
      ActiveCell.Offset(1, 0).Select
    Next A
End If

If FlatNo = 23 Then
    For A = 27 To 531
      If Cells(A, 13) <> "" Then
        ActiveCell.FormulaR1C1 = Flat23
      End If
      ActiveCell.Offset(1, 0).Select
    Next A
End If

If FlatNo = 24 Then
    For A = 27 To 531
      If Cells(A, 13) <> "" Then
        ActiveCell.FormulaR1C1 = Flat24
      End If
      ActiveCell.Offset(1, 0).Select
    Next A
End If

Application.Run "SameValueCalculation"

    Range("P22:R22").Select
    With Selection
      .VerticalAlignment = xlCenter
    End With
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Selection.Borders(xlRight).LineStyle = xlContinuous
    Selection.Borders(xlTop).LineStyle = xlContinuous
    Selection.Borders(xlBottom).LineStyle = xlContinuous
    With Selection.Interior
      .ColorIndex = 35
      .Pattern = xlSolid
    End With
    Range("R27:R531").Select
    Selection.Borders(xlLeft).LineStyle = xlContinuous
    Range("Q531").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous

Range("Q27:Q531").Select
Selection.NumberFormat = "#,##0  "
Range("P9").Select

GoTo ENDSAMEMETH:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSAMEMETH
:
Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

    With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
    End With

End Sub
```

```
Sub SameValueMethodOptionsA()
'
' SameValueMethodOptions Macro


On Error GoTo Errhandler
'
'Application.Wait (Now + TimeValue("0:00:10"))

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

   Range("AY126").Select
   ActiveCell.FormulaR1C1 = 1

If Range("AU129") = 1 Or Range("AU129") = 2 Then
   Range("AY130").Select
   Selection.Clear
   Application.CutCopyMode = False
   Range("AY130").Select
End If

   Range("P27:Q531").Select
   Selection.Clear
   Application.CutCopyMode = False

   Range("AY129:AY142").Select
   Selection.Clear
   Application.CutCopyMode = False

If Range("AU129") = 3 Then
   Range("AL145:AL156").Select
   Selection.Copy
   Range("AY130").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
End If

If Range("AU129") = 4 Then
   Range("AM145:AM156").Select
   Selection.Copy
   Range("AY130").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
End If

If Range("AU129") = 5 Then
   Range("AN145:AN156").Select
   Selection.Copy
   Range("AY130").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
End If

If Range("AU129") = 6 Then
   Range("AO145:AO156").Select
   Selection.Copy
   Range("AY130").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
End If

If Range("AU129") = 7 Then
   Range("AP145:AP156").Select
   Selection.Copy
   Range("AY130").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
End If

If Range("AU129") = 8 Then
   Range("AQ145:AQ156").Select
   Selection.Copy
   Range("AY130").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
End If

If Range("AU129") = 9 Then
   Range("AR145:AR156").Select
   Selection.Copy
   Range("AY130").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
End If

   Range("R24").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With
   Range("R25").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With

   Range("M27:R531").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With
   Range("S22").Select

   Range("O27:R531").Select
   With Selection.Borders(xlEdgeLeft)
     .LineStyle = xlContinuous
     .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeRight)
```

```
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = 1
  End With
  With Selection.Borders(xlInsideVertical)
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = 1
  End With


  Range("O27:R531").Select
  With Selection.Borders(xlEdgeBottom)
    .LineStyle = xlContinuous
    .Weight = xlHairline
    .ColorIndex = 1
  End With

  With Selection.Borders(xlInsideHorizontal)
    .LineStyle = xlContinuous
    .Weight = xlHairline
    .ColorIndex = 1
  End With

Columns("N:N").EntireColumn.AutoFit

  Range("P22:R22").Select
  With Selection.Borders(xlEdgeLeft)
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = 1
  End With
  With Selection.Borders(xlEdgeTop)
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = 1
  End With
  With Selection.Borders(xlEdgeBottom)
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = 1
  End With
  With Selection.Borders(xlEdgeRight)
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = 1
  End With
  With Selection.Borders(xlInsideVertical)
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = 1
  End With
  Selection.Borders(xlInsideHorizontal).LineStyle =
xlNone
  With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
  End With
  Range("P22:R22").Select
  With Selection
    .VerticalAlignment = xlCenter
  End With

AllowFormattingCells = True
Range("P22:R22").Interior.Color = RGB(204, 255, 204)
AllowFormattingCells = False

Range("L1").Select


GoTo ENDSMVALMETHA:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSMVALMETHA:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

  With Application
    .Calculation = xlAutomatic
    .MaxChange = 0.001
  End With

End Sub



Sub SameValueMethodOptionsB()


On Error GoTo Errhandler

If Range("R10") = False Then
  Range("AU129").Select
  ActiveCell.FormulaR1C1 = 1
  Range("P23").Select
Exit Sub
End If

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

'Flat means that there is a flat price regardless of species
or size.
```

```
'FlatNo is the current selection (ex: 3 is $100/cord)
'Flat6 is the value/cf of the first choice (Cell AY132)
FlatNo = Range("AY126")

Flat6 = Range("AY134")
Flat7 = Range("AY135")
Flat8 = Range("AY136")
Flat9 = Range("AY137")
Flat10 = Range("AY138")
Flat11 = Range("AY139")
Flat12 = Range("AY140")
Flat13 = Range("AY141")

Range("P27").Select

If FlatNo = 8 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat6
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 9 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat7
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 10 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat8
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 11 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat9
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 12 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat10
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 13 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat11
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 14 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat12
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 15 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat13
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

If FlatNo = 16 Then
   For A = 27 To 531
      If Cells(A, 13) <> "" Then
         ActiveCell.FormulaR1C1 = Flat17
      End If
      ActiveCell.Offset(1, 0).Select
   Next A
End If

Application.Run "SameValueCalculation"

Range("Q27:Q531").Select
Selection.NumberFormat = "#,##0  "
Range("P9").Select

   Range("R24").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With
   Range("R25").Select
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With

   Range("M27:R531").Select
```

```
   With Selection.Font
      .Name = "Arial"
      .Size = 11
   End With

   Range("P27:P531").Select
   Selection.NumberFormat = "$#,##0.00     "
   With Selection
      .HorizontalAlignment = xlRight
      .VerticalAlignment = xlBottom
   End With

   Range("R27:R531").Select
   Selection.NumberFormat = "$#,##0.00     "
   With Selection
      .HorizontalAlignment = xlRight
      .VerticalAlignment = xlBottom
   End With

   Range("Q27:Q531").Select
   Selection.NumberFormat = "#,##0     "

Range("L1").Select


   'Colors cells orange
   AllowFormattingCells = True
   Range("P27:P531").Interior.Color = RGB(255, 204,
153)
   AllowFormattingCells = False


GoTo ENDSMVLMETHOPB:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSMVLMETHOPB:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate

   With Application
      .Calculation = xlAutomatic
      .MaxChange = 0.001
   End With

End Sub



Sub SaveCUFIMPRO1()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames permanently for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   'This is name of main program: CUFIM-PRO date/
time
   Dim CUFIMName As String
   CUFIMName = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim ProgBackup As String
   ProgBackup = Range("S17")

   Sheets("ID").Select
   Dim CUFIMProgramName As String
   CUFIMProgramName = Range("S18") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsm"

   'SAVE CUFIM Program File with new name (date/
time).....
   ChDir (BaseDir & CUFIMFolder & ProgBackup)
   ActiveWorkbook.SaveAs CUFIMProgramName

GoTo ENDIMPORT:
'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)
ENDIMPORT:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub
```

```
Sub SaveCUFIMPRO2()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames permanently for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   'This is name of main program: CUFIM-PRO date/
time
   Dim CUFIMName As String
   CUFIMName = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim ProgBackup As String
   ProgBackup = Range("S17")

   Dim CUFIMProgramName As String
   CUFIMProgramName = Range("S18") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsm"

   Sheets("SETUP").Select

   'SAVE CUFIM Program File with new name (date/
time).....
   ChDir (BaseDir & CUFIMFolder & ProgBackup)
   ActiveWorkbook.SaveAs CUFIMProgramName

GoTo ENDIMPORT:
'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)
ENDIMPORT:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub
```

```
Sub SaveCUFIMPRO3()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames permanently for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   'This is name of main program: CUFIM-PRO date/
time
   Dim CUFIMName As String
   CUFIMName = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim ProgBackup As String
   ProgBackup = Range("S17")

   Dim CUFIMProgramName As String
   CUFIMProgramName = Range("S18") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsm"

   Sheets("DATABASE").Select

   'SAVE CUFIM Program File with new name (date/
time).....
   ChDir (BaseDir & CUFIMFolder & ProgBackup)
   ActiveWorkbook.SaveAs CUFIMProgramName

GoTo ENDIMPORT:
'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)
ENDIMPORT:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
```

