**Project Administration Data Sheet**

- **Project No.:** D-48-679
- **Project Director:** John Myers
- **Sponsor:** U.S. Department of the Interior - National Park Service

**Type Agreement:** Delivery Order No. PX-0001-4-1219 (Under ROA No. CX-0001-2-0036)

**Award Period:** From 9/25/84 To 7/30/85 (Performance) 10/25/84 To 4/25/85 (Reports)

- **Sponsor Amount:**
  - Estimated: $9,548
  - Funded: $9,548

- **Cost Sharing Amount:** None

**Title:** "Census of Treated Historic Masonry Building - Analytical Framework"

**Administrative Data**

<table>
<thead>
<tr>
<th>1) Sponsor Technical Contact:</th>
<th>2) Sponsor Admin/Contractual Matters:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brian J. Lindberg x4820</td>
<td>U.S. Department of the Interior</td>
</tr>
<tr>
<td></td>
<td>National Park Service</td>
</tr>
<tr>
<td></td>
<td>Preservation Assistance Division (424)</td>
</tr>
<tr>
<td></td>
<td>Washington, D.C. 20240</td>
</tr>
<tr>
<td></td>
<td>(202) 343-8149</td>
</tr>
</tbody>
</table>

**Defense Priority Rating:** N/A  
**Military Security Classification:** N/A

**Restrictions**

- See Attached Gov't Supplemental Information Sheet for Additional Requirements.
- Foreign travel must have prior approval — Contact OCA in each case. Domestic travel requires sponsor approval where total will exceed greater of $500 or 125% of approved proposal budget category.
- Equipment: Title vests with Sponsor if acquisition cost is $1,000 or more; However, none is proposed.

**Comments:**

- I.D. #02.111.000.85.002
- Procurement/EES Supply Services
- Library
- Project File
- Other: A. Jones

**Copies To:**

- Project Director
- Research Administrative Network
- Research Property Management
- Accounting
  
  FORM OCA 4.383
Project No. D-48-679
Includes Subproject No(s).

Project Director(s) John Myers

Sponsor U.S. Department of the Interior-National Park Service

Title Census of Treated Historic Masonry Building-Analytical Framework

Effective Completion Date: 7/30/85 (Performance) 7/30/85 (Reports)

Grant/Contract Closeout Actions Remaining:
- [ ] None
- [x] Final Invoice or Final Fiscal Report
- [ ] Closing Documents
- [x] Final Report of Inventions
- [ ] Govt. Property Inventory & Related Certificate
- [ ] Classified Material Certificate
- [ ] Other

Continues Project No. Continued by Project No.

COPIES TO:
- Project Director
- Research Administrative Network
- Research Property Management
- Accounting
- Procurement/GTRI Supply Services
- Research Security Services
- Reports Coordinator (OCA)
- Library
- GTRC
- Research Communications (2)
- Project File
- Other

FORM OCA 69.285
Mr. Dwight Baker  
Preservation Assistance Division  
National Park Service (424)  
P.O. Box 37127  
Washington, DC 20013-7127

Dear Mr. Baker,

A summary of monthly progress and activities on PX-0001-4-1219 under contract CX-0001-2-0036 follows per our discussion.

October 1984 - Work on Task I, consisting of review and analysis of the Census program data, software options and preparation for NPS review meeting, October 17, 1984. Trip to WASO for meeting defined as Task II, 10/17/84.

November 1984 - Review of all data collected, analysis and extraction of data elements per approved plan from October review. Re-evaluation of structure based on additional work sent down by NPS on November 7, to also include a "short form" report.

December 1984 - Completion of the draft, Task III, of the program structure. Submission of that structure for approval. Preparation of recommendations for adjustments to field size and content in manual program to be more efficient in automated version.

January 1985 - Assessment of final program structure and requirements. Submission of a new recommendation to NPS on January 14 to change from Dbase III to Rbase because of potential benefits to NPS by having 1) stand-alone capability and 2) routines in PASCAL.

February 1985 - Draft program structures created in Rbase following approval. 90-day extension requested on recommendation of Anne Grimmer. System programming in progress.

March 1985 - Programming and testing of system with dummy data. Preparation of first demonstration and training for NPS on operating system.

April 1985 - Completion of menus and data entry screens. Meeting on 4/24 to demo system operation: menus, entry screens, user interface, to get NPS input prior to completion of program.
May 1985 - Completion of linkages, refinements based on 4/24 meeting and entry of live data to debug routines. Request of 90-day extension to complete debugging and refinements. Add Census buildings per Task No. 5.


This summarizes the monthly progress to date. If you would like additional details for these tasks, please let me know. As you know, monthly financial summaries and requisitions are submitted separately by our accounting office.

Sincerely,

John H. Myers
Director

JHM:dph

cc: Anne Grimmer

Baker 6.13
Admin #17
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>INSTALLING THE PROGRAM</td>
<td>3</td>
</tr>
<tr>
<td>ADDING NEW CENSUS FORMS</td>
<td>4</td>
</tr>
<tr>
<td>EDIT OPTION</td>
<td>5</td>
</tr>
<tr>
<td>PRINT OPTION</td>
<td>6</td>
</tr>
<tr>
<td>UTILITIES OPTION</td>
<td>7</td>
</tr>
<tr>
<td>QUERY OPTION</td>
<td>8</td>
</tr>
<tr>
<td>CONCLUSION</td>
<td>9</td>
</tr>
</tbody>
</table>

APPENDIX A: NATIONAL REGISTER STATUS CODES
INTRODUCTION

The Census of Treated Historic Masonry Buildings is a program which identifies, collects, stores and tracks information on a wide range of treatments applied to historic masonry buildings. The purpose of the Census is to provide increased levels of accurate data for monitoring the long term affects of treatments on architectural materials. The Census program is an open-ended program which collects large amounts of data on each building including multiple material study areas with multiple treatments. The manual retrieval and use of the data in the census will become increasingly difficult as the number of buildings in the program expands, however the software which is described in this user's guide will permit efficient, user-friendly entry, storage, search and retrieval of Census data.

The Census software program will allow NPS staff to do the following:

1) Add new data to the file in any of the three forms which may occur, i.e. a comprehensive Census form, a short form, and a follow-up form

2) Edit any of the forms which have been entered into the software program

3) Print complete Census forms, short forms or follow-up forms for each building

4) Print a list of all buildings and building numbers entered into the Census program

5) Delete any building form from the Census

6) Search through the data by three levels of location, three types of materials, three treatment types, and restrict the search to buildings which are Tax Act projects, National Historic Landmarks or Federally owned properties.

The Census program is totally "menu-driven", which means that the user is presented menus, or lists of options, and selects the desired option by entering a single letter which is highlighted in each menu choice. The path to specific operations is controlled by additional sub-menus and if user input is required, simple straightforward questions will prompt the user on the screen. Two conventions apply throughout the entire Census program, one is that the option desired is always selected by the entry of a single letter followed by a carriage return <CR>, and the second is that all menus have an option to exit to the previous menu and hence back out of the program to DOS.

NOTE: Software users should first become familiar with the objectives, structure and procedures of the Census of Treated Historic Masonry Buildings. This knowledge will be a prerequisite to using the Census software, since familiarity with the Census will result in
Intuitive recognition of the Census software options, structure and content. Such intuitive recognition will be necessary to effectively utilize the Census database. Prior to running the software, users should consult with the Census program manager in the Preservation Assistance Division.

The following sections outline the functions and capabilities of each of the Census options.
INSTALLING THE PROGRAM

This is a menu-driven program to allow the entry, edit, printing and searching of Census reports. The program uses the structure and selected modules from the micro-computer database program R-base, but Census routines are written in the programming language PASCAL.

The program was prepared for the Preservation Assistance Division, U.S. National Park Service, Department of Interior by the Center for Architectural Conservation, College of Architecture, Georgia Tech, Atlanta, Georgia.

I. CENSUS - INITIALLY LOADING THE PROGRAM

   Step 1. Turn on computer, Boot-up to C> prompt
   Step 2. At the C> prompt type: MD*CENSUS
   Step 3. At the C> prompt type: CD*CENSUS
   Step 4. Place the CENSUS disk in Drive A
   Step 5. Type: Copy A:*. *

   (PROGRAM IS LOADED ON HARD DRIVE UNDER DIRECTORY "CENSUS")

II. CENSUS - RUNNING THE PROGRAM

   (If computer is on, begin with Step 2)

   Step 1. Turn on computer, boot up to C> prompt
   Step 2. At the C> prompt type: CD*CENSUS
   Step 3. At the C> prompt type: C
   Step 4. Follow menu options

NOTES:

If any unusual control characters appear on the screens, edit the file "CONFIG.SYS" on the root directory and confirm that "DEVICE = ANSI.SYS". This should clear up the screens.

This program allows any number of Masonry Study Areas per building and any number of Treatments per Study Area.

It is recommended that the program and all data entered into the program be periodically backed up on floppy disks and stored safely.
ADD OPTION

The "Add" option is displayed on the main menu of the Census program. This option allows the user to conduct four operations:

1) Add a Census long form
2) Add a Census follow-up form
3) Add a Census short form
4) Return to the Main Menu

The user may also exit the program from the Main Menu screen.

When the "Add" option is selected the user will be prompted by a sub-menu which displays the choices listed above. When the user makes a selection the system will ask for the two-letter abbreviation for the state in which the building is located. A unique number will then be assigned to each form and the number will consist of three parts as follows: the first two letters are of the state abbreviation, the third letter will consist of a "C" or an "S" which indicates whether the file is a long Census form or a short form, and the third part is a four-digit number which begins with 0001 and is assigned in sequence within each state. Note the following examples:

<table>
<thead>
<tr>
<th>BLDG NO</th>
<th>BUILDING NAME</th>
<th>STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORC0001</td>
<td>U.S. COURTHOUSE &amp; POST OFFICE</td>
<td>OR</td>
</tr>
<tr>
<td>CAC0001</td>
<td>OLD SAN FRANCISCO MINT</td>
<td>CA</td>
</tr>
</tbody>
</table>

In the "Add" option, screens which are similar in appearance to the Census forms will be displayed on the CRT and the data may be entered on the screens. Upon completion of the last screen, the form will be saved. The "Add" option allows the initial entry of a form. If additional data is to be added to an existing file, it should be added via the "Edit" mode.

Adding Study Areas and Treatments:

An unlimited number of Study Areas may be entered into the program for each building. Following the entry of each Masonry Study Area, an unlimited number of Treatments may be added. Treatments are added following the Study Area to which they apply. Upon completion of Treatment entries, the user is asked if it is desired to enter another Study Area. This process will continue until the user indicates that there are no additional Study Areas or Treatments to enter. The program prompts the user to answer via simple, straightforward questions on the screen.

Entering Data:

Following the entry of data on each screen the user should type the "ESC" key. The user will then be presented with the choice of editing the screen or advancing to the next screen.
EDIT OPTION

The Edit option permits the modification of data in existing files, including the addition of new data to an existing Census file. The Edit mode is selected from the Main Menu by typing "E" followed by <CR>. The Edit Menu may be used to edit:

1) A Census long form
2) A Census short form
3) A Census follow-up form

Census long and short forms are accessed by the "Building Number", e.g. PAC0001. The user may identify the building numbers on file by printing a list of buildings in the Print mode. (A list of buildings may be displayed on the screen by executing a null query, i.e. running the Query option, but answering "N" to all questions so that all buildings will be displayed on the screen.)

Follow-up forms are accessed by the date of entry. For the convenience of the user, a list of all follow-up entry dates will be displayed on the screen after the building number has been entered.

The Edit mode is designed to allow modular access to the files via a sub-menu which displays the following organization:

LONG FORM

Building Data (Part I)
Material Data (Cont. I)
Study Area Data (Part II)
Treatment Data (Part III)
Enter Data
Follow-Up Data
Return to Main Menu

This process allows highly selective access to portions of the file so that any part of the Census file can be quickly accessed and changed. As each screen or item of information is edited, the user may save the data and go on by pressing the "ESC" key. Three choices will be presented to the user:

- Re-edit the screen on display
- Go on to the next screen in the module
- Quit and return to the sub-menu.
The Print option allows the user to print out hard copy reports of Census files. The specific paths within this option are controlled by a sub-menu which allows the user to do the following:

1) Print a Census long form
2) Print a Census short form
3) Print all follow-up forms for a building
4) Print a list of all buildings in the Census program

The Census reports are designed to resemble the Census forms as closely as possible. The Census long form prints out in the order of entry, and each masonry study area is printed followed by all of the treatments which apply to that area. The building list prints the name, location, and building number for each building.

The selection of the report type to be printed will trigger a prompt asking for the building number of the Census report to be printed. When the number is entered there is a pause to allow the user to set up the printer, i.e. to make sure it is turned on, on-line and set to the top of the form (TOF).

(Note to Users: While there is no option to print a blank form at this time, the user can manipulate the system into printing a blank form in the following way. Create a Census file of the appropriate form to be printed, and enter one or two blank spaces in each field in the form. Once this entry has been completed, the Print option may be used to print the dummy building report which will contain all blanks, and may be used as a basic input form.)
UTILITIES OPTION

File management may be accomplished in several ways, such as printing a list of buildings via the "PRINT" option on the Main Menu. Two other important options have been built into the "Utilities Menu". They are:

1) Deleting a building from the Census
2) Deleting a follow-up form
3) Converting a "short" form to a "long" form

DELETE OPTION:

The Delete option is very straightforward; it will delete a Census long form or short form. All data in the file will be erased and the building number will be deleted from the record. Unless the building number deleted is the last number assigned within the State, the number cannot be reused. If the building deleted was the last building entered within its State, the number will be reassigned to the next building entered in that State.

OPTION TO CONVERT SHORT FORMS TO LONG FORMS

The Census Short Form is an abbreviated report which allows the program to track treatments to buildings where the full Census cannot be completed. The software program allows the entry, storage, printing, searching and deleting of "Short Forms". It is possible that at some point after a Short Form is completed and entered, the full amount of Census data will become available. Under the Utilities Menu, a Short Form may be converted to a long Census form.

The Conversion option will take the data on record in a Short Form and transfer it to the appropriate matching field in the long Census form. It also changes the building number to indicate a Census building and deletes the Short Form number. For example, a Short Form in Washington, D.C. (DCS0041) would be converted and the number in the directory or building list would become DCC0041. The Census data could then be entered to complete a long form via the "Edit" option.
QUERY OPTION

Data in the Census program may be searched for analytical and file management purposes. A query option has been designed to allow searches by the following parameters:

1) Location:

Buildings may be searched on three levels, by:

   a) City
   b) State
   c) NPS Region

If no location is specified, all buildings on file will be included in the search.

2) Materials:

Buildings may be searched by up to three materials. The user simply follows the screen prompts and enters one to three materials in appropriate blocks on the screen. The buildings are searched across the materials in the "Masonry Study Area" field.

3) Treatments:

Buildings may be searched by up to three treatments. The treatments may be combined with materials or any other search parameters.

4) Other Options:

The search may be further restricted by three types of buildings of interest to the programs of the PAD. Buildings may be searched by whether they are:

   1) Tax Act Projects
   2) National Historic Landmarks
   3) Federally Owned Properties

Queries are executed by selecting the "Query" option and answering the questions displayed in sequence on the screen. The results of the query will be displayed on the screen, one screen at a time.

Screens may be printed by using the "Shift Key" - "Print Screen" option of DOS, i.e. hold down the shift key and touch the "PrtSc" key.

A quick technique for looking up a directory of buildings on the screen is to run a search and answer "N" to all questions. A list of all files in the program will be displayed on the screen.
CONCLUSION

The Census program is designed to provide maximum flexibility and user-friendliness in the automation of the Census of Treated Historic Masonry Buildings. Additional options and enhancements may be added to the menus as program needs require them. The microcomputer software program "Rbase" may be used to perform a variety of manipulations and reports by a user with "Rbase" experience. This allows the option of performing sophisticated data manipulation, search and analysis without additional programming. Routine system use, data entry, edit and query may be performed by personnel with minimal training and experience using the menu-driven software.

Questions and comments on the Census software program should be addressed to:

Director
Center for Architectural Conservation
College of Architecture
Georgia Institute of Technology
Atlanta, Georgia 30332
Phone: (404) 894-3390
APPENDIX A

NATIONAL REGISTER STATUS CODES
(To be used in response to Census Item #16)
CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

**NATIONAL REGISTER STATUS CODES**

These codes are to be used to answer item 16 in the Census forms. (Use of codes 0 through 6 will result in compatibility with NPS Real Property Coding for internal buildings.)

<table>
<thead>
<tr>
<th>CODE</th>
<th>CODE VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Undetermined R Status</td>
</tr>
<tr>
<td>1</td>
<td>Entered on Register</td>
</tr>
<tr>
<td>2</td>
<td>Entered - Undocumented</td>
</tr>
<tr>
<td>3</td>
<td>Determined Eligible - Keeper of National Register</td>
</tr>
<tr>
<td>4</td>
<td>Determined Ineligible - Keeper of National Register</td>
</tr>
<tr>
<td>5</td>
<td>Determined Eligible - State Historic Preservation Officer</td>
</tr>
<tr>
<td>6</td>
<td>Determined Ineligible - State Historic Preservation Officer</td>
</tr>
<tr>
<td>7</td>
<td>Listed in an NR District</td>
</tr>
<tr>
<td>8</td>
<td>National Historic Landmark</td>
</tr>
<tr>
<td>9</td>
<td>National Monument</td>
</tr>
</tbody>
</table>

**THE FOLLOWING APPLY TO NON-FEDERAL LISTINGS WHICH MEET NR CRITERIA**

<table>
<thead>
<tr>
<th>CODE</th>
<th>CODE VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>State Register - Individual</td>
</tr>
<tr>
<td>11</td>
<td>State Register - District</td>
</tr>
<tr>
<td>12</td>
<td>Local Register - Individual</td>
</tr>
<tr>
<td>13</td>
<td>Local Register - District</td>
</tr>
</tbody>
</table>
CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

TECHNICAL MANUAL

DEVELOPED FOR:

PRESERVATION ASSISTANCE DIVISION
UNITED STATES NATIONAL PARK SERVICE
DEPARTMENT OF INTERIOR
WASHINGTON, D.C. 20240

PREPARED BY:

CENTER FOR ARCHITECTURAL CONSERVATION
COLLEGE OF ARCHITECTURE
GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA, GA 30332
INTRODUCTION

This Technical Manual is provided as a supplement to the Census of Treated Historic Masonry Buildings, "Software User's Guide". This volume provides the database structures and program source code necessary to evaluate and maintain the program.

The Census software is a stand-alone, menu-driven program reflecting the organization and structure of the Census. This design promotes an easy transition from the manual approach used in the early Census to this automated Census program. Users familiar with the Census objectives, structure and procedures will intuitively recognize the content and structure of the Census software.

The first part of this volume is a listing of all database "table" structures. Census data is maintained in tables compatible with the proprietary database management system (DBMS), Rbase 5000. By design, in addition to the menu-driven routines, the Census program data can be queried on an ad hoc basis, reported and edited using Rbase 5000, by users familiar with that software.

The second part of this volume is a listing of Census program source code. All program routines are listed out. The program is written in Microsoft Pascal, Version 1.0, for flexibility, speed and efficiency. The program is compiled with the Microsoft Pascal Compiler, Version 1.0. Questions concerning the program should be directed to:

Director
Center for Architectural Conservation
College of Architecture
Georgia Institute of Technology
Atlanta, Georgia 30332
SECTION A

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

DATABASE FILE STRUCTURES
<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>ENDDATE</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>BNAME</td>
<td>TEXT</td>
<td>40 characters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ONAME</td>
<td>TEXT</td>
<td>40 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>OSTREET</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>OCITY</td>
<td>TEXT</td>
<td>20 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>OSTATE</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>OZIP</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>OPHONE</td>
<td>TEXT</td>
<td>14 characters</td>
<td></td>
</tr>
</tbody>
</table>
Table: FOLLOW2
Read Password: NO
Modify Password: NO

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>ENTDATE</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>TRTDATE</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>TRTLOC</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>TRTTYPE</td>
<td>TEXT</td>
<td>90 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MASTYPE</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>PROD1</td>
<td>TEXT</td>
<td>60 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>PROD2</td>
<td>TEXT</td>
<td>60 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>MATCHUNG</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>CHNGDES</td>
<td>TEXT</td>
<td>180 characters</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>AREADES</td>
<td>TEXT</td>
<td>180 characters</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>LAPSETM</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>REPTRT</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>REPDATE</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>FUTTRT</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>FUTDES</td>
<td>TEXT</td>
<td>180 characters</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>COMMENTS</td>
<td>TEXT</td>
<td>180 characters</td>
<td></td>
</tr>
</tbody>
</table>
Table: FOLLOW3
Read Password: NO
Modify Password: NO

Column definitions
# Name    Type    Length     Key
1 BNUMBER TEXT     8 characters yes
2 ENTDATE TEXT     8 characters yes
3 RNAME TEXT      40 characters
4 RORGAN TEXT     30 characters
5 RTITLE TEXT      30 characters
6 RSTREET TEXT     30 characters
7 RCITY TEXT      20 characters
8 RSTATE TEXT      4 characters
9 RZIP TEXT       10 characters
10 RPHONE TEXT     14 characters
Table: TREATMENT
Read Password: NO
Modify Password: NO

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>MASTYPE</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>NUMBER</td>
<td>INTEGER</td>
<td>1 value(s)</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>TRTTYPE</td>
<td>TEXT</td>
<td>90 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>TNUM</td>
<td>INTEGER</td>
<td>1 value(s)</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>BEGDATE</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>ENDDATE</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>TESTED</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>TESTDES</td>
<td>TEXT</td>
<td>180 characters</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>PROD1</td>
<td>TEXT</td>
<td>60 characters</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>PROD2</td>
<td>TEXT</td>
<td>60 characters</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>PROD3</td>
<td>TEXT</td>
<td>60 characters</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>temp</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>humidity</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>precip</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>AREATRT</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>APPEAR</td>
<td>TEXT</td>
<td>180 characters</td>
<td></td>
</tr>
</tbody>
</table>
Table: STUDY1
Read Password: NO
Modify Password: NO

Column definitions

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>STUDYLOC</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MASTYPE</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>NUMBER</td>
<td>INTEGER</td>
<td>1 value(s)</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>MASQRRY</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MASCOUR</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>MASFIN</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MASUSE</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>MASCOND</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>MASCOLR</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>MASDENS</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>MASPOROS</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>MASTREN</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>ADJMAT1</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>ADJMAT2</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>ADJMAT3</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>MORTTYPE</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>MORTCOLR</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>MORTSOFT</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>JOINTYPE</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>JOINTDEP</td>
<td>TEXT</td>
<td>16 characters</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>MORTCOND</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>MORTANAL</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
</tbody>
</table>
Table: STUDY2
Read Password: NO
Modify Password: NO