```
Application.Calculation = xlAutomatic

End Sub


Sub SaveCUFIMPRO4()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames permanently for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   'This is name of main program: CUFIM-PRO date/
time
   Dim CUFIMName As String
   CUFIMName = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim ProgBackup As String
   ProgBackup = Range("S17")

   Dim CUFIMProgramName As String
   CUFIMProgramName = Range("S18") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsm"

   Sheets("STATS").Select

   'SAVE CUFIM Program File with new name (date/
time).....
   ChDir (BaseDir & CUFIMFolder & ProgBackup)
   ActiveWorkbook.SaveAs CUFIMProgramName

GoTo ENDIMPORT:
'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)
ENDIMPORT:
Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub


Sub SaveCUFIMPRO5()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames permanently for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   'This is name of main program: CUFIM-PRO date/
time
   Dim CUFIMName As String
   CUFIMName = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim ProgBackup As String
   ProgBackup = Range("S17")

   Dim CUFIMProgramName As String
   CUFIMProgramName = Range("S18") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsm"

   Sheets("OUTPUT PARAMETERS").Select

   'SAVE CUFIM Program File with new name (date/
time).....
   ChDir (BaseDir & CUFIMFolder & ProgBackup)
   ActiveWorkbook.SaveAs CUFIMProgramName

GoTo ENDIMPORT:
```

```
'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)
ENDIMPORT:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub SaveCUFIMPRO6()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames permanently for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   'This is name of main program: CUFIM-PRO date/
time
   Dim CUFIMName As String
   CUFIMName = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim ProgBackup As String
   ProgBackup = Range("S17")

   Dim CUFIMProgramName As String
   CUFIMProgramName = Range("S18") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsm"

   Sheets("REMOVAL").Select

   'SAVE CUFIM Program File with new name (date/
time).....
   ChDir (BaseDir & CUFIMFolder & ProgBackup)
   ActiveWorkbook.SaveAs CUFIMProgramName

GoTo ENDIMPORT:
'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)
ENDIMPORT:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub SaveCUFIMPRO7()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames permanently for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   'This is name of main program: CUFIM-PRO date/
time
   Dim CUFIMName As String
   CUFIMName = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim ProgBackup As String
   ProgBackup = Range("S17")

   Dim CUFIMProgramName As String
   CUFIMProgramName = Range("S18") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsm"
```

```
   Sheets("MGT OPTIONS").Select

   'SAVE CUFIM Program File with new name (date/
time).....
   ChDir (BaseDir & CUFIMFolder & ProgBackup)
   ActiveWorkbook.SaveAs CUFIMProgramName

GoTo ENDIMPORT:
'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)
ENDIMPORT:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub SaveCUFIMPRO8()

On Error GoTo Errhandler

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Fixes Pathnames permanently for this session
   Sheets("ID").Select
   Range("X3:X6").Select
   Selection.Copy
   Range("R3").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=False
   Range("S10").Select
   Application.CutCopyMode = False

   'This is name of main program: CUFIM-PRO date/
time
   Dim CUFIMName As String
   CUFIMName = Range("R6")
   Dim BaseDir As String
   BaseDir = Range("S16")
   Dim CUFIMFolder As String
   CUFIMFolder = Range("S23")
   Dim ProgBackup As String
   ProgBackup = Range("S17")

   Dim CUFIMProgramName As String
   CUFIMProgramName = Range("S18") & "  " &
Format(Date, "mmm dd, yyyy") & "--Time is" &
Format(Time, " hh mm ss") & ".xlsm"

   Sheets("VALUATION").Select

   'SAVE CUFIM Program File with new name (date/
time).....
   ChDir (BaseDir & CUFIMFolder & ProgBackup)
   ActiveWorkbook.SaveAs CUFIMProgramName

GoTo ENDIMPORT:
'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)
ENDIMPORT:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub GoToParameters()


If Range("I81") = True And Cells(46, 20) <> 2 Then
   RndMsg = MsgBox("Go to OUTPUT PARAMETERS
sheet and Reset Max and Min to None.")
   Range("I81") = False
   Exit Sub
End If

End Sub



Sub Seg1()


Sheets("REMOVAL").Select

If Range("AA5") = False And Range("Z5") = False Then
   RndMsg = MsgBox("Select Box A or B, then Choice 1,
2 or 3.")
   Exit Sub
End If

If Range("AA5") = True And Range("Z5") = True Then
   RndMsg = MsgBox("Select Box A or B, then Choice 1,
2 or 3.")
```

```
   Exit Sub
End If

If Range("AA5") = True And Range("L82") = 0 Then
   Range("Q20").Select
   RndMsg = MsgBox("Select Box A or B, then Choice 1,
2 or 3.")
   Exit Sub
End If

If Range("Z5") = True And Range("AC82") = 0 Then
   Range("AA17").Select
   RndMsg = MsgBox("Select Box A or B, then Choice 1,
2 or 3.")
   Exit Sub
End If



'Dim calcMode, updateMode
'calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Application.Run "ClearStandStock"
Sheets("REMOVAL").Select


'==========

'Clears Actual No. Trees Removed Column on Removal
Sheet

If Range("Z5") = True Then          'Box B selected
   Range("AH26:AH40").Select
   Selection.ClearContents
   Range("AH20").Select
End If

If Range("AA5") = True Then          'Box A selected
   Range("Q26:Q40").Select
   Selection.ClearContents
   Range("Q20").Select
End If


Sheets("OUTPUT PARAMETERS").Select
Range("AL15:AN64").Select   'Excluded trees
Selection.Sort Key1:=Range("AL15"),
Order1:=xlAscending, Header:=xlNo, _
   OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
Range("AP15:AR64").Select    'Included trees
Selection.Sort Key1:=Range("AP15"),
Order1:=xlAscending, Header:=xlNo, _
   OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
Range("AO4").Select

'Transfers % chance values to DATABASE sheet if Box B
is checked.
   Sheets("DATABASE").Select
If Cells(40, 51) = "True" Then
   Sheets("REMOVAL").Select
   Range("AD26:AD40").Select
   Selection.Copy
   Sheets("DATABASE").Select
   Range("AX40").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
   Range("AW38").Select
End If


'Transfers % chance values to DATABASE sheet if Box A
is checked.

If Cells(40, 52) = "True" Then
   Sheets("REMOVAL").Select
   Range("BM27:BM41").Select
   Selection.Copy
   Sheets("DATABASE").Select
   Range("AX40").Select
   Selection.PasteSpecial Paste:=xlValues
   Application.CutCopyMode = False
   Range("AW38").Select
End If

Sheets("REMOVAL").Select

End Sub



Sub Seg2()

Sheets("DATABASE").Select
Application.Run "SortDBH"
Application.Run "GenerateNoTreesinDBHclassA"
Application.Run "GenerateNoTreesinDBHclassB"

'=====================================
   'Inputs random number into Data matrix for 6" trees
to be removed
   'R1S =1, first number of removal
   'R1E = 109, last number of removal
   'D1S = 2, Tree SEQ Num for 1st Start for 6" dbh class
```

```
   'D1E = n, Tree SEQ Num for last tree for 6" dbh class
'====================================

   Sheets("DATABASE").Select
   Range("DL38:DN50038").Select
   Selection.ClearContents   'Clears DL38 to DN2038
   Range("DA38:DA50038").Select
   Selection.ClearContents   'Clears DA38 to DA20038
   Range("R54:R50053").Select
   Selection.ClearContents   'Clears Removal codes
   Range("O45").Select


End Sub




Sub SortArray()
'
' Macro recorded 5/6/02 by NRM
'
   Range("BC11:BC87").Select
   Selection.Copy
   Range("BB11").Select
   ActiveSheet.Paste
   Range("BD11:BD111").Select
   Application.CutCopyMode = False
   Selection.Copy
   Range("BB11").Select
   Selection.End(xlDown).Select
   Range("BB88").Select
   ActiveSheet.Paste
   Range("BE11:BE119").Select
   Application.CutCopyMode = False
   Selection.Copy
   Range("BB11").Select
   Selection.End(xlDown).Select
   Range("BB189").Select
   ActiveSheet.Paste
   Range("BF11:BF121").Select
   Application.CutCopyMode = False
   Selection.Copy
   Range("BB11").Select
   Selection.End(xlDown).Select
   Range("BB298").Select
   ActiveSheet.Paste
   Range("BG11:BG121").Select
   Application.CutCopyMode = False
   Selection.Copy
   Range("BB11").Select
   Selection.End(xlDown).Select
   Range("BB366").Select
   ActiveSheet.Paste
   Range("BH11:BH121").Select
   Application.CutCopyMode = False
   Selection.Copy
   Range("BB11").Select
   Selection.End(xlDown).Select
   Range("BB395").Select
   ActiveSheet.Paste
   Range("BI11:BI121").Select
   Application.CutCopyMode = False
   Selection.Copy
   Range("BB11").Select
   Selection.End(xlDown).Select
   Range("BB409").Select
   ActiveSheet.Paste
   Range("BJ11:BJ121").Select
   Application.CutCopyMode = False
   Selection.Copy
   Range("BB11").Select
   Selection.End(xlDown).Select
   Range("BB413").Select
   ActiveSheet.Paste
   Application.CutCopyMode = False
   Range("BB11:BB415").Select
   Selection.Sort Key1:=Range("BB11"),
Order1:=xlAscending, Header:=xlGuess _
      , OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
   Range("BB9").Select
End Sub