<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>MASTYPE</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>NUMBER</td>
<td>INTEGER</td>
<td>1 value(s)</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>UNTRETCD</td>
<td>TEXT</td>
<td>60 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>COATCOND</td>
<td>TEXT</td>
<td>60 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MOISTPRB</td>
<td>TEXT</td>
<td>60 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>TRT1</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>TRTDATE1</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>TRT2</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>TRTDATE2</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>TRT3</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>TRTDATE3</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>TRT4</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>TRTDATE4</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>SPECS</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>PRECOND</td>
<td>TEXT</td>
<td>180 characters</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>DETDES</td>
<td>TEXT</td>
<td>180 characters</td>
<td></td>
</tr>
</tbody>
</table>
Table: IDENT1
Read Password: NO
Modify Password: NO

Column definitions

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>BNAME</td>
<td>TEXT</td>
<td>40 characters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ALTNAME</td>
<td>TEXT</td>
<td>40 characters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>BSTREET</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>BCITY</td>
<td>TEXT</td>
<td>20 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>BCOUNTY</td>
<td>TEXT</td>
<td>20 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>BSTATE</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>BZIP</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>BUTM</td>
<td>TEXT</td>
<td>18 characters</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>NPSREG</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>FEDOWN</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>NHL</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>TRA</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>HABSNO</td>
<td>TEXT</td>
<td>20 characters</td>
<td>*</td>
</tr>
</tbody>
</table>

* HABSNO is the field name for National Register Number which appears in the Census and Short-form screens and reports. Enough characters have been allowed to add a parenthetical note if desired, following the N.R. number.
Table: IDENT2
Read Password: NO
Modify Password: NO

Column definitions

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>OSHIP</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ONAME</td>
<td>TEXT</td>
<td>40 characters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>OSTREET</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>OCITY</td>
<td>TEXT</td>
<td>20 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>OSTATE</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>OZIP</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>OPHONE</td>
<td>TEXT</td>
<td>14 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>MANAGER</td>
<td>TEXT</td>
<td>40 characters</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>MSTREET</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>MCITY</td>
<td>TEXT</td>
<td>20 characters</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>MSTATE</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>MZIP</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>MPHONE</td>
<td>TEXT</td>
<td>13 characters</td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>Name</td>
<td>Type</td>
<td>Length</td>
<td>Key</td>
</tr>
<tr>
<td>---</td>
<td>---------</td>
<td>-------</td>
<td>-----------</td>
<td>---------</td>
</tr>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>OCCSTAT</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>CURUSE</td>
<td>TEXT</td>
<td>70 characters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ACCESS</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>CONST</td>
<td>TEXT</td>
<td>20 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>FEDFUND</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>NRSTAT</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>SF</td>
<td>INTEGER</td>
<td>1 value(s)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>stories</td>
<td>TEXT</td>
<td>16 characters</td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>Name</td>
<td>Type</td>
<td>Length</td>
<td>Key</td>
</tr>
<tr>
<td>----</td>
<td>--------</td>
<td>--------</td>
<td>---------------------</td>
<td>---------</td>
</tr>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>RNAME</td>
<td>TEXT</td>
<td>40 characters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>RTITLE</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>RORGAN</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>RSTREET</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>RCITY</td>
<td>TEXT</td>
<td>20 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>RSTATE</td>
<td>TEXT</td>
<td>4 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>RZIP</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>RPHONE</td>
<td>TEXT</td>
<td>14 characters</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>ENTDATE</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
</tbody>
</table>
Table: MAT1  
Read Password: NO  
Modify Password: NO

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>CONSTDT</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ARCH</td>
<td>TEXT</td>
<td>34 characters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>BUILDER</td>
<td>TEXT</td>
<td>34 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ALTDATETIME</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>ALTARCH1</td>
<td>TEXT</td>
<td>34 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>ALTBLDR1</td>
<td>TEXT</td>
<td>34 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>ALTDATETIME</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>ALTARCH2</td>
<td>TEXT</td>
<td>34 characters</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>ALTBLDR2</td>
<td>TEXT</td>
<td>34 characters</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ALTDATETIME</td>
<td>TEXT</td>
<td>8 characters</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>ALTARCH3</td>
<td>TEXT</td>
<td>34 characters</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>ALTBLDR3</td>
<td>TEXT</td>
<td>34 characters</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>STRUCSYS</td>
<td>TEXT</td>
<td>90 characters</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>ROOFTYPE</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>ROOFMAT</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>DRAINTYP</td>
<td>TEXT</td>
<td>30 characters</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>DRAINCON</td>
<td>TEXT</td>
<td>10 characters</td>
<td></td>
</tr>
</tbody>
</table>
Table: MAT2
Read Password: NO
Modify Password: NO

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BNUMBER</td>
<td>TEXT</td>
<td>8 characters</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>MASTYPE1</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MASQRRY1</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MASCOUR1</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>MASFIN1</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MASUSE1</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>MASLOC1</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MASCOND1</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>MASCOLR1</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>MASTYPE2</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>MASQRRY2</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>MASCOUR2</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>MASFIN2</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>MASUSE2</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>MASLOC2</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>MASCOND2</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>MASCOLR2</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>MASTYPE3</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>MASQRRY3</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>MASCOUR3</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>MASFIN3</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>MASUSE3</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>MASLOC3</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>MASCOND3</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>MASCOLR3</td>
<td>TEXT</td>
<td>24 characters</td>
<td></td>
</tr>
</tbody>
</table>
SECTION B

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS
LISTING OF PROGRAM SOURCE CODE
implementation of census;

    function vdoxqq extern;
    function nulli extern;
    procedure rbstrt extern;
    procedure rbstat i extern;
    procedure rbopen i extern;
    procedure rbfind i extern;
    procedure rbwher i extern;
    procedure rbsorti extern;
    procedure rbget i extern;
    procedure rbputi extern;
    procedure rbdel i extern;
    procedure rbloadi extern;
    procedure rbclosi extern;
    procedure rbuseri extern;
    procedure rbseti extern;
    procedure rbwhri2i extern;
    procedure intmvoi extern;
    procedure lnbci extern;
    procedure intcad i extern;
    procedure datel extern;
    procedure strmv i extern;
    procedure tmei extern;
    procedure scrinti extern;
    procedure sphini extern;
    procedure cndyi extern;
    procedure rbformi extern;
    procedure rbdrawi extern;
    procedure rbedit i extern;
procedure beep; extern;

procedure bright;
begin;
    write(esc:1,'[1m');
end;

procedure clear_screen;
var [extern] vrBXqq, vrCXqq, vrDXqq: word;
var ax:word;
begin
    vrBXqq := byword(0700); \ [white on black text]
    vrCXqq := byword(ch,cl); \ [upper left corner of screen]
    vrDXqq := byword(dh,dl); \ [lower right corner of screen]
    ax := vdoxqq(byword(6,0));
end;

procedure blink;
begin;
    write(esc:1,'[5m');
end;

procedure under_line;
begin;
    writefesc:1,'[4m';
end;

procedure normal;
begin;
    writefesc:1,'[0m';
end;

procedure cancel;
begin;
    writefesc:1,'[8m';
end;

procedure reverse;
begin;
    write(esc:1,'[7m');
end;

procedure scr_loc;
var [extern] vrBXqq, vrCXqq, vrDXqq: word;
var ax:word;
begin
    vrBXqq := byword(010);
    vrCXqq := byword(row,col);
    ax := vdoxqq(byword(2,0));
    vrBXqq := byword(01att);
    vrCXqq := 1;
    ax := vdoxqq(byword(9,ich));
end;

procedure cur_pos;
var [extern] vrBXqq, vrCXqq, vrDXqq: word;
var ax:word;
begin
    vrBXqq := byword(0,0);
    vrDXqq := byword(row,col);
end;
procedure scroll_up;
var [extern] vrBXqq, vrCXqq, vrDXqq: word;
var
  ax: word;
  ah, al, bh, bl, ch, cl, dh, dl: word;
begin
  ah := 6;
  al := 1;
  bh := 0;
  bl := 0;
  ch := wrd(ur);
  cl := wrd(uc);
  dh := wrd(lr);
  dl := wrd(lc);
  vrBXqq := byword(bh, bl);
  vrCXqq := byword(ch, cl);
  vrDXqq := byword(dh, dl);
  ax := vdxqq(byword(ah, al));
end;

procedure scroll_dn;
var [extern] vrBXqq, vrCXqq, vrDXqq: word;
var
  ax: word;
  ah, al, bh, bl, ch, cl, dh, dl: word;
begin
  ah := 7;
  al := 1;
  bh := 0;
  bl := 0;
  ch := wrd(ur);
  cl := wrd(uc);
  dh := wrd(lr);
  dl := wrd(lc);
  vrBXqq := byword(bh, bl);
  vrCXqq := byword(ch, cl);
  vrDXqq := byword(dh, dl);
  ax := vdxqq(byword(ah, al));
end;

procedure rbcheck;
begin
  rbstat(rbs);
  eor[p] := rbs[0];
  if eor[p] then
    begin
      cur_pos(24, 0);
      write('Failure in \textquote{operation} operation. Error = \textquote{rbs}, Path = \textquote{p}');
      cur_pos(0, 0);
    end;
  end;
procedure search;

type

  whttype = RECORD
    CASE INTEGER OF
      1: (STR: STRING(122));
      2: (INT: ARRAY[1..61] OF INTEGER);
    END;
  ENO;

  namel0 = array[1..8] of names;

var

  answer : char;
  natt1,natt7,natt8 : integer;
  quest : packed array[1..30] of char;
  one : integer;
  i : integer;
  temp,strlen : integer;
  wherel,where7,where8 : whttype;
  att11, att17, att18 : namel0;
  whera1, whera7, whera8 : shorts;
  wherop3, wherop7, wherop8 : namel0;
  ind1, ind7, ind8 : integer;
  ct : integer;

begin

  one := 1;
  ind1 := 1;
  natt1 := 0;
  quest := ' '; 
  temp := 0;

  for i := 1 to 5 do begin
    att17[i] := ' ';  
    att18[i] := ' ', 
    att11[i] := ' ',
  end;

  clear_screen(23,0,23,79);
  cur_pos(23,15);
  write('Do you want to restrict the search by a location? 

  if answer = 'Y' then begin
    cur_pos(12,1); write('CITY 
    cur_pos(14,1); write('STATE 
    cur_pos(16,1); write('REGION
    cur_pos(12,6); reverse; write(quest:15);
    cur_pos(12,6); readln(quest);
    if quest[1] = ' ' then begin
      cur_pos(12,6); normal; write(quest:15);
      cur_pos(14,7); reverse; write(quest:2);
      cur_pos(14,7); readln(quest);
      if quest[1] = ' ' then begin
        cur_pos(14,7); normal; write(quest:2);
        cur_pos(16,8); reverse; write(quest:2);
        cur_pos(16,8); readln(quest);
        if quest[1] = ' ' then begin
          cur_pos(16,8); normal; write(quest:2);
          cur_pos(23,15); blink;
          write('Invalid location; no specific location assumed');
          normal;
        end
      end
    end
  end

end.
end (if state not specified)
else begin

end (if city not specified)
else begin

strlen := 30;
attl[1] := 'BCITY '; lnbc(ads quest,one,strlen,temp); strlen := temp;
where1.int[ind1] := strlen;
if (strlen mod 2) <> 0 then strlen := strlen + 1;
for i := 1 to strlen do where1.str[i+2] := quest[i]; nattl := nattl + 1;
whereop1 := 'CONTAINS'; wherealo := 'AND '; ind1 := ind1 + (strlen div 2) + 1;
end; (if city specified)
end (if specific location desired)

ind7 := 1;
natt7 := 0;
for i := 1 to 3D do quest[i] := ";
clear_screen(210,23,79);
cur_pos(23,15); normal;
write('Do you want to search for a specific material? '); readln(answer);
if answer = 'Y' then begin

cur_pos(12,25); write('1 '); cur_pos(14,25); write('2 '); cur_pos(16,25); write('3 '); cur_pos(12,27); reverse; write(quest:20);
cur_pos(12,27); readln(quest); normal;
if quest[1] <> " then begin

natt7 := natt7 + 1;
strlen := 30;
att7[1] := 'MASTYPE '; where7[1] := 'CONTAINS'; wherea7[1] := 'OR '; lnbc(ads quest,one,strlen,temp);
strlen := temp;
where7.int[ind7] := strlen;
if (strlen mod 2) <> 0 then strlen := strlen + 1;
for i := 1 to strlen do

where7.str[(ind7*2)+1] := quest[i]; ind7 := ind7 + (strlen div 2) + 1;
cur_pos(12,27); reverse; write(quest:20);
cur_pos(12,27); readln(quest); normal;
end; (if city specified)
end (if specific location desired)
cur_pos(14,27); readln(quest): normal;
if quest[1] <> ' ' then begin
  natt7 := natt7 + 1;
  strlen := 30;
  att17[2] := 'MASTYPE ';
  wherop7[2] := 'CONTAINS';
  wherao7[1] := 'OR ';
  lnbc(ads quest,one,strlen,temp);
  strlen := temp;
  where7.int[ind7] := strlen;
  if ((strlen mod 2) <> 0) then strlen := strlen + 1;
  for i := 1 to strlen do
    where7.str[(ind7*2)+i] := quest[i];
  ind7 := ind7 + (strlen div 2) + 1;
  cur_pos(14,27); normal; write(quest:20);
  for i := 1 to 20 do quest[i] := '
  cur_pos(14,27); reverse; write(quest:20);
  cur_pos(14,27); readln(quest); normal;
  if quest[1] <> ' ' then begin
    natt7 := natt7 + 1;
    strlen := 30;
    att17[3] := 'MASTYPE ';
    wherop7[3] := 'CONTAINS';
    lnbc(ads quest,one,strlen,temp);
    strlen := temp;
    where7.int[ind7] := strlen;
    if ((strlen mod 2) <> 0) then strlen := strlen + 1;
    for i := 1 to strlen do
      where7.str[(ind7*2)+i] := quest[i];
    ind7 := ind7 + (strlen div 2) + 1;
  end (3 material types specified)
else begin
  cur_pos(16,27); write(quest:20);
end (2 material types specified)
else begin
  cur_pos(14,27); write(quest:20);
end (1 material type specified)
else begin
  cur_pos(12,27); write(quest:20);
end; (if material criteria desired)

ind8 := 1;
natt8 := 0;
for i := 1 to 30 do quest[i] := '
clear_screen(2370,23179);
cur_pos(23,15);
write('Do you want to search for a specific treatment?');
readln(answer);
if answer = 'Y' then begin
  cur_pos(12,45); write('1 ');
cur_pos(14,45); write('2 ');
cur_pos(16,45); write('3 ');
cur_pos(12,47); reverse; write(quest:30);
cur_pos(12,47); readln(quest); normal;
if quest[1] <> ' ' then begin
  strlen := 30;
  natt8 := natt8 + 1;
  att18[1] := 'TRTTYPE ';
  wherop8[1] := 'CONTAINS';
  wherao8[1] := 'OR ';
  lnbc(ads quest,one,strlen,temp);
  strlen := temp;
end; (if material criteria desired)
if ((strlen mod 2) > 0) then strlen := strlen + 1;
for i := 1 to strlen do
    where8.str[(ind8*2)+i] := quest[i];
ind8 := ind8 + (strlen div 2) + 1;
cur_pos(12,47); normal; write(quest:30);
for i := 1 to 30 do quest[i] := ' ';
cur_pos(14,47); reverse; write(quest:30);
cur_pos(14,47); readln(quest); normal;
if quest[1] <> ' ' then begin
    natt8 := natt8 + 1;
    strlen := 30;
    attl8[2] := 'TRTTYPE '
    wherop8[2] := 'CONTAINS'
    wherao8[2] := 'OR '
    Inbc(ads quest,one,strlen,temp);
    strlen := temp;
    where8.int[ind8] := strlen;
    if ((strlen mod 2) > 0) then strlen := strlen + 1;
    for i := 1 to strlen do
        where8.str[(ind8*2)+i] := quest[i];
    ind8 := ind8 + (strlen div 2) + 1;
cur_pos(14,47); normal; write(quest:30);
    for i := 1 to 30 do quest[i] := ";
cur_pos(16,47); reverse; write(quest:30);
cur_pos(16,47); readln(quest); normal;
    if quest[1] <> ' ' then begin
        natt8 := natt8 + 1;
        strlen := 30;
        attl8[3] := 'TRTTYPE '
        wherop8[3] := 'CONTAINS'
        Inbc(ads quest,one,strlen,temp);
        strlen := temp;
        where8.int[ind8] := strlen;
        if ((strlen mod 2) > 0) then strlen := strlen + 1;
        for i := 1 to strlen do
            where8.str[(ind8*2)+i] := quest[i];
        ind8 := ind8 + (strlen div 2) + 1;
    end [three treatment types selected]
else begin
    cur_pos(12,47); write(quest:30);
    end;
end [two treatment types selected]
else begin
    cur_pos(14,47); write(quest:30);
    end;
end [one treatment type selected]
else begin
    cur_pos(16,47); write(quest:30);
    end;
end [treatment criteria desired]

temp := 0;
clear_screen(22,0;23,79);
cur_pos(22,1); normal;
write('Do you want to limit the search to: ');
cur_pos(23,15);
write('Federally Owned Buildings (Y/N) ');
readln(answer);
if answer = 'Y' then begin
    natt1 := natt1 + 1;
    wherao1[natt1] := 'AND '
    attl1[natt1] := 'FEDOWN '
    wherop1[natt1] := 'EQ '
    where1.int[ind1] := 4;
end;
wherel.str[(ind1*2)+1] := answer;

ind1 := ind1 + (strlen div 2) + 1;
end;
clear_screen(23,0,23,79);
cur_pos(23,15);
write('National Historic Landmarks (Y/N) ');
readln(answer);
if answer = 'Y' then begin
  nattl := nattl + 1;
  wherao1[nattl] := 'AND ';
  attl1[nattl] := 'NHL ';
  wheropl1[nattl] := 'ED ';
  wherel.int[indl] := 4;
  for i := 2 to 4 do wherel.str[(ind1112)+i] := "
  wherel.str[(ind1112)+1] := answer;
  indl := indl + (strlen div 2) + 1;
end;
clear_screen(23,0,23,79);
cur_pos(23,15);
write('Tax Act Projects  (Y/N) ');
readln(answer);
if answer = 'Y' then begin
  nattl := nattl + 1;
  wherao1[nattl] := 'AND ';
  attl1[nattl] := 'IRA ';
  wheropl1[nattl] := 'EO ');
  wherel.int[indl] := 4;
  for i := 2 to 4 do wherel.str[(ind1112)+i] := "
  wherel.str[(ind1112)+1] := answer;
  indl := indl + (strlen div 2) + 1;
end;

if natt7 > 0 then begin
  wherao7[natt7] := 'AND ';
  natt7 := natt7 + 1;
  wherel.int[ind7] := 8;
  attl7[natt7] := 'BNUMBER ';
  wheropl7[natt7] := 'EQ ';
end;
if natt8 > 0 then begin
  wherao8[natt8] := 'AND ';
  natt8 := natt8 + 1;
  wherel.int[ind8] := 8;
  attl8[natt8] := 'BNUMBER ';
  wheropl8[natt8] := 'EQ ';
end;
clear_screen(10,0,23,79);
cur_pos(10,0);
bright;
write(' BLDG NO BUILDING NAME CITY STATE');
normal;
ct := 0;
rbfind(path[1],rename[1]);
if natt1 <> 0 then begin
  rbwhere(path[1],att11[1],wheropl1[1],ads where1,wherao1[1],natt1);
  rbstat(rbs);
end
else rbs := 0;
if rbs <> -1 then begin
  rbget(path[1],ads row1);
  rbstat(rbs);
while not eor[1] do begin
    if natt7 > 0 then begin  
        [criteria specified for materials]
        for i := 1 to 8 do where7.str[(ind7*2) + i] := rowl.bnumber[i];
        rbfind(path[2], relname[8]);
        rbwher(path[2], att17[1], wherop7[1], ads where7, wherao7[1], natt7);
        rbstat(rbs);
    if rbs <> -1 then begin
        if natt8 > 0 then begin  
            [criteria specified for treatment]
            for i := 1 to 8 do where8.str[(ind8*2) + i] := rowl.bnumber[i];
            rbfind(path[3], relname[7]);
            rbwher(path[3], att8[1], wherop8[1], ads where8, wherao8[1], natt8);
            rbstat(rbs);
        if rbs <> -1 then begin
            ct := ct + 1;
            cur_pos(ct + 10, 1);
            with rowl do
                write(bnumber:8, ', bname:40, ', bcity:20, ', bstate:2);
        end; (if rows found)
    end (if material specified)
    else begin
        if ct = 12 then
            scroll_up(11, 1, 23, 79)
        else
            ct := ct + 1;
            cur_pos(ct + 10, 1);
            with rowl do
                write(bnumber:8, ', bname:40, ', bcity:20, ', bstate:2);
        end; (if material not specified)
    end (if treatment specified)
else begin
    if natt8 > 0 then begin
        for i := 1 to 8 do where8.str[(ind8*2) + i] := rowl.bnumber[i];
        rbfind(path[3], relname[7]);
        rbwher(path[3], att8[1], wherop8[1], ads where8, wherao8[1], natt8);
        rbstat(rbs);
    if rbs <> -1 then begin
        if ct = 12 then
            scroll_up(11, 1, 23, 79)
        else
            ct := ct + 1;
        cur_pos(ct + 10, 1);
        with rowl do
            write(bnumber:8, ', bname:40, ', bcity:20, ', bstate:2);
    end; (if rows found)
end (else treatment not specified)
    rbget(path[1], ads row1);
    rbstat(rbs);
    eor[1] := rbs <> 0;
end; (while records for row1)
if ct = 0 then begin
    cur_pos(15, 51; bright; write('No Buildings Found for Specified Criteria');
end; 
cur_pos(23, 1); normal;
end [if rows found for location]
else begin
    cur_pos(15,5);
    bright; write('No Buildings Found for Specified Criteria');
    cur_pos(23,1); normal;
    write('Enter any key to continue');
end;
spchin(ch);
end [procedure]