Sub SortDBH()
'
' SortDBH Macro
' Macro recorded 5/5/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort Dbh" Then
Exit Sub
End If

   Range("P54:AO50053").Select
   Selection.Sort Key1:=Range("U54"),
Order1:=xlAscending, Key2:=Range("V54" _
      ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
      False, Orientation:=xlTopToBottom

   Range("CU23").Select
   ActiveCell.FormulaR1C1 = "Sort Dbh"
```

```
   Range("U49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub



Sub SortMeth2aSPCode()
'
' SortMeth2aSPCode Macro
' Macro recorded 8/1/2002 by Norman Pillsbury
'
   Range("W27:AK76").Select
   Selection.Sort Key1:=Range("W27"),
Order1:=xlAscending, Header:=xlNo, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
   Range("W9").Select
End Sub



Sub SortMeth2aValue()
'
' SortMeth2aValue Macro
' Macro recorded 8/1/2002 by Norman Pillsbury
'
   Range("W27:AK76").Select
   Range("AK27").Activate
   Selection.Sort Key1:=Range("AK27"),
Order1:=xlDescending, Header:=xlNo, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
   Range("W9").Select
End Sub



Sub SortMeth2bTotValue()
'
' SortMeth2bTotValue Macro
' Macro recorded 8/1/2002 by Norman Pillsbury
'
   Range("W27:AI76").Select
   Range("AI27").Activate
   Selection.Sort Key1:=Range("AI27"),
Order1:=xlDescending, Header:=xlNo, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
   Range("W9").Select
End Sub



Sub SortMeth2aVolume()
'
' SortMeth2aVolume Macro
' Macro recorded 8/1/2002 by Norman Pillsbury
'
   Range("W27:AK76").Select
   Range("AJ27").Activate
   Selection.Sort Key1:=Range("AJ27"),
Order1:=xlDescending, Header:=xlNo, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom
   Range("W9").Select
End Sub



Sub SortSPCODE()

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort Species Code" Then
Exit Sub
End If

   Range("P54:AO50053").Select
   Selection.Sort Key1:=Range("Q54"),
Order1:=xlAscending, Header:=xlNo, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

   Range("CU23").Select
   ActiveCell.FormulaR1C1 = "Sort Species Code"
   Range("O49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub
```

```
Sub SortStatSpeciesList()
'
' SortStatSpeciesList Macro
'

'
    Range("V21:Z525").Select
    ActiveWorkbook.Worksheets("STATS").Sort.Sort-
Fields.Clear
    ActiveWorkbook.Worksheets("STATS").Sort.Sort-
Fields.Add Key:=Range("W21:W525") _
        , SortOn:=xlSortOnValues, Order:=xlAscending,
DataOption:=xlSortNormal
    With ActiveWorkbook.Worksheets("STATS").Sort
        .SetRange Range("V21:Z525")
        .Header = xlNo
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With
    Range("Z18").Select
End Sub
```

```
Sub SortStatSpGroup()
'
' SortStatSpGroup Macro
'

'
    Range("V21:Z525").Select
    ActiveWorkbook.Worksheets("STATS").Sort.Sort-
Fields.Clear
    ActiveWorkbook.Worksheets("STATS").Sort.Sort-
Fields.Add Key:=Range("X21:X525") _
        , SortOn:=xlSortOnValues, Order:=xlAscending,
DataOption:=xlSortNormal
    With ActiveWorkbook.Worksheets("STATS").Sort
        .SetRange Range("V21:Z525")
        .Header = xlNo
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With
    Range("Z18").Select
End Sub
```

```
Sub SortStatVolume()
'
' SortStatVolume Macro
'

'
    Range("V21:Z525").Select
    ActiveWorkbook.Worksheets("STATS").Sort.Sort-
Fields.Clear
    ActiveWorkbook.Worksheets("STATS").Sort.Sort-
Fields.Add Key:=Range("Z21:Z525") _
        , SortOn:=xlSortOnValues, Order:=xlDescending,
DataOption:=xlSortNormal
    With ActiveWorkbook.Worksheets("STATS").Sort
        .SetRange Range("V21:Z525")
        .Header = xlNo
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With
    Range("Z18").Select
End Sub
```

```
Sub SortTREERECORD()

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort Tree Record" Then
Exit Sub
End If

    Range("P54:AO50053").Select
    Selection.Sort Key1:=Range("P54"),
Order1:=xlAscending, Header:=xlNo, _
        OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

    Range("CU23").Select
    ActiveCell.FormulaR1C1 = "Sort Tree Record"
    Range("P49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub
```

```
Sub SortTREESEQNUMBER()

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort Tree Seq Number" Then
Exit Sub
End If
   'This sort starts with Col O; none of the others do.
   Range("O54:AO50053").Select
   Selection.Sort Key1:=Range("O54"),
Order1:=xlAscending, Header:=xlNo, _
      OrderCustom:=1, MatchCase:=False,
Orientation:=xlTopToBottom

   Range("CU23").Select
   ActiveCell.FormulaR1C1 = "Sort Tree Seq Number"
   Range("O49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub



Sub SortVAR1()
'
' SortVAR1Macro
' Macro recorded 5/5/02 by NRM
'Normally this is Location/Address

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort VAR1" Then
Exit Sub
End If

   Range("p54:AO50053").Select
   Selection.Sort Key1:=Range("AF54"),
Order1:=xlAscending, Key2:=Range("AG54" _
      ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
      False, Orientation:=xlTopToBottom

   Range("CU23").Select
   ActiveCell.FormulaR1C1 = "Sort VAR1"
   Range("U49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub


Sub SortVAR2()
'
' SortVAR2Macro
' Macro recorded 5/5/02 by NRM
'Normally this is Date Measured

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort VAR2" Then
Exit Sub
End If

   Range("P54:AO50053").Select
   Selection.Sort Key1:=Range("AG54"),
Order1:=xlAscending, Key2:=Range("Q54" _
      ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
      False, Orientation:=xlTopToBottom

   Range("CU23").Select
   ActiveCell.FormulaR1C1 = "Sort VAR2"
   Range("U49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub


Sub SortVAR3()
'
' SortVAR3Macro
```

```
‘ Macro recorded 5/5/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets(“DATABASE”).Select

If Range(“CU23”) = “Sort VAR3” Then
Exit Sub
End If

   Range(“P54:AO50053”).Select
   Selection.Sort Key1:=Range(“AH54”),
Order1:=xlAscending, Key2:=Range(“Q54” _
      ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
      False, Orientation:=xlTopToBottom

   Range(“CU23”).Select
   ActiveCell.FormulaR1C1 = “Sort VAR3”
   Range(“U49”).Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub


Sub SortVAR4()
‘
‘ SortVAR4Macro
‘ Macro recorded 5/5/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets(“DATABASE”).Select

If Range(“CU23”) = “Sort VAR4” Then
Exit Sub
End If

   Range(“P54:AO50053”).Select
   Selection.Sort Key1:=Range(“AI54”),
Order1:=xlAscending, Key2:=Range(“Q54” _
      ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
      False, Orientation:=xlTopToBottom

   Range(“CU23”).Select
   ActiveCell.FormulaR1C1 = “Sort VAR4”
   Range(“U49”).Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub

Sub SortVAR5()
‘
‘ SortVAR5Macro
‘ Macro recorded 5/5/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets(“DATABASE”).Select

If Range(“CU23”) = “Sort VAR5” Then
Exit Sub
End If

   Range(“P54:AO50053”).Select
   Selection.Sort Key1:=Range(“AJ54”),
Order1:=xlAscending, Key2:=Range(“Q54” _
      ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
      False, Orientation:=xlTopToBottom

   Range(“CU23”).Select
   ActiveCell.FormulaR1C1 = “Sort VAR5”
   Range(“U49”).Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub

Sub SortVAR6()
‘
‘ SortVAR6Macro
‘ Macro recorded 5/5/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False
```

```
Sheets("DATABASE").Select

If Range("CU23") = "Sort VAR6" Then
Exit Sub
End If

   Range("P54:AO50053").Select
   Selection.Sort Key1:=Range("AK54"),
Order1:=xlAscending, Key2:=Range("Q54" _
      ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
      False, Orientation:=xlTopToBottom