procedure blist;
begin
    attna[e][1] := 'BNUMBER '; 
    natt_sort := 1;
    dirlist[1] := 0;
    bright; cur_pos(22,20); write('Prepare Printer for Building List Report');
    cur_pos(23,25); write('Press any key to continue');
    spchin(ch);
    assign(f,'lpt1:');
    rewrite(f);
    rbfind(path[2],rename[1]);
    rbcheck('rbfind ',2);
    rbsort(path[2],attna[e][1],natt_sort,dirlist[1]);
    rbcheck('rbsort ',2);
    writeln(f);
    writeln(f);
    writeln(f);
    writeln(f);
    writeln(f,' :42,'CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
    writeln(f);
    writeln(f,' :39,Programmed for the Preservation Assistance Division');
    writeln(f,' :51,U.S. National Park Service');
    writeln(f);
    writeln(f);
    writeln(f);
    writeln(f,' :52,LIST OF TREATED BUILDINGS');
    writeln(f);
    writeln(f);
    writeln(f);
    writeln(f,' :15,Building');
    writeln(f,' :10,# Number Building Name Building Address');
    writeln(f,' :8,'Building City, State');
    writeln(f,' :8,'---------------------------' );
    w := 0;
    repeat
        with row1 do begin
            w := w + 1;
            rbget(path[2],ads row1);
            rbstat(rbs);
            if rbs <> -1 then begin
                writeln(f,' :8,w4,' :3,bnumber:8,' :3,bname:40,' :3,bstreet:30,' :3,bcity:20,' :3,bstate:2);
            end;
        end;
        until rbs = -1;
    writeln(f,' :for_feed卣');
    writeln(f,' :nocow卣');
    close(f);
end;
end.
procedure open_db;
begin
  dbname := 'C:CENSUS';
  rbstrt;
  rbopen (dbname);
end;

procedure close_db;
begin
  rbclso;
end;

procedure b_prompt;
var
  b : packed array [1..4] of char;
  b2 : packed array [1..4] of char;
  b_temp : lstring(4);
  state_temp : packed array [1..2] of char;
  test : boolean;
label cont;
begin
  attlist[1] := 'BNUMBER';
  attname[1] := 'BNUMBER';
  wherop[1] := 'CONTAINS';
  wherea[1] := 'AND';
  natt := 1;
  natt_sort := 1;
  dirlist[1] := 0;
  cur_pos(22,14);
  writeln('Enter the TWO LETTER State Code of the building surveyed: ');
  cur_pos(22,71);readln(state_temp);
  u := 0;
  z := 0;
  [ WRIITELN('*',STATE TEMP,'*'); ]
  where.int[1] := 2;
  for w := 1 to 2 do begin where.str[w+2] := state_temp[w];write('*',where.str[w+2],'*'); end;
  rbfind(path[1],relname[1]);
  rbcheck('rbfind ','1');
  rbwher(path[1],attlist[1],wherap[1],ads where,wherea[1],natt);
  rbstat(rbs);
  if rbs = -1 then begin
    b2[1] := '0';
    b2[2] := '0';
    b2[3] := '0';
    b2[4] := '0';
    goto cont;
  end;
  rbsort(path[1],attname[1],natt_sort,dirlist[1]);
  repeat
    rbget(path[1],ads row1);
    rbstat(rbs);
  [write1n(row1.bnumber)];
  until rbs = -1;
  b2[1] := row1.bnumber[4];
  b2[2] := row1.bnumber[5];
  b2[3] := row1.bnumber[6];
  b2[4] := row1.bnumber[7];
cont:
  copylist(b2,b_temp);
{writeln(u;b_temp)test};
  u := u + 1;
  test := encode(b_temp,u);{writeln(u;b_temp)test};
  v := 1;
  repeat
    if b_temp[v] = ' ' then b_temp[v] := '0';
    v := v + 1;
  until v = 4;
  bnum[1] := state_temp[1];
  case temp of
    otherwise
    end;
  bnum[4] := b_temp[1];
  bnum[7] := b_temp[4];
  bnum[8] := ' ';
  writeln(bnum);{writeln(u;b_temp)test};
  where.int[1] := 8;
  for w := 1 to 8 do where.str[w+2] := bnum[w];
end;
procedure entdate_prompt;
var ent : packed array [1..8] of char;
begin
  clear_screen(10,0;24,79);
  bright;cur_pos(9;35);write('ENTRY DATES');normal;
  rbfind(path[4],relname[6]);
  rbwher(path[4],atts[1],wherop[1],ads where,whera[1],natts);
  rbstat(rbs);
  if rbs <> -1 then begin
    n := 0;
    repeat
      rbget(path[4],ads row6);
      rbstat(rbs1);
      if rbs1 <> -1 then begin
        n := n + 1;
        cur_pos(n+10;35);write(row6.entdate;8);
      end;
    until rbs1 = -1;
    cur_pos(23;14);
    z := 0;
    writeln('Enter the Follow-up Entry Date (i.e. 04/12/85): ');
    cur_pos(23;62);readln(ent);
    u := 0;
    z := 0;
    rbs1 := 0;
    where.int[6] := 8;
    for w := 1 to 8 do where.str[w+2] := ent[w];
  end else begin
    cur_pos(18;25);write('NO FOLLOW_UP REPORTS FOR THIS BUILDING');
    rbs1 := -1;
  end;
end;
procedure show_bnum;
begin
  bright:;}
write(bnum);
normal;

end;

procedure show_mas;
begin
  bright;
  cur_pos(24,60);
  if fmen = 'E' or else fmen = 'e' then
    write('MAS: ',row8.number:2,')
  else write('MAS: ',mas:2,' 	 ');
normal;
end;

procedure show_treat;
begin
  bright;
  cur_pos(24,68);
  if fmen = 'E' or else fmen = 'e' then
    write('TREAT: ',row7.tnum:2)
  else write('TREAT: ',treat:2);
normal;
end;

procedure load_sident1;
lable exit:
begin
  row1.bnumber := bnum;
  if fmen = 'E' or else fmen = 'e' then
    rbput(path[5],ads row1)
  else begin
    rbfind(path[5],relname[1]);
    rbcheck('rbfind 1',5);
    rbload(path[5],ads row1);
    rbstat(rbs);
    if rbs <> 0 then
      begin
        clscrn;
        cur_pos(24,0);bright;
        write('Error on load, status=',rbs);
        normal;
        goto exit;
      end;
edfunc := 0;
  end;
exit:
end;

procedure load_sident2;
lable exit:
begin
  row2.bnumber := bnum;
  if fmen = 'E' or else fmen = 'e' then
    rbput(path[2],ads row2)
  else begin
    rbfind(path[2],relname[2]);
    rbcheck('rbfind 2',2);
    rbload(path[2],ads row2);
    rbstat(rbs);
    if rbs <> 0 then
      begin
        clscrn;
        cur_pos(24,0);bright;
        write('Error on load, status=',rbs);
        normal;
        goto exit;
      end;
edfunc := 0;
  end;
exit:
end;
write('Error on load, status=',rbs);
normal;
goto exit;
end;
edfunct := 0;
end;
exit:
end;
procedure load_sident3;

label exit;

begin
row3.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[2],ads row3)
else begin
  rbfind(path[2],rename[3]);
  rbcheck('rbfind 3',2);
  rbload(path[2],ads row3);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clscrn;
      cur_pos(24,0);bright;
      write('Error on load, status=',rbs);
      normal;
      goto exit;
    end;
edfunct := 0;
end;
exit:
end;
procedure load_smat1;

label exit;

begin
row9.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[2],ads row9)
else begin
  rbfind(path[2],rename[9]);
  rbcheck('rbfind 4',2);
  rbload(path[2],ads row9);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clscrn;
      cur_pos(24,0);bright;
      write('Error on load, status=',rbs);
      normal;
      goto exit;
    end;
edfunct := 0;
end;
exit:
end;
procedure load_smat2;

label exit;

begin
row10.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[2],ads row10)
else begin
  rbfind(path[2],rename[10]);
  rbcheck('rbfind 5',2);
  rbload(path[2],ads row10);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clscrn;
      cur_pos(24,0);bright;
      write('Error on load, status=',rbs);
      normal;
      goto exit;
    end;
edfunct := 0;
end;
exit:
end;
else begin
  rbfind(path[2], relname[10]);
  rbcheck('rbfind 5', 2);
  rbload(path[2], ads row10);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clscrn;
      cur_pos(24, 0); bright;
      write('Error on load, status=', rbs);
      normal;
      goto exit;
    end;
  edfunct := 0;
end;
exit:
end;

procedure load_sstudy1;
label exit;
begin exit;
begin
  row8.bnumber := bnum;
  if fmen = 'E' or else fmen = 'e' then
    rbput(path[2], ads row8)
  else begin
    row8.number := mas;
    rbfind(path[2], relname[8]);
    rbcheck('rbfind 6', 2);
    rbload(path[2], ads rowel);
    rbstat(rbs);
    if rbs <> 0 then
      begin
        clscrn;
        cur_pos(24, 0); bright;
        write('Error on load, status=', rbs);
        normal;
        goto exit;
      end;
    edfunct := 0;
  end;
end;
exit:
end;

procedure load_sstudy2;
label exit;
begin
if fmen = 'E' or else fmen = 'e' then
  rbput(path[3], ads row11)
else begin
  row11.number := mas;
  row11.bnumber := bnum;
  row11.mastype := row8.mastype;
  rbfind(path[3], relname[11]);
  rbcheck('rbfind 7', 3);
  rbload(path[3], ads row11);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clscrn;
      cur_pos(24, 0); bright;
      write('Error on load, status=', rbs);
      normal;
      goto exit;
    end;
  edfunct := 0;
end;
exit:
end;
goto exit;
edfunct := 0;
end;
exit;
end;

procedure load_streat1;

label exit;

begin
row7.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
    rbput(path[4],ads row7)
else begin
    row7.tnum := treat;
    row7.number := row11.number;
    row7.mastype := row8.mastype;
    rbfind(path[4],relname[7]);
    rbcheck('rbfind 8','4);
    rblload(path[4],ads row7);
    rbstat(rbs);
    if rbs <> 0 then
        begin
            clscrn;
            cur_pos(24,0);bright;
            write('Error on load, status=',rbs);
            normal;
            goto exit;
        end;
edfunct := 0;
end;
exit:
end;

procedure load_sident4;

label exit;

begin
row4.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
    rbput(path[2],ads row4)
else begin
    rbfind(path[2],relname[4]);
    rbcheck('rbfind 9','2');
    rblload(path[2],ads row4);
    rbstat(rbs);
    if rbs <> 0 then
        begin
            clscrn;
            cur_pos(24,0);bright;
            write('Error on load, status=',rbs);
            normal;
            goto exit;
        end;
edfunct := 0;
end;
exit:
end;

procedure senter_edit(formname:names; route:integer);

var count : integer;

label exit;

count := 1;
rftform(path[1], forname, ads formbuf);
rbsstat(rbs);
if rbs <> 0 then
  begin
    cur_pos(24,0); bright;
    writeln('Error on form initialize, status=',rbs);
    normal;
    goto exit;
  end;

if fmen = 'E' or else fmen = 'e' then begin
  case route of
    1:
    2:
    3:
    4:
    6:
    8:
    9:
  end;
end;

begin rbget(path[5], ads row1); rbstat(rbs);
begin rbget(path[2], ads row2); rbstat(rbs);
begin rbget(path[2], ads row3); rbstat(rbs);
begin rbget(path[2], ads row9); rbstat(rbs);
begin rbget(path[2], ads row8); rbstat(rbs);
begin rbget(path[2], ads row4); rbstat(rbs);

if rbs = -1 then goto exit;

end; loop, reading new rows with }

while true do begin
  z := 0;
  repeat
    case route of
      1:
      2:
      3:
      4:
      6:
      8:
      9:
    end;
  end;

  bright;
  cur_pos(24,0);
  write('Press [ESC] when done');
  normal;
  count := 0;
  show_bnum;
  if mas <> 0 then show_mas;
  if treat <> 0 then show_treat;
  case route of
    1:
    2:
    3:
    4:
    6:
    8:
    9:
  end;

rbstat(rbs);
dataerr := rbs (> 0;
if dataerr then begin
  if rbs = 34 then
    begin
      cur_pos(24,0);
      bright;
      end;

edfunct := 1;
cur_pos(1,0);
end
else
begin
clscrn;
cur_pos(24,0); bright;
writeln('Error on data entry, status=', rbs);
normal;
goto exit;
end;
end else z := 1;
until z = 1;
while true do
begin
cur_pos(24,0); bright;
write('N(ext), E(dit again), Q(uit)?
');
normal;
cur_pos(24,29);
spchin(ch);
case chr(ch) of
'q', 'Q':
begin
clscrn;
rbclos;
goto exit;
end;
'y', 'Y':
begin
was := 0;
treat := 0;
goto exit;
end;
'n', 'N':
begin
if not dataerr then
begin
begin
case route of
1 : load_sident1;
2 : load_sident2;
3 : load_sident3;
4 : begin load_smat1; load_smat2; end;
6 : begin load_sstudy1; load_sstudy2; end;
8 : load_sreat1;
9 : load_sident4;
otherwise
cur_pos(24,0);
write('');
goto exit;
end;
end
else
begin
bright; blink;
cur_pos(24,0);
write('Cannot load illegal data');
cur_pos(24,25);
spchin(ch);
end;
end;
e', 'E':
begin
end
end;
break;
end;
otherwise
beep;
end;

procedure add_short;
var route: integer;
    screen: integer;
label exit:
begin
    mas := 0;
treat := 0;
edfunct := 0;
attlist[1] := 'BNUMBER '
whereop[1] := 'EQ 
whereao[1] := 'AND '
natt := 1;

b_prompt;
senter_edit('SIDENT1 ',1);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 0;
senter_edit('SIDENT2 ',2);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 0;
senter_edit('SIDENT3 ',3);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 0;
senter_edit('SMAT1 ',4);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
it := 0;
repeat
    loop_name := 'Study Area';
it2 := 0;
treat := 0;
mas := mas + 1;
edfunct := 0;
senter_edit('STUDY1 ',6);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
repeat
    treat := treat + 1;
edfunct := 0;
senter_edit('STREAT1 ',8);
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
    bright;
cur_pos(24,0);
write('Input Another Treatment (Y/N)? ');
normal;
cur_pos(24,29);
spchin(ch);
    if chr(ch) = 'n' or else chr(ch) = 'N' then it2 := 1;
until it2 = 1;
bright;
cur_pos(24,0);

normal;
cur_pos(24,35);
s贴近(ch);
  if chr(ch) = 'n' or else chr(ch) = 'N' then it := 1;
until it = 1;
edfunct := 0;
senter_edit('SIDENT4 ',9);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
exit:
(load_building)
end;

procedure edit_sident;
var route : integer;
screen: integer;
label exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ ';
wherao[1] := 'AND ';
natt := 1;
rblind(path[5],relname[1]);
rbher(path[5],attlist[1],wherop[1], ads where,wherao[1],natt);
senter_edit('SIDENT1 ',I);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 1;
rblind(path[2],relname[2]);
rbher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
senter_edit('SIDENT2 ',2);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 1;
rblind(path[2],relname[3]);
rbher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
senter_edit('SIDENT3 ',3);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
exit:
end;
procedure edit_smat;
var route : integer;
screen: integer;
label exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherap[1] := 'EQ ';
wherao[1] := 'AND ';
natt := 1;
rblind(path[2],relname[9]);
rbher(path[2],attlist[1],wherap[1], ads where,wherao[1],natt);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
end;

procedure edit_sstudy;
var    route : integer;
       screen: integer;
label  exit;
begin
  mas := 0;
  treat := 0;
  edfunct := 1;
  attlist[1] := 'NUMBER '
  wherop[1] := 'EQ '
  wherao[1] := 'AND '
  natt := 1;
  rbsl := 0;

  rbfind(path[2].rename[8]);
  rbwher(path[2].attlist[1],wherop[1], ads where,wherao[1],natt);
  repeat
    mas := mas + 1;
    senter_edit('SSTUDY1 ',6);
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
  until rbsl = -1;
exit:
end;

procedure edit_streat;
var    route : integer;
       screen: integer;
       mastype : string(24);
label  exit;
begin
  mas := 1;
  treat := 0;
  edfunct := 1;
  attlist[1] := 'NUMBER '
  wherop[1] := 'EQ '
  wherao[1] := 'AND '
  natt := 1;
  rbsl := 0;
  edfunct := 1;
  rbfind(path[4].rename[7]);
  rbwher(path[4].attlist[1],wherop[1], ads where,wherao[1],natt);
  repeat
    senter_edit('STREAT1 ',8);
    mastype := row7.mastype;
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
  until rbsl = -1;
exit:
end;

procedure edit_sentry;
var 
route : integer;
screen: integer;

label exit:

begin
mas := 0;
treat := 0;
edfunc := 1;
attlist[1] := 'SNUMBER,'
wherop[1] := 'EQ '
wherao[1] := 'AND '
natt := 1;
rbsl := 0;
edfunc := 1;
rbfind(pathE21,relname[4]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
senter_edit('SIDENT4 ',9);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbsl = -1;
exit:
end;
procedure load_ident1;

label exit;

begin
row1.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[5],ads row1)
else begin
  rbfind(path[5],relname[1]);
  rbcheck('rbfind 1',5);
  rbload(path[5],ads row1);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clrscr;
      cur_pos(24,0); bright;
      write('Error on load, status=',rbs);
      normal;
      goto exit;
    end;
edfunct := 0;
end;
exit:
end;

procedure load_ident2;

label exit;

begin
row2.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[2],ads row2)
else begin
  rbfind(path[2],relname[2]);
  rbcheck('rbfind 2',2);
  rbload(path[2],ads row2);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clrscr;
      cur_pos(24,0); bright;
      write('Error on load, status=',rbs);
      normal;
      goto exit;
    end;
edfunct := 0;
end;
exit:
end;

procedure load_ident3;

label exit;

begin
row3.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[3],ads row3)
else begin
  rbfind(path[3],relname[3]);
  rbcheck('rbfind 3',3);
  rbload(path[3],ads row3);
  rbstat(rbs);
  if rbs <> 0 then
procedure load_mat1;

label exit;

begin
row9.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[2],ads row9)
else begin
  rbfind(path[2],relname[9]);
  rbcheck('rbfind 4',2);
  rbload(path[2],ads row9);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clscrn;
      cur_pos(24,0);bright;
      write('Error on load, status=',rbs);
      normal;
      goto exit;
    end;
edfunct := 0;
end;
exit:
end;

procedure load_mat2;

label exit;

begin
row10.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[2],ads row10)
else begin
  rbfind(path[2],relname[10]);
  rbcheck('rbfind 5',2);
  rbload(path[2],ads row10);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clscrn;
      cur_pos(24,0);bright;
      write('Error on load, status=',rbs);
      normal;
      goto exit;
    end;
edfunct := 0;
end;
exit:
end;

procedure load_studyl;

label exit;
begin
row8.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[2],ads row8)
else begin
  row8.number := mas;
  rbfind(path[2],relname[8]);
  rbcheck('rbfind 6';2);
  rbload(path[2],ads row8);
  rbstat(rbs);
  if rbs > 0 then
  begin
    clrscrn;
    cur_pos(24,0);bright;
    write('Error on load, status=',rbs);
    normal;
    goto exit;
  end;
edfunct := 0;
end;
exit:
end;

procedure load_study2;

label exit:

begin
row11.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[3],ads row11)
else begin
  row11.number := mas;
  row11.mastype := row8.mastype;
  rbfind(path[3],relname[11]);
  rbcheck('rbfind 7';7);
  rbload(path[3],ads row11);
  rbstat(rbs);
  if rbs > 0 then
  begin
    clrscrn;
    cur_pos(24,0);bright;
    write('Error on load, status=',rbs);
    normal;
    goto exit;
  end;
edfunct := 0;
end;
exit:
end;

procedure load_treat1;

label exit:

begin
row7.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[4],ads row7)
else begin
  row7.tnum := treat;
  row7.number := row11.number;
  row7.mastype := row8.mastype;
  rbfind(path[4],relname[7]);
  rbcheck('rbfind 8';8);
  rbload(path[4],ads row7);
  rbstat(rbs);
  if rbs > 0 then
  begin
    clrscrn;
    cur_pos(24,0);bright;
    write('Error on load, status=',rbs);
    normal;
    goto exit;
  end;
edfunct := 0;
end;
exit:
end;
begin
  clscrn;
  cur_pos(24,0);bright;
  write('Error on load, status=',rbs);
  normal;
  goto exit;
end;
edfunct := 0;

end;

procedure load_ident4;

label exit;

begin
  row4.bnumber := bnum;
  if fmen = 'E' or else fmen = 'e' then
    rbput(path[2],ads row4)
  else begin
    rbfind(path[2],relname[4]);
    rbcheck('rbfind 9/2');
    rbload(path[2],ads row4);
    rbstat(rbs);
    if rbs <> 0 then
      begin
        clscrn;
        cur_pos(24,0);bright;
        write('Error on load, status=',rbs);
        normal;
        goto exit;
      end;
  edfunct := 0;
end;
exit:
end;

procedure load_follow1;

label exit;

begin
  row6.bnumber := bnum;
  if fmen = 'E' or else fmen = 'e' then
    rbput(path[2],ads row6)
  else begin
    rbfind(path[2],relname[6]);
    rbcheck('rbfind 1/2');
    rbload(path[2],ads row6);
    rbstat(rbs);
    if rbs <> 0 then
      begin
        clscrn;
        cur_pos(24,0);bright;
        write('Error on load, status=',rbs);
        normal;
        goto exit;
      end;
  edfunct := 0;
end;
exit:
end;

procedure load_follow2;