   Range("CU23").Select
   ActiveCell.FormulaR1C1 = "Sort VAR6"
   Range("U49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub

Sub SortVAR7()
'
' SortVAR7Macro
' Macro recorded 5/5/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort VAR7" Then
Exit Sub
End If

   Range("P54:AO50053").Select
   Selection.Sort Key1:=Range("AL54"),
Order1:=xlAscending, Key2:=Range("Q54" _
      ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
      False, Orientation:=xlTopToBottom

   Range("CU23").Select
   ActiveCell.FormulaR1C1 = "Sort VAR7"
   Range("U49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub

Sub SortVAR8()
'
' SortVAR8Macro
' Macro recorded 5/5/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort VAR8" Then
Exit Sub
End If

   Range("P54:AO50053").Select
   Selection.Sort Key1:=Range("AM54"),
Order1:=xlAscending, Key2:=Range("Q54" _
      ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
      False, Orientation:=xlTopToBottom

   Range("CU23").Select
   ActiveCell.FormulaR1C1 = "Sort VAR8"
   Range("U49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub

Sub SortVAR9()
'
' SortVAR9Macro
' Macro recorded 5/5/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort VAR9" Then
Exit Sub
End If

   Range("P54:AO50053").Select
   Selection.Sort Key1:=Range("AN54"),
Order1:=xlAscending, Key2:=Range("Q54" _
```

```
    ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
    False, Orientation:=xlTopToBottom

  Range("CU23").Select
  ActiveCell.FormulaR1C1 = "Sort VAR9"
  Range("U49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub


Sub SortVAR10()
'
' SortVAR10Macro
' Macro recorded 5/5/02 by NRM

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select

If Range("CU23") = "Sort VAR10" Then
Exit Sub
End If

  Range("P54:AO50053").Select
  Selection.Sort Key1:=Range("AO54"),
Order1:=xlAscending, Key2:=Range("Q54" _
    ), Order2:=xlAscending, Header:=xlGuess, Order-
Custom:=1, MatchCase:= _
    False, Orientation:=xlTopToBottom

  Range("CU23").Select
  ActiveCell.FormulaR1C1 = "Sort VAR10"
  Range("U49").Select

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

End Sub



Sub StandReturntoTop()
'
' StandReturntoTop Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'
  Range("R1").Select
End Sub



Sub StandTableA1()
'
' StandTableA1 Macro
' Macro recorded 5/28/02 by Norman Pillsbury

On Error GoTo Errhandler

'Stand Table for trees that will be removed by the Re-
moval sheet.

Sheets("DATABASE").Select
If Range("R48") = 0 Then
  Sheets("MGT OPTIONS").Select
  RndMsg = MsgBox("First you must complete the Re-
moval process.  See REMOVAL sheet.")
  Exit Sub
End If

Sheets("MGT OPTIONS").Select
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Clears Stand Array
  Range("U34:BS154").Select   'Don't clear Col S
  Selection.ClearContents
  Range("S160:BT288").Select
  Selection.ClearContents
  Range("S20").Select

Dim Stand#(120, 50)      'Data Matrix for Stand Table [8
dbh classes, 50 groups]
Dim B(120, 2)            'Array for upper dbh class values

For A = 1 To 120
  If Cells(34, 137) > 0 Then
    B(A, 1) = Cells(33 + A, 139)      'Upper dia for dbh
classes
    B(A, 2) = Cells(33 + A, 137)      'dbh class
  Else
    A = 120
  End If
Next A

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database
```

```
Sheets("DATABASE").Select
'Application.Run "SortDBH"

'==========
   'For A = ;  searches total database
   'B array is number of dbh classes
   'For C =;   is number of dbh classes
   'For D =;   is number of species groups
   'R54 Selects first possible removal tree in Removal Col.
   'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                    'in C loop dbh class, then....
'==========

   For A = 54 To MaxNo + 54
      If Cells(A, 18) <> "" Then            'Removal Code
Col.  It could have any number in it, not just "1".
      For C = 1 To 120
         If Cells(A, 21) < B(C, 1) Then
         For D = 1 To 50
            If Cells(A, 20) = D Then
               Stand(C, D) = Stand(C, D) + 1: C = 120: D =
50
            End If
         Next D
         End If
      Next C
      End If
   Next A

Sheets("MGT OPTIONS").Select
'==========
   'Outputs Stand Table data
   Range("V34").Select
   For C = 1 To 120
      Range(Cells(C + 33, 21), Cells(C + 33, 21)).Select
      ActiveCell.FormulaR1C1 = B(C, 2)
      For D = 1 To 50
      Range(Cells(C + 33, D + 21), Cells(C + 33, D + 21)).
Select
      If Stand(C, D) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = Stand(C, D)
      End If
      Next D
      If Cells(C + 33, 21) = "" Then
         D = 50: C = 120
      End If
   Next C
'==========
Range("S26").Select
ActiveCell.FormulaR1C1 = "Number of Trees to be Re-
moved (by Dbh Class and by Species Group)"


GoTo ENDSTNA1:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSTNA1:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub StandTableA2()
'
' StandTableA2 Macro
' Macro recorded 5/28/02 by Norman Pillsbury


On Error GoTo Errhandler

'Stand Table for trees that will be removed by the Re-
moval sheet.

Sheets("MGT OPTIONS").Select

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Clears Stand Array
   Range("U34:BS154").Select      'Don't clear Col S
   Selection.ClearContents
   Range("S160:BT288").Select
   Selection.ClearContents
   Range("CI23").Select

Dim Stand#(120, 50)      'Data Matrix for Stand Table [8
dbh classes, 50 groups]
Dim B(120, 2)            'Array for upper dbh class values

For A = 1 To 120
   If Cells(34, 137) > 0 Then
      B(A, 1) = Cells(33 + A, 139)       'Upper dia for dbh
classes
      B(A, 2) = Cells(33 + A, 137)       'dbh class
   Else
      A = 120
   End If
Next A
```

```
Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("database").Select
'Application.Run "SortDBH"

'==========
  'For A = ;  searches total database
  'B array is number of dbh classes
  'For C =;   is number of dbh classes
  'For D =;   is number of species groups
  'R54 Selects first possible removal tree in Removal Col.
  'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                    'in C loop dbh class, then....
'==========

  For A = 54 To MaxNo + 54
    For C = 1 To 120
      If Cells(A, 21) < B(C, 1) Then
      For D = 1 To 50
        If Cells(A, 20) = D Then
          Stand(C, D) = Stand(C, D) + 1: C = 120: D =
50
        End If
      Next D
      End If
    Next C
  Next A

Sheets("MGT OPTIONS").Select
'==========
  'Outputs Stand Table data
  Range("V34").Select
  For C = 1 To 120
    Range(Cells(C + 33, 21), Cells(C + 33, 21)).Select
    ActiveCell.FormulaR1C1 = B(C, 2)
    For D = 1 To 50
    Range(Cells(C + 33, D + 21), Cells(C + 33, D + 21)).
Select
    If Stand(C, D) > 0 Then
      'MsgBox (D1mat(A, 1))
      ActiveCell.FormulaR1C1 = Stand(C, D)
    End If
    Next D
    If Cells(C + 33, 21) = "" Then
      D = 50: C = 120
    End If
  Next C
'==========
Range("S26").Select
ActiveCell.FormulaR1C1 = "Number of Trees in Inven-
tory (by Dbh Class and by Species Group)"

GoTo ENDSTNA2:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDSTNA2:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Range("S25").Select
End Sub



Sub StandTableFromIncludedList()
'
' StandTableFromIncludedList Macro
' Macro recorded 5/28/02 by Norman Pillsbury

'Sheets("DATABASE").Select
'If Range("R48") = 0 Then
'   Sheets("DATABASE").Select
'   RndMsg = MsgBox("First you must complete the
Removal process.  See REMOVAL sheet.")
'   Exit Sub
'End If

Sheets("OUTPUT PARAMETERS").Select
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

'Clears Stand Array
  Range("BA15:CX135").Select   'Don't clear Col AX
  Selection.ClearContents
  Range("S20").Select

Sheets("DATABASE").Select
Dim Stand#(120, 50)      'Data Matrix for Stand Table [8
dbh classes, 50 groups]
Dim B(120, 2)            'Array for upper dbh class values

For A = 1 To 120
  If Cells(34, 137) > 0 Then
    B(A, 1) = Cells(33 + A, 139)      'Upper dia for dbh
classes
    B(A, 2) = Cells(33 + A, 137)      'dbh class
  Else
    A = 120
  End If
Next A
```

```
Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("DATABASE").Select

'===========
  'For A = ;  searches total database
  'B array is number of dbh classes
  'For C =;   is number of dbh classes
  'For D =;   is number of species groups
  'R54 Selects first possible removal tree in Removal Col.
  'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                    'in C loop dbh class, then....
'===========