label exit;
begin
row12.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[2],ads row12)
else begin
row12.entdate := row6.entdate;
rbfind(path[2],relname[12]);
rbcheck('rbfind 1',2);
rbload(path[2],ads row12);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clrscr;
      cur_pos(24,0);bright;
      write('Error on load, status=',rbs);
      normal;
      goto exit;
    end;
edfunct := 0;
end;
exit:
end;
procedure load_follow3;
label exit;
begin
row13.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
  rbput(path[2],ads row13)
else begin
row13.entdate := row6.entdate;
rbfind(path[2],relname[13]);
rbcheck('rbfind 1',2);
rbload(path[2],ads row13);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      clrscr;
      cur_pos(24,0);bright;
      write('Error on load, status=',rbs);
      normal;
      goto exit;
    end;
edfunct := 0;
end;
exit:
end;
procedure enter_edit(formname:names; route:integer);
var 	 count : integer;
label 	 exit;
begin
  count := 1;
  rbform(path[1],formname,ads forbuf);
  rbstat(rbs);
  if rbs <> 0 then
    begin
      cur_pos(24,0);bright;
      writeln('Error on form initialize, status=',rbs);
      normal;
      goto exit;
    end;
if fmen = 'E' or else fmen = 'e' then begin
  case route of
    1 : begin rbget(path[5], ads row1); rbstat(rbs); end;
    2 : begin rbget(path[2], ads row2); rbstat(rbs); end;
    3 : begin rbget(path[2], ads row3); rbstat(rbs); end;
    4 : begin rbget(path[2], ads row9); rbstat(rbs); end;
    5 : begin rbget(path[2], ads row10); rbstat(rbs); end;
    6 : begin rbget(path[2], ads row8); rbstat(rbs); end;
    7 : begin rbget(path[3], ads row11); rbstat(rbs); end;
    8 : begin rbget(path[2], ads row7); rbstat(rbs); end;
    9 : begin rbget(path[2], ads row4); rbstat(rbs); end;
   10 : begin rbget(path[2], ads row6); rbstat(rbs); end;
   11 : begin rbget(path[2], ads row12); rbstat(rbs); end;
   12 : begin rbget(path[2], ads row13); rbstat(rbs); end;
  end;
end;

if rbs = -1 then goto exit;

while true do 
  begin
    z := 0;
    repeat
      case route of
        1 : rbdraw(path[1], ads forabuf, ads row1, edfunct);
        2 : rbdraw(path[1], ads forabuf, ads row2, edfunct);
        3 : rbdraw(path[1], ads forabuf, ads row3, edfunct);
        4 : rbdraw(path[1], ads forabuf, ads row9, edfunct);
        5 : rbdraw(path[1], ads forabuf, ads row10, edfunct);
        6 : rbdraw(path[1], ads forabuf, ads row8, edfunct);
        7 : rbdraw(path[1], ads forabuf, ads row11, edfunct);
        8 : rbdraw(path[1], ads forabuf, ads row7, edfunct);
        9 : rbdraw(path[1], ads forabuf, ads row4, edfunct);
       10 : rbdraw(path[1], ads forabuf, ads row6, edfunct);
       11 : rbdraw(path[1], ads forabuf, ads row12, edfunct);
       12 : rbdraw(path[1], ads forabuf, ads row13, edfunct);
      end;
    bright;
    cur_pos(24, 0);
    write('Press [ESC] when done');
    normal;
    count := 0;
    show_bnum;
    if mas <> 0 then show_mas;
    if treat <> 0 then show_treat;
    case route of
      1 : rbedit(path[1], ads forabuf, ads row1, count, ch);
      2 : rbedit(path[1], ads forabuf, ads row2, count, ch);
      3 : rbedit(path[1], ads forabuf, ads row3, count, ch);
      4 : rbedit(path[1], ads forabuf, ads row9, count, ch);
      5 : rbedit(path[1], ads forabuf, ads row10, count, ch);
      6 : rbedit(path[1], ads forabuf, ads row8, count, ch);
      7 : rbedit(path[1], ads forabuf, ads row11, count, ch);
      8 : rbedit(path[1], ads forabuf, ads row7, count, ch);
      9 : rbedit(path[1], ads forabuf, ads row4, count, ch);
     10 : rbedit(path[1], ads forabuf, ads row6, count, ch);
     11 : rbedit(path[1], ads forabuf, ads row12, count, ch);
     12 : rbedit(path[1], ads forabuf, ads row13, count, ch);
  end;
rbstat(rbs);
dataerr := rbs <> 0;
if dataerr then begin
  if rbs = 7 then 

begin
  cur_pos(24,0);
  bright;
  write('Data error, please correct ... Press [ESC] when done');
  edfunct := 1;
  cur_pos(1,0);
end else
begin
  clscrn;
  cur_pos(24,0); bright;
  writeln('Error on data entry, status='irbs);
  normal;
  goto exit;
end:
end else z := 1;
until z = 1;
while true do
begin
  cur_pos(24,0); bright;
  write('N(ext), E(dit again), Q(uit)?

  normal;
  cur_pos(24,29);
  spchin(ch);

  case chr(ch) of
    'q','Q':
      begin
        clscrn;
        rbclos;
        goto exit;
      end;
    'y','Y':
      begin
        mas := 0;
        treat := 0;
        goto exit;
      end;
    'n','N':
      begin
        if not dataerr then
          begin
            case route of
              1 : load_ident1;
              2 : load_ident2;
              3 : load_ident3;
              4 : load_matl;
              5 : load_matt;
              6 : load_study1;
              7 : load_study2;
              8 : load_treat1;
              9 : load_ident4;
              10 : load_follow1;
              11 : load_follow2;
              12 : load_follow3;
              otherwise
              cur_pos(24,0);
              write('

          end;
          goto exit;
          end;
else
begin
  bright; blink;
  Ml.

  normal;
  cur_pos(1,0);
end
write('Cannot load illegal data');
cur_pos(24,25);
sphin(ch);
end;
end;

'g', 'G':
begin
  edfunct := 1;
  break;
end;
otherwise
  beep;
end; [ case ]
while [ while ]
exit;
end; [ form ]


case

procedure add_census;

var route: integer;
screen: integer;
label exit;

begin
  mas := 0;
treat := 0;
edfunct := 0;
attlist[1] := 'BNUMBER '
wherop[1] := 'EQ '
wherao[1] := 'AND '
natt := 1;

b_prompt;
enter_edit('IDENT1 ',1);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 0;
enter_edit('IDENT2 ',2);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 0;
enter_edit('IDENT3 ',3);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 0;
enter_edit('MAT1 ',4);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 0;
enter_edit('MAT2 ',5);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
it := 0;
repeat
  loop_name := 'Study Area';
it2 := 0;
treat := 0;
mas := mas + 1;
edfunct := 0;
enter_edit('STUDY1 ',6);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 0;
enter_edit('STUDY2 ',7);

end; [ form ]
end; [ while ]
repeat
    treat := treat + 1;
    edfunct := 0;
    enter_edit('TREAT1 ',8);
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
    bright;
    cur_pos(24,0);
    write('Input Another Treatment (Y/N)? ');
    normal;
    cur_pos(24,29);
    spchin(ch);
    if chr(ch) = 'n' or else chr(ch) = 'N' then it2 := 1;
    until it2 = 1;
    bright;
    cur_pos(24,0);
    write('Input Another Study Area (Y/N)? ');
    normal;
    cur_pos(24,35);
    spchin(ch);
    if chr(ch) = 'n' or else chr(ch) = 'N' then it := 1;
    until it = 1;
    edfunct := 0;
    enter_edit('IDENT4 ',9);
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
    exit:
    [load_building;]
end;

procedure edit_ident;
var route : integer;
    screen: integer;

    exit:
begin
begin
    mas := 0;
    treat := 0;
    edfunct := 1;
    attlist[1] := 'BNUMBER ';
    wherp[1] := 'EQ ';
    wherea[1] := 'AND ';
    natt := 1;
    rbfind(path[5], relname[1]);
    rbwhere(path[5], attlist[1], wherop[1], wherao[1], natt);
    enter_edit('IDENT1 ',1);
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
    edfunct := 1;
    rbfind(path[2], relname[2]);
    rbwhere(path[2], attlist[1], wherop[1], wherea[1], natt);
    enter_edit('IDENT2 ',2);
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
    edfunct := 1;
    rbfind(path[2], relname[3]);
    rbwhere(path[2], attlist[1], wherop[1], wherea[1], natt);
    enter_edit('IDENT3 ',3);
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
exit:
end;
procedure edit_mat;
screen: integer;
label exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
atlist[1] := 'BNUMBER ';
whereop[1] := 'EQ ';
wherao[1] := 'AND ';
natt := 1;

rbfind(path[2],rename[9]);
rbwhere(path[2],attlist[1],whereop[1], ads where,wherao[1],natt);
enter_edit('MAT1 ',4);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 1;
rbfind(path[2],rename[10]);
rbwhere(path[2],attlist[1],whereop[1], ads where,wherao[1],natt);
enter_edit('MAT2 ',7);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
exit:
end;

procedure edit_study;
var route: integer;
screen: integer;
label exit:

begin
mas := 0;
treat := 0;
edfunct := 1;
atlist[1] := 'BNUMBER ';
whereop[1] := 'EQ ';
wherao[1] := 'AND ';
natt := 1;
rs1 := 0;

rbfind(path[2],rename[8]);
rbwhere(path[2],attlist[1],whereop[1], ads where,wherao[1],natt);
rbsl := 0;

rbfind(path[3],rename[11]);
rbwhere(path[3],attlist[1],whereop[1], ads where,whereo[1],natt);
repeat
mas := mas + 1;
enter_edit('STUDY1 ',6);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 1;
enter_edit('STUDY2 ',7);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rs1 = -1;
exit:
end;

procedure edit_treat;
var route: integer;
screen: integer;
begin
mas := 1;
treat := 0;
edfunc := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ ';
wherao[1] := 'AND ';
natt := 1;
rbs1 := 0;
edfunc := 1;
rbfind(path[4], relname[7]);
rbmher(path[4], attlist[1], wherop[1], ads where, where, where[1], natt);
repeat
treat := treat + 1;
enter_edit('TREAT1 ', 8);
astp := row7.mastype;
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbs1 = -1;
exit:
end;

procedure edit_entry;
var route : integer;
        screen : integer;

begin
mas := 0;
treat := 0;
edfunc := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ ';
wherao[1] := 'AND ';
natt := 1;
rbs1 := 0;
edfunc := 1;
rbfind(path[4], relname[4]);
rbmher(path[4], attlist[1], wherop[1], ads where, where, where[1], natt);
repeat
enter_edit('IDENT4 ', 9);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbs1 = -1;
exit:
end;

procedure edit_follow;
var route : integer;
        screen : integer;

begin
mas := 0;
treat := 0;
exit:
entdate_prompt;
edfunt := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ ';
whereo[1] := 'AND ';
natt := 2;

if rbsl = -1 then goto exit;
rbfind(path[2],rename[6]);
rbwher(path[2],attlist[1],wherop[1], ads where,whereo[1],natt);
edfunt := 1;
enter_edit('FOLLOW1 ',10);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunt := 1;
rbfind(path[2],rename[12]);
rbwher(path[2],attlist[1],wherop[1], ads where,whereo[1],natt);
repeat
enter_edit('FOLLOW2 ',11);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbsl = -1;
edfunt := 1;
rbfind(path[2],rename[13]);
rbwher(path[2],attlist[1],wherop[1], ads where,whereo[1],natt);
enter_edit('FOLLOW3 ',12);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
exit:
end;

procedure b_ask(public);

var b : packed array [1..4] of char;
bnun_ent : packed array [1..8] of char;