  For A = 54 To MaxNo + 54
    If Cells(A, 17) <> "" Then            'Species code ex-
ists.
      For C = 1 To 120
        If Cells(A, 21) < B(C, 1) Then    'Tree dbh
        For D = 1 To 50
          If Cells(A, 17) = D Then
'Matches Species code to dbh class
            Stand(C, D) = Stand(C, D) + 1: C = 120: D =
50
          End If
        Next D
        End If
      Next C
      End If
  Next A

'Stop


Sheets("OUTPUT PARAMETERS").Select
'===========
  'Outputs Stand Table data
  Range("BA15").Select
  For C = 1 To 120
    Range(Cells(C + 14, 52), Cells(C + 14, 52)).Select
    ActiveCell.FormulaR1C1 = B(C, 2)
    For D = 1 To 50
    Range(Cells(C + 14, D + 52), Cells(C + 14, D + 52)).
Select
    If Stand(C, D) > 0 Then
      'MsgBox (D1mat(A, 1))
      ActiveCell.FormulaR1C1 = Stand(C, D)
    End If
    Next D
    If Cells(C + 14, 52) = "" Then
      D = 50: C = 120
    End If
  Next C
'===========
Range("AX2").Select
'ActiveCell.FormulaR1C1 = "Number of Trees to be Re-
moved (by Dbh Class and by Species Group)"

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub StandTableOnOutputParameterSheet()
'
' StandTableOnOutputParameterSheet1 Macro
' Macro recorded 5/28/02 by Norman Pillsbury

'Sheets("DATABASE").Select
'If Range("R48") = 0 Then
'   Sheets("DATABASE").Select
'   RndMsg = MsgBox("First you must complete the
Removal process.  See REMOVAL sheet.")
'   Exit Sub
'End If

'Example of vbYesNo
OutPut = MsgBox("If you continue with Step 9b, ALL
selections and choices on the Removal Sheet with be
Deleted.    This CANNOT be Undone." & vbCr & vbCr
& "Do You Want To Continue?", vbYesNo, "YOU ARE
BEGINNING THE 100% REMOVAL OPTION")
If OutPut = 7 Then    'Output = 7(No)
  Exit Sub
  Else
  'Output = 6(Yes)
End If


Sheets("OUTPUT PARAMETERS").Select
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("DATABASE").Select
'Clears Removal Column on Database sheet
  Range("R54:R50053").Select
  Selection.Clear

'Formats Removal Column on Database sheet
  Range("R54:R50053").Select
  With Selection
    .HorizontalAlignment = xlCenter
```

```
    End With

    Range("R52:R53").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous

    Range("R54:R50052").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Selection.Borders(xlEdgeLeft).LineStyle = xlNone
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    Selection.Borders(xlEdgeRight).LineStyle = xlNone
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With

    Range("R54:R50052").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    Range("O45").Select


'----------------------------------------
'Clears and Resets Box A choices on REMOVAL sheet
    Sheets("REMOVAL").Select
    Range("AA5").Select
    ActiveCell.FormulaR1C1 = False

    Range("I69").Select
    ActiveCell.FormulaR1C1 = 1

    Sheets("REMOVAL").Select
    Range("I81").Select
    ActiveCell.FormulaR1C1 = False
    Range("J81").Select
    ActiveCell.FormulaR1C1 = False
    Range("K81").Select
    ActiveCell.FormulaR1C1 = False

    Range("I81").Select
    ActiveCell.FormulaR1C1 = False
    Range("J81").Select
    ActiveCell.FormulaR1C1 = False
    Range("K81").Select
    ActiveCell.FormulaR1C1 = False

    Range("Q26:Q40").Select
    Selection.ClearContents

    Range("L26:L40").Select
    Selection.ClearContents

'Clears and Resets Box B choices on REMOVAL sheet
    Application.Run "ClearChoicesB"
    Range("AH26:AH40").Select
    Selection.ClearContents

    Range("AC26:AD40").Select
    Selection.ClearContents

    Range("Z5").Select
    ActiveCell.FormulaR1C1 = False
'----------------------------------------

Sheets("OUTPUT PARAMETERS").Select
'Clears Stand Array
    Range("BA15:CX64").Select   'Don't clear Col S
    Selection.ClearContents
    Range("S20").Select

    'Copies and transposes Species Codes from Inclusion
list to Stand table
    Range("AP15:AP64").Select
    Selection.Copy
    Range("BA14").Select
```

```
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
      False, Transpose:=True
   Range("BA12").Select
   Application.CutCopyMode = False

Sheets("DATABASE").Select
Dim Stand#(120, 50)      'Data Matrix for Stand Table [8
dbh classes, 50 groups]
Dim B(120, 2)            'Array for upper dbh class values
Dim F(50, 1)    'Data Matrix for trees included in OUT-
PUT PARAMETERS stand table

For A = 1 To 120
   If Cells(34, 137) > 0 Then
      B(A, 1) = Cells(33 + A, 139)       'Upper dia for dbh
classes
      B(A, 2) = Cells(33 + A, 137)       'dbh class
   Else
      A = 120
   End If
Next A

'Reads in Sp Codes from Inclusion list on DATABASE
sheet
For A = 1 To 50
   If Cells(75, 55) > 0 Then
      F(A, 1) = Cells(24 + A, 55)
   Else
      A = 120
   End If
Next A

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database
Sheets("DATABASE").Select

'===========
   'For A = ;  searches total database
   'B array is number of dbh classes
   'For C =;   is number of dbh classes
   'For D =;   is number of species groups
   'R54 Selects first possible removal tree in Removal Col.
   'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                  'in C loop dbh class, then....
'===========

   For A = 54 To MaxNo + 54            'Cks all trees in
database

  'This tree record based on restrictions in Step 7 for
<lower limit and >upper limit
    If Cells(A, 21) < Cells(20, 55) Or Cells(A, 21) >
Cells(21, 55) Then
     GoTo NextA
    End If

    If Cells(A, 17) <> "" Then          'Species code ex-
ists.

     For E = 1 To 50                    'Cks if tree is on
Inclusion list
      If (Cells(A, 17)) = Cells(E + 24, 55) Then

      For C = 1 To 120                  'dbh classes
       If Cells(A, 21) < B(C, 1) Then   'Tree dbh
        For D = 1 To 50
          If Cells(A, 17) = F(D, 1) Then
'Matches Species code to dbh class
            Stand(C, D) = Stand(C, D) + 1: C = 120: D =
50: E = 50
          End If
        Next D
        End If
      Next C
     End If
    Next E
   End If

NextA:   Next A


Sheets("OUTPUT PARAMETERS").Select
'===========
   'Outputs Stand Table data
   Range("BA15").Select
   For C = 1 To 120
      Range(Cells(C + 14, 52), Cells(C + 14, 52)).Select
      ActiveCell.FormulaR1C1 = B(C, 2)
      For D = 1 To 50
      Range(Cells(C + 14, D + 52), Cells(C + 14, D + 52)).
Select
      If Stand(C, D) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = Stand(C, D)
      End If
      Next D
      If Cells(C + 14, 52) = "" Then
         D = 50: C = 120
      End If
   Next C
'===========
Application.Calculate

   Range("BA10:CX64").Select
   ActiveWorkbook.Worksheets("OUTPUT PARAM-
ETERS").Sort.SortFields.Clear
   ActiveWorkbook.Worksheets("OUTPUT PARAM-
ETERS").Sort.SortFields.Add Key:=Range _
      ("BA10:CX10"), SortOn:=xlSortOnValues,
Order:=xlDescending, DataOption:= _
      xlSortNormal
```

```
   With ActiveWorkbook.Worksheets("OUTPUT PA-
RAMETERS").Sort
      .SetRange Range("BA10:CX64")
      .Header = xlNo
      .MatchCase = False
      .Orientation = xlLeftToRight
      .SortMethod = xlPinYin
      .Apply
   End With
   Range("AX1").Select
   Beep

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub




Sub StatsStandTable()


On Error GoTo Errhandler

Sheets("STATS").Select

'Clears Stand Arrays
   Range("Q23:Q41").Select
   Selection.ClearContents

   Range("O20").Select
'===========

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("STATS").Select
Dim StandStats#(18, 1)     'Data Matrix for Stand Table
[8 dbh classes, 40 groups]
Dim B#(18, 1)              'Array for upper dbh class values

B(1, 1) = Range("P23")   'Upper diameter for dbh class 1
B(2, 1) = Range("P24")
B(3, 1) = Range("P25")
B(4, 1) = Range("P26")
B(5, 1) = Range("P27")
B(6, 1) = Range("P28")
B(7, 1) = Range("P29")
B(8, 1) = Range("P30")
B(9, 1) = Range("P31")
B(10, 1) = Range("P32")
B(11, 1) = Range("P33")
B(12, 1) = Range("P34")
B(13, 1) = Range("P35")
B(14, 1) = Range("P36")
B(15, 1) = Range("P37")   'Upper diameter for dbh class
15
B(16, 1) = Range("P38")
B(17, 1) = Range("P39")
B(18, 1) = Range("P40")


Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("DATABASE").Select

'====================================


   For A = 54 To MaxNo + 54
      If Cells(A, 15) <> "" Then           'A tree exists in the
database
         For C = 1 To 18
            If Cells(A, 21) < B(C, 1) Then
               StandStats(C, 1) = StandStats(C, 1) + 1: C =
18
            End If
         Next C
      End If
   Next A

Sheets("STATS").Select

'===========

   'Outputs Stand Table data
   Range("Q23").Select
   For C = 1 To 18
      Range(Cells(C + 22, 17), Cells(C + 22, 17)).Select
      If StandStats(C, 1) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = StandStats(C, 1)
      End If
   Next C
'===========
Range("O20").Select


GoTo ENDREMOVAL:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDREMOVAL:

Application.ScreenUpdating = updateMode
```

```
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub




Sub StatsVolTable()


On Error GoTo Errhandler

Sheets("STATS").Select

'Clears Stand Arrays
   Range("R23:R41").Select
   Selection.ClearContents

   Range("R20").Select
'==========

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("STATS").Select
Dim StandStats#(18, 1)     'Data Matrix for Stand Table
[8 dbh classes, 40 groups]
Dim B#(18, 1)              'Array for upper dbh class values

B(1, 1) = Range("P23")   'Upper diameter for dbh class 1
B(2, 1) = Range("P24")
B(3, 1) = Range("P25")
B(4, 1) = Range("P26")
B(5, 1) = Range("P27")
B(6, 1) = Range("P28")
B(7, 1) = Range("P29")
B(8, 1) = Range("P30")
B(9, 1) = Range("P31")
B(10, 1) = Range("P32")
B(11, 1) = Range("P33")
B(12, 1) = Range("P34")
B(13, 1) = Range("P35")
B(14, 1) = Range("P36")
B(15, 1) = Range("P37")   'Upper diameter for dbh class
15
B(16, 1) = Range("P38")
B(17, 1) = Range("P39")
B(18, 1) = Range("P40")

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("DATABASE").Select

'====================================


   For A = 54 To MaxNo + 54
      If Cells(A, 15) <> "" Then           'A tree exists in the
database
         For C = 1 To 18
            If Cells(A, 21) < B(C, 1) Then
'Checks to see if dbh is < upper limit of 1st dbh class;
then 2nd, etc.
               StandStats(C, 1) = StandStats(C, 1) + Cells(A,
23): C = 18
            End If
         Next C
      End If
   Next A

Sheets("STATS").Select

'==========

   'Outputs Stand VOL data
   Range("R23").Select
   For C = 1 To 18
      Range(Cells(C + 22, 18), Cells(C + 22, 18)).Select
      If StandStats(C, 1) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = StandStats(C, 1)
      End If
   Next C
'==========
Range("R20").Select


GoTo ENDREMOVAL:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDREMOVAL:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub StockReturntoTop()
```

```
'
' StockReturntoTop Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'
    Range("BZ1").Select
End Sub



Sub StockTableOnOutputParameterSheet()
'
' StockTableOnOutputParameterSheet1 Macro
' Macro recorded 5/28/02 by Norman Pillsbury

'Sheets("DATABASE").Select
'If Range("R48") = 0 Then
'    Sheets("DATABASE").Select
'    RndMsg = MsgBox("First you must complete the
Removal process.  See REMOVAL sheet.")
'    Exit Sub
'End If

Sheets("OUTPUT PARAMETERS").Select
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("DATABASE").Select
'Clears Removal Column on Database sheet
    Range("R54:R50053").Select
    Selection.Clear

'Formats Removal Column on Database sheet
    Range("R54:R50053").Select
    With Selection
        .HorizontalAlignment = xlCenter
    End With

    Range("R52:R53").Select
    Selection.Borders(xlBottom).LineStyle = xlContinuous

    Range("R54:R50052").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Selection.Borders(xlEdgeLeft).LineStyle = xlNone
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    Selection.Borders(xlEdgeRight).LineStyle = xlNone
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With

    Range("R54:R50052").Select
    Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = 1
    End With
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .Weight = xlHairline
        .ColorIndex = 1
    End With
    Range("O45").Select


'----------------------------------------
'Clears and Resets Box A choices on REMOVAL sheet
    Sheets("REMOVAL").Select
    Range("AA5").Select
    ActiveCell.FormulaR1C1 = False

    Range("I69").Select
    ActiveCell.FormulaR1C1 = 1

    Sheets("REMOVAL").Select
    Range("I81").Select
    ActiveCell.FormulaR1C1 = False
    Range("J81").Select
    ActiveCell.FormulaR1C1 = False
```

```
    Range("K81").Select
    ActiveCell.FormulaR1C1 = False

    Range("I81").Select
    ActiveCell.FormulaR1C1 = False
    Range("J81").Select
    ActiveCell.FormulaR1C1 = False
    Range("K81").Select
    ActiveCell.FormulaR1C1 = False

    Range("Q26:Q40").Select
    Selection.ClearContents

    Range("L26:L40").Select
    Selection.ClearContents

'Clears and Resets Box B choices on REMOVAL sheet
    Application.Run "ClearChoicesB"
    Range("AH26:AH40").Select
    Selection.ClearContents

    Range("AC26:AD40").Select
    Selection.ClearContents

    Range("Z5").Select
    ActiveCell.FormulaR1C1 = False
'-----------------------------------------

Sheets("OUTPUT PARAMETERS").Select
'Clears Stock Array
    Range("AZ78:CX127").Select   'Don't clear Col AX
    Selection.ClearContents
    Range("AV12").Select

    'Copies and transposes Species Codes from Inclusion
list to Stock table
    Range("AP15:AP64").Select
    Selection.Copy
    Range("BA77").Select
    Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=True
    Range("AV12").Select
    Application.CutCopyMode = False

Sheets("DATABASE").Select
Dim Stand#(120, 50)      'Data Matrix for Stand Table [8
dbh classes, 50 groups]
Dim B(120, 2)            'Array for upper dbh class values
Dim F(50, 1)     'Data Matrix for trees included in OUT-
PUT PARAMETERS stand table

For A = 1 To 120
    If Cells(34, 137) > 0 Then
        B(A, 1) = Cells(33 + A, 139)       'Upper dia for dbh
classes
        B(A, 2) = Cells(33 + A, 137)       'dbh class
    Else
        A = 120
    End If
Next A

'Reads in Sp Codes from Inclusion list into F[*]
For A = 1 To 50
    If Cells(75, 55) > 0 Then
        F(A, 1) = Cells(24 + A, 55)
    Else
        A = 120
    End If
Next A

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database
Sheets("DATABASE").Select

'==========
    'For A = ;  searches total database
    'B array is number of dbh classes
    'For C =;   is number of dbh classes
    'For D =;   is number of species groups
    'R54 Selects first possible removal tree in Removal Col.
    'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                    'in C loop dbh class, then....
'==========

    For A = 54 To MaxNo + 54             'Cks all trees in
database

   'This tree record based on restrictions in Step 7 for
<lower limit and >upper limit
      If Cells(A, 21) < Cells(20, 55) Or Cells(A, 21) >
Cells(21, 55) Then
      GoTo NextA
      End If

       If Cells(A, 17) <> "" Then            'Species code ex-
ists.
         For E = 1 To 50                      'Cks if tree is on
Inclusion list
          If (Cells(A, 17)) = Cells(E + 24, 55) Then
          For C = 1 To 120                    'dbh classes
           If Cells(A, 21) < B(C, 1) Then     'Tree dbh
            For D = 1 To 50
              If Cells(A, 17) = F(D, 1) Then
'Matches Species code to dbh class
                Stand(C, D) = Stand(C, D) + Cells(A, 23): C
= 120: D = 50: E = 50
              End If
            Next D
            End If
          Next C
         End If
```

```
      Next E
    End If
NextA:    Next A

Sheets("OUTPUT PARAMETERS").Select
'===========
   'Outputs Stand Table data
   Range("BA78").Select
   For C = 1 To 120
      Range(Cells(C + 77, 52), Cells(C + 77, 52)).Select
      ActiveCell.FormulaR1C1 = B(C, 2)
      For D = 1 To 50
      Range(Cells(C + 77, D + 52), Cells(C + 77, D + 52)).
Select
      If Stand(C, D) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = Stand(C, D)
      End If
      Next D
      If Cells(C + 77, 52) = "" Then
         D = 50: C = 120
      End If
   Next C
'===========
Application.Calculate

   Range("BA73:CX127").Select
   ActiveWorkbook.Worksheets("OUTPUT PARAM-
ETERS").Sort.SortFields.Clear
   ActiveWorkbook.Worksheets("OUTPUT PARAM-
ETERS").Sort.SortFields.Add Key:=Range _
      ("BA73:CX73"), SortOn:=xlSortOnValues,
Order:=xlDescending, DataOption:= _
      xlSortNormal
   With ActiveWorkbook.Worksheets("OUTPUT PA-
RAMETERS").Sort
      .SetRange Range("BA73:CX127")
      .Header = xlNo
    '  .MatchCase = False
      .Orientation = xlLeftToRight
      .SortMethod = xlPinYin
      .Apply
   End With
   Range("BA8").Select

Range("AX1").Select
'ActiveCell.FormulaR1C1 = "Number of Trees to be Re-
moved (by Dbh Class and by Species Group)"

Beep

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub
```

```
Sub StatsStandTable()


On Error GoTo Errhandler

Sheets("STATS").Select

'Clears Stand Arrays
   Range("Q23:Q41").Select
   Selection.ClearContents

   Range("O20").Select
'===========

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("STATS").Select
Dim StandStats#(18, 1)     'Data Matrix for Stand Table
[8 dbh classes, 40 groups]
Dim B#(18, 1)              'Array for upper dbh class values

B(1, 1) = Range("P23")   'Upper diameter for dbh class 1
B(2, 1) = Range("P24")
B(3, 1) = Range("P25")
B(4, 1) = Range("P26")
B(5, 1) = Range("P27")
B(6, 1) = Range("P28")
B(7, 1) = Range("P29")
B(8, 1) = Range("P30")
B(9, 1) = Range("P31")
B(10, 1) = Range("P32")
B(11, 1) = Range("P33")
B(12, 1) = Range("P34")
B(13, 1) = Range("P35")
B(14, 1) = Range("P36")
B(15, 1) = Range("P37")   'Upper diameter for dbh class
15
B(16, 1) = Range("P38")
B(17, 1) = Range("P39")
B(18, 1) = Range("P40")


Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("DATABASE").Select
```

```
'====================================

   For A = 54 To MaxNo + 54
      If Cells(A, 15) <> "" Then          'A tree exists in the
database
         For C = 1 To 18
            If Cells(A, 21) < B(C, 1) Then
               StandStats(C, 1) = StandStats(C, 1) + 1: C =
18
            End If
         Next C
      End If
   Next A

Sheets("STATS").Select

'===========

   'Outputs Stand Table data
   Range("Q23").Select
   For C = 1 To 18
      Range(Cells(C + 22, 17), Cells(C + 22, 17)).Select
      If StandStats(C, 1) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = StandStats(C, 1)
      End If
   Next C
'===========
Range("O20").Select


GoTo ENDREMOVAL:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDREMOVAL:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub




Sub StatsVolTable()


On Error GoTo Errhandler

Sheets("STATS").Select

'Clears Stand Arrays
   Range("R23:R41").Select
   Selection.ClearContents

   Range("R20").Select
'===========

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("STATS").Select
Dim StandStats#(18, 1)     'Data Matrix for Stand Table
[8 dbh classes, 40 groups]
Dim B#(18, 1)              'Array for upper dbh class values

B(1, 1) = Range("P23")   'Upper diameter for dbh class 1
B(2, 1) = Range("P24")
B(3, 1) = Range("P25")
B(4, 1) = Range("P26")
B(5, 1) = Range("P27")
B(6, 1) = Range("P28")
B(7, 1) = Range("P29")
B(8, 1) = Range("P30")
B(9, 1) = Range("P31")
B(10, 1) = Range("P32")
B(11, 1) = Range("P33")
B(12, 1) = Range("P34")
B(13, 1) = Range("P35")
B(14, 1) = Range("P36")
B(15, 1) = Range("P37")   'Upper diameter for dbh class
15
B(16, 1) = Range("P38")
B(17, 1) = Range("P39")
B(18, 1) = Range("P40")

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database

Sheets("DATABASE").Select

'====================================

   For A = 54 To MaxNo + 54
      If Cells(A, 15) <> "" Then          'A tree exists in the
database
         For C = 1 To 18
            If Cells(A, 21) < B(C, 1) Then
'Checks to see if dbh is < upper limit of 1st dbh class;
then 2nd, etc.
               StandStats(C, 1) = StandStats(C, 1) + Cells(A,
23): C = 18
            End If
```

```
        Next C
      End If
   Next A

Sheets("STATS").Select

'==========

   'Outputs Stand VOL data
   Range("R23").Select
   For C = 1 To 18
      Range(Cells(C + 22, 18), Cells(C + 22, 18)).Select
      If StandStats(C, 1) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = StandStats(C, 1)
      End If
   Next C
'==========
Range("R20").Select


GoTo ENDREMOVAL:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDREMOVAL:

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub




Sub StockReturntoTop()
'
' StockReturntoTop Macro
' Macro recorded 11/2/2002 by Norman Pillsbury
'
   Range("BZ1").Select
End Sub




Sub StockTableOnOutputParameterSheet()
'
' StockTableOnOutputParameterSheet1 Macro
' Macro recorded 5/28/02 by Norman Pillsbury

'Sheets("DATABASE").Select
'If Range("R48") = 0 Then
'   Sheets("DATABASE").Select
'   RndMsg = MsgBox("First you must complete the
Removal process.  See REMOVAL sheet.")
'   Exit Sub
'End If

Sheets("OUTPUT PARAMETERS").Select
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

Sheets("DATABASE").Select
'Clears Removal Column on Database sheet
   Range("R54:R50053").Select
   Selection.Clear

'Formats Removal Column on Database sheet
   Range("R54:R50053").Select
   With Selection
      .HorizontalAlignment = xlCenter
   End With

   Range("R52:R53").Select
   Selection.Borders(xlBottom).LineStyle = xlContinuous

   Range("R54:R50052").Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   Selection.Borders(xlEdgeLeft).LineStyle = xlNone
   With Selection.Borders(xlEdgeTop)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlEdgeBottom)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
   Selection.Borders(xlEdgeRight).LineStyle = xlNone
   Selection.Borders(xlInsideVertical).LineStyle = xlNone
   With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With

   Range("R54:R50052").Select
   Selection.Borders(xlDiagonalDown).LineStyle =
xlNone
   Selection.Borders(xlDiagonalUp).LineStyle = xlNone
   With Selection.Borders(xlEdgeLeft)
      .LineStyle = xlContinuous
      .Weight = xlThin
```

```
    .ColorIndex = 1
  End With
  With Selection.Borders(xlEdgeTop)
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = 1
  End With
  With Selection.Borders(xlEdgeBottom)
    .LineStyle = xlContinuous
    .Weight = xlHairline
    .ColorIndex = 1
  End With
  With Selection.Borders(xlEdgeRight)
    .LineStyle = xlContinuous
    .Weight = xlThin
    .ColorIndex = 1
  End With
  Selection.Borders(xlInsideVertical).LineStyle = xlNone
  With Selection.Borders(xlInsideHorizontal)
    .LineStyle = xlContinuous
    .Weight = xlHairline
    .ColorIndex = 1
  End With
  Range("O45").Select


'-----------------------------------------
'Clears and Resets Box A choices on REMOVAL sheet
  Sheets("REMOVAL").Select
  Range("AA5").Select
  ActiveCell.FormulaR1C1 = False

  Range("I69").Select
  ActiveCell.FormulaR1C1 = 1

  Sheets("REMOVAL").Select
  Range("I81").Select
  ActiveCell.FormulaR1C1 = False
  Range("J81").Select
  ActiveCell.FormulaR1C1 = False
  Range("K81").Select
  ActiveCell.FormulaR1C1 = False

  Range("I81").Select
  ActiveCell.FormulaR1C1 = False
  Range("J81").Select
  ActiveCell.FormulaR1C1 = False
  Range("K81").Select
  ActiveCell.FormulaR1C1 = False

  Range("Q26:Q40").Select
  Selection.ClearContents

  Range("L26:L40").Select
  Selection.ClearContents

'Clears and Resets Box B choices on REMOVAL sheet
  Application.Run "ClearChoicesB"
  Range("AH26:AH40").Select
  Selection.ClearContents

  Range("AC26:AD40").Select
  Selection.ClearContents

  Range("Z5").Select
  ActiveCell.FormulaR1C1 = False
'-----------------------------------------

Sheets("OUTPUT PARAMETERS").Select
'Clears Stock Array
  Range("AZ78:CX127").Select   'Don't clear Col AX
  Selection.ClearContents
  Range("AV12").Select

  'Copies and transposes Species Codes from Inclusion
list to Stock table
  Range("AP15:AP64").Select
  Selection.Copy
  Range("BA77").Select
  Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:= _
    False, Transpose:=True
  Range("AV12").Select
  Application.CutCopyMode = False

Sheets("DATABASE").Select
Dim Stand#(120, 50)      'Data Matrix for Stand Table [8
dbh classes, 50 groups]
Dim B(120, 2)            'Array for upper dbh class values
Dim F(50, 1)    'Data Matrix for trees included in OUT-
PUT PARAMETERS stand table

For A = 1 To 120
  If Cells(34, 137) > 0 Then
    B(A, 1) = Cells(33 + A, 139)      'Upper dia for dbh
classes
    B(A, 2) = Cells(33 + A, 137)      'dbh class
  Else
    A = 120
  End If
Next A

'Reads in Sp Codes from Inclusion list into F[*]
For A = 1 To 50
  If Cells(75, 55) > 0 Then
    F(A, 1) = Cells(24 + A, 55)
  Else
    A = 120
  End If
Next A

Sheets("STATS").Select
MaxNo = Range("I22")      'Max number of trees in
database
```

```
Sheets("DATABASE").Select

'==========
   'For A = ;  searches total database
   'B array is number of dbh classes
   'For C =;   is number of dbh classes
   'For D =;   is number of species groups
   'R54 Selects first possible removal tree in Removal Col.
   'If Cells(A,21)<>;    if dbh of current tree is less than
upper dbh of dbh class
                        'in C loop dbh class, then....
'==========

   For A = 54 To MaxNo + 54              'Cks all trees in
database

  'This tree record based on restrictions in Step 7 for
<lower limit and >upper limit
    If Cells(A, 21) < Cells(20, 55) Or Cells(A, 21) >
Cells(21, 55) Then
     GoTo NextA
     End If

      If Cells(A, 17) <> "" Then            'Species code ex-
ists.
       For E = 1 To 50                       'Cks if tree is on
Inclusion list
        If (Cells(A, 17)) = Cells(E + 24, 55) Then
        For C = 1 To 120                     'dbh classes
         If Cells(A, 21) < B(C, 1) Then      'Tree dbh
          For D = 1 To 50
            If Cells(A, 17) = F(D, 1) Then
'Matches Species code to dbh class
              Stand(C, D) = Stand(C, D) + Cells(A, 23): C
= 120: D = 50: E = 50
            End If
          Next D
          End If
       Next C
      End If
      Next E
    End If
NextA:    Next A

Sheets("OUTPUT PARAMETERS").Select
'==========
   'Outputs Stand Table data
   Range("BA78").Select
   For C = 1 To 120
      Range(Cells(C + 77, 52), Cells(C + 77, 52)).Select
      ActiveCell.FormulaR1C1 = B(C, 2)
      For D = 1 To 50
      Range(Cells(C + 77, D + 52), Cells(C + 77, D + 52)).
Select
      If Stand(C, D) > 0 Then
         'MsgBox (D1mat(A, 1))
         ActiveCell.FormulaR1C1 = Stand(C, D)
      End If
      Next D
      If Cells(C + 77, 52) = "" Then
         D = 50: C = 120
      End If
   Next C
'==========
Application.Calculate

   Range("BA73:CX127").Select
   ActiveWorkbook.Worksheets("OUTPUT PARAM-
ETERS").Sort.SortFields.Clear
   ActiveWorkbook.Worksheets("OUTPUT PARAM-
ETERS").Sort.SortFields.Add Key:=Range _
      ("BA73:CX73"), SortOn:=xlSortOnValues,
Order:=xlDescending, DataOption:= _
      xlSortNormal
   With ActiveWorkbook.Worksheets("OUTPUT PA-
RAMETERS").Sort
      .SetRange Range("BA73:CX127")
      .Header = xlNo
     '  .MatchCase = False
      .Orientation = xlLeftToRight
      .SortMethod = xlPinYin
      .Apply
   End With
   Range("BA8").Select


Range("AX1").Select
'ActiveCell.FormulaR1C1 = "Number of Trees to be Re-
moved (by Dbh Class and by Species Group)"

Beep

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub


Sub Top50DbhClassesOnStatsSheet()
'
' ACTUALLY, it's the top 500 on STATS sheet
'
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Sheets("SETUP").Select
```

```
   Application.Run "NoOfSpeciesOnSetupSheet"
   Range("B36:F540").Select
   'Range("B540").End(xlUp).Select
   'RowNumber = ActiveCell.Row
   Range(ActiveCell, ActiveCell.Offset(0,
4).End(xlDown)).Select
   Selection.Copy

   Sheets("STATS").Select
   Range("V21").Select
   Selection.PasteSpecial Paste:=xlValues,
Operation:=xlNone, SkipBlanks:=False, Transpose:=False
   Selection.PasteSpecial Paste:=xlFormats,
Operation:=xlNone, SkipBlanks:=False, Transpose:=False

   Range("V21:Z525").Select
   Range("Y21").Activate
   ActiveWorkbook.Worksheets("STATS").Sort.Sort-
Fields.Clear
   ActiveWorkbook.Worksheets("STATS").Sort.Sort-
Fields.Add Key:=Range("Y21"), _
      SortOn:=xlSortOnValues, Order:=xlDescending,
DataOption:=xlSortNormal
   With ActiveWorkbook.Worksheets("STATS").Sort
      .SetRange Range("V21:Z525")
      .Header = xlNo
    '  .MatchCase = False
      .Orientation = xlTopToBottom
      .SortMethod = xlPinYin
      .Apply
   End With

   Columns("W:W").EntireColumn.AutoFit

   Range("V21:Z525").Select
   Application.CutCopyMode = False

    Selection.Borders(xlInsideVertical).LineStyle = xl-
None
   With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
      With Selection.Borders(xlInsideVertical)
      .LineStyle = xlContinuous
      .Weight = xlThin
      .ColorIndex = 1
   End With
   With Selection.Borders(xlInsideHorizontal)
      .LineStyle = xlContinuous
      .Weight = xlHairline
      .ColorIndex = 1
   End With
   Range("Y12").Select

AllowFormattingCells = True
Range("V21:Z525").Interior.Color = RGB(255, 204, 153)
AllowFormattingCells = False

   Sheets("SETUP").Select
   Application.CutCopyMode = False
   Range("E32").Select

   Sheets("STATS").Select

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

End Sub



Sub UpdateVolbyBranchSize()
'
' UpdateVolbyBranchSize Macro
' Macro recorded 9/24/2002 by Norman Pillsbury

On Error GoTo Errhandler

Application.Run ("DoItBRANCHES")

Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.Calculation = xlManual
Application.ScreenUpdating = False

Sheets("DATABASE").Select
Application.Run "SORTSpeciesGroups"
Range("X54").Select

For A = 1 To 50000
   If Cells(A + 53, 23) <> "" Then
      For B = 30 To 80
         If Cells(A + 53, 20) = Cells(B, 74) Then
            Cells(A + 53, 24) = Cells(B, 81) * Cells(A + 53,
23) / 100
            Cells(A + 53, 25) = Cells(B, 82) * Cells(A + 53,
23) / 100
            Cells(A + 53, 26) = Cells(B, 83) * Cells(A + 53,
23) / 100
            Cells(A + 53, 27) = Cells(B, 84) * Cells(A + 53,
23) / 100
            Cells(A + 53, 28) = Cells(B, 85) * Cells(A + 53,
23) / 100
            Cells(A + 53, 29) = Cells(B, 86) * Cells(A + 53,
23) / 100
            B = 80
         End If
      Next B
```

```
   Else
      A = 50000
   End If
Next A

Application.Run "SortTREESEQNUMBER"

GoTo ENDBRSIZE:

'What an error reports via the msgbox
Errhandler:
MsgBox "Error#" & Err & " : " & Error(Err)

ENDBRSIZE:

Application.ScreenUpdating = updateMode
Application.Calculation = calcMode
Application.Calculate
Application.Calculation = xlAutomatic

Sheets("DATABASE").Select

Application.Run ("DoItBRANCHES")
Beep

End Sub
```

```
Sub VIIIaMethodOne()
'
' Clear Method VIII a Method One Macro
' Macro recorded 11/8/2002 by Norman Pillsbury
'
Dim calcMode, updateMode
calcMode = Application.Calculation
updateMode = Application.ScreenUpdating
Application.ScreenUpdating = False
Application.Calculation = xlManual

   Range("R10").Select
   ActiveCell.FormulaR1C1 = "TRUE"
   ActiveCell.FormulaR1C1 = "FALSE"
   Range("M27:O531").Select
   Selection.ClearContents

   Range("S22:V81,W77:W81").Select

   AllowFormattingCells = True
   Range("A5:B8").Interior.Color = RGB(0, 0, 0)
   AllowFormattingCells = False
   Range("W77").Activate

   Range("T22").Select

   Application.Run "Method1Check"

Application.ScreenUpdating = updateMode
Application.Calculate
Application.Calculation = calcMode
Application.Calculation = xlAutomatic

Range("R1").Select

End Sub
```