begin
attlist[1] := 'BNUMBER ';
attname[1] := 'BNUMBER ';
wherop[1] := 'CON ';
whereo[1] := 'AND ';
natt := 1;
natt_sort := 1;
dirlist[1] := 0;
repeat
cur_pos(22,14);
z := 0;
writeln('Enter the Building Number to Edit (i.e. GAC0005): ');
cur_pos(22,64);readln(bnum_ent);
u := 0;
z := 0;
where.int[1] := 8;
for w := 1 to 8 do begin where.str[w+2] := bnum_ent[w];write(4',where.str[w+2],4');end;
rbfind(path[1],rename[1]);
rbcheck('rbfind ',1);
rbwher(path[1],attlist[1],wherop[1], ads where,whereo[1],natt);
rbstat(rbs);
if rbs = -1 then begin
bright:blink;
cur_pos(23,19);write('Building Not Found ... Please Reenter');
normal;
end else begin
end;
end;
z := 1;
end;
until z = 1;
rbget(path[1],ads rowl);
bnum := rowl.bnumber;
if fmen = 'E' or else fmen = 'e' then begin
  case bnum_ent[3] of
    'C' : begin
      case fmen1 of
        'B', 'b' : edit_ident;
        'M', 'm' : edit_mat;
        'S', 's' : edit_study;
        'T', 't' : edit_treat;
        'E', 'e' : edit_entry;
        'F', 'f' : edit_follow;
      end;
    end;
    'S' : begin
      case fmen1 of
        'B', 'b' : edit_sident;
        'M', 'm' : edit_smat;
        'S', 's' : edit_sstudy;
        'T', 't' : edit_streat;
        'E', 'e' : edit_sentry;
        'F', 'f' : edit_follow;
      end;
    end;
  end;
end;
end;

procedure add_follow;
var route, screen, exit:
begin
  mas := 0;
  treat := 0;
  b_ask
  edfunct := 0;
  attlist[1] := 'BNUMBER';
  whereop[1] := 'EQ';
  wherao[1] := 'AND';
  natt := 1;
  enter_edit('FOLLOW1 ',10);
  if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

  loop_name := 'Study Area';
  it2 := 0;
  mas := 0;
  treat := 0;
repeat
  treat := treat + 1;
  edfunct := 0;
  enter_edit('FOLLOW2 ',11);
  if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
  bright;
  cur_pos(24,0);
  write('Input Another Treatment Follow-up (Y/N) ? ');
  normal;
  cur_pos(24,42);
if chr(ch) = 'n' or else chr(ch) = 'N' then it2 := 1;
until it2 = 1;
edfunct := 0;
end;

procedure delDisp(vars disp: integer);
begin
cur_pos(24,30); write('Removing ', disp:2, ' rows');
disp := 0;
end;

procedure delete_bldg;
var count : integer;
begin
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ ';
wherao[1] := 'AND ';
natt := 1;
b_ask;
count := 0;
bright;
cur_pos(23,1); write(' Removing Building from database please wait ');
normal;
rbfind(path[1], relname[1]);
rwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row1);
rstat(rbs);
if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
delDisp(count);
rbfind(path[1], relname[2]);
rwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row2);
rstat(rbs);
if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
delDisp(count);
rbfind(path[1], relname[3]);
rwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row3);
rstat(rbs);
if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
delDisp(count);
rbfind(path[1], relname[4]);
rwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row4);
rstat(rbs);
if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
delDisp(count);
rbfind(path[1], relname[5]);
rwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row5);
rstat(rbs);
if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
delDisp(count);
rbfind(path[1], relname[6]);
rwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row6);
rstat(rbs);
if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
delDisp(count);
rbfind(path[1], relname[7]));
rwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row7);
repeat
  rbget(path[1], ads row6);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
  until rbs = -1;
  del_disp(count);
  rbfind(path[1], relname[7]);
  rbwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
  rbget(path[1], ads row7);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
  until rbs = -1;
  del_disp(count);
  rbfind(path[1], relname[8]);
  rbwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
  rbget(path[1], ads row8);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
  until rbs = -1;
  del_disp(count);
  rbfind(path[1], relname[9]);
  rbwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
  rbget(path[1], ads row9);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
  until rbs = -1;
  del_disp(count);
  rbfind(path[1], relname[10]);
  rbwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
  rbget(path[1], ads row10);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
  until rbs = -1;
  del_disp(count);
  rbfind(path[1], relname[11]);
  rbwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
  rbget(path[1], ads row11);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
  until rbs = -1;
  del_disp(count);
  rbfind(path[1], relname[12]);
  rbwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
  rbget(path[1], ads row12);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
  if rbs <> -1 then rbdel(path[1]);
  until rbs = -1;
  del_disp(count);
  rbfind(path[1], relname[13]);
  rbwher(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
  rbget(path[1], ads row13);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
  until rbs = -1;
  del_disp(count);
  rbclose;
rbclose;
rbcopen(dbname);
procedure convert;

var  bnum_temp : packed array [1..8] of char;

begin

attlist[1] := 'BNUMBER ';  
wherop[1] := 'EQ ';    
wherea[1] := 'AND ';    
natt := 1;

b_ask;
brightness;
cur_pos(23,1);write(' Converting Short form to Long Census format ');

rbfind(path[1],relname[1]);
rbcheck('find ',1);
rbwher(path[1],attlist[1],wherop[1], ads where,wherea[1],natt);
rbcheck('where ',1);
repeat
  rbget(path[1],ads row1);  
  rbstat(rbs);
  if rbs <> -1 then begin
    bnum_temp := row1.bnumber;
    row1.bnumber := bnum_temp;
    rbput(path[1],ads row1);
  end;
until rbs = -1;
rbfind(path[1],relname[2]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherea[1],natt);
repeat
  rbget(path[1],ads row2);  
  rbstat(rbs);
  if rbs <> -1 then begin
    bnum_temp := row2.bnumber;
    row2.bnumber := bnum_temp;
    rbput(path[1],ads row2);
  end;
until rbs = -1;
rbfind(path[1],relname[3]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherea[1],natt);
repeat
  rbget(path[1],ads row3);  
  rbstat(rbs);
  if rbs <> -1 then begin
    bnum_temp := row3.bnumber;
    row3.bnumber := bnum_temp;
    rbput(path[1],ads row3);
  end;
until rbs = -1;
rbfind(path[1],relname[4]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherea[1],natt);
repeat
  rbget(path[1],ads row4);  
  rbstat(rbs);
  if rbs <> -1 then begin
    bnum_temp := row4.bnumber;
    row4.bnumber := bnum_temp;
    rbput(path[1],ads row4);
  end;
until rbs = -1;
until rbs = -1;
rbfind(path[1], relname[7]);
rbwhere(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row7);
rstat(rbs);
if rbs <> -1 then begin
  bnum_temp := row7.bnumber;
  row7.bnumber := bnum_temp;
rput(path[1], ads row7);
end;
until rbs = -1;
rbfind(path[1], relname[6]);
rbwhere(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row6);
rstat(rbs);
if rbs <> -1 then begin
  bnum_temp := row6.bnumber;
  row6.bnumber := bnum_temp;
rput(path[1], ads row6);
end;
until rbs = -1;
rbfind(path[1], relname[8]);
rbwhere(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row8);
rstat(rbs);
if rbs <> -1 then begin
  bnum_temp := row8.bnumber;
  row8.bnumber := bnum_temp;
rput(path[1], ads row8);
end;
until rbs = -1;
rbfind(path[1], relname[9]);
rbwhere(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row9);
rstat(rbs);
if rbs <> -1 then begin
  bnum_temp := row9.bnumber;
  row9.bnumber := bnum_temp;
rput(path[1], ads row9);
end;
until rbs = -1;
rbfind(path[1], relname[10]);
rbwhere(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row10);
rstat(rbs);
if rbs <> -1 then begin
  bnum_temp := row10.bnumber;
  row10.bnumber := bnum_temp;
rput(path[1], ads row10);
end;
until rbs = -1;
rbfind(path[1], relname[11]);
rbwhere(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
rbget(path[1], ads row11);


if rbs <> -1 then begin
    bnum_temp := row11.bnumber;
    row11.bnumber := bnum_temp;
    rbput(path[1], ads row11);
end;
until rbs = -1;
rbfind(path[1], relname[12]);
rwhere(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
    rbget(path[1], ads row12);
    rbstat(rbs);
    if rbs <> -1 then begin
        bnum_temp := row12.bnumber;
        row12.bnumber := bnum_temp;
        rbput(path[1], ads row12);
    end;
until rbs = -1;
rbfind(path[1], relname[13]);
rwhere(path[1], attlist[1], wherop[1], ads where, wherao[1], natt);
repeat
    rbget(path[1], ads row13);
    rbstat(rbs);
    if rbs <> -1 then begin
        bnum_temp := row13.bnumber;
        row13.bnumber := bnum_temp;
        rbput(path[1], ads row13);
    end;
until rbs = -1;
rbclos;
rbopen(dbsname);
end;
interface;
unit census(pointer, names, name3, name9, dollar, shortnames, char3, char7, char8, char40, char6, shorts, string8, name15, integer5, boolean6, string180, string50, whertype, rel1, rel2, rel3, rel4, rel5, rel6, rel7, rel8, rel9, rel10, rel11, rel12, rel13, compress, form_feed, no_com, esc, rbs, rbs1, rbs2, owner, dbname, formname, status, forbuf, line, col, dataerr, blank, edfunc, in2, in1, relname, attnm, wherop, whereattlist, dirlist, attname, path, orrow1, orrow2, orrow3, orrow4, orrow5, orrow6, orrow7, row8, orrow8, row9, row10, row11, row12, row13, jks, lmn, xzy, yuv, wlm, pchxx, chnat, sort, buildnum, i, num, mas, treat, it, it2, loop_name, fmen, fmen1, fmen2, fmen3, fmen4, fmenu, choice1, choice2, choice1, choice2, choice2, choice1, page_no, margin, today, date, title, time, now, cl, fuuid, new, rbrstrt, rbropen, rbfind, rbwhere, rbsort, rbsort2, rbget, rbput, rbdel, rbcload, rbcload2, rbcuser, rbcset, rbcset2, intmov, lnbc, intcode, date, strmov, time, scriptr, chin, conrdy, rbform, rbdraw, rbedit, ilscr, moveto, conout, blin, beep, bright, clear_screen, blink, under_line, normal, cancel, reverse, scr_loc, scr_pos, scroll_up, scroll_down, rbccheck, search, blister, bnum);

TYPE
pointer = ADS of word;
names = packed array[1..8] of char;
names3 = packed array[1..3] of char;
names9 = packed array[1..9] of char;
dollar = array[1..4] of integer;
shortnames = packed array[1..4] of char;
char3 = packed array[1..3] of char;
char7 = packed array[1..7] of char;
char8 = packed array[1..8] of char;
char40 = packed array[1..40] of char;
char6 = packed array[1..6] of char;
shorts = array[1..4] of shortnames;
string8 = string(8);
names15 = array[1..15] of names;
integer5 = array[1..5] of integer;
bool6 = array[1..6] of boolean;
string180 = packed array[1..180] of char;
string50 = packed array[1..50] of char;

whertype = record case integer of
1: (STR: string(80));
2: (I4: array[1..20] of integer4);
3: (INT: array[1..40] of integer);
end: [end where type record]

rel1 = record
BNUMBER := string(8);
BNAME := string(40);
ALNAME := string(40);
BSTREET := string(30);
BCITY := string(20);
BCOUNTY := string(20);
BSTATE := string(4);
BZIP := string(10);
BUTM := string(18);
NPSREG := string(4);
FEDOWN := string(4);
NHL := string(4);
TRA := string(4);
HABSNO := string(20);
end: [ident1]

rel2 = record
BNUMBER := string(8);
OSHIP := string(8);
end: [ident2]
OSTREET : string(30);
OCITY : string(20);
OSTATE : string(4);
OZIP : string(10);
OPHONE : string(14);
MANAGER : string(40);
MSTREET : string(30);
MCITY : string(20);
MSTATE : string(4);
MZIP : string(10);
MPHONE : string(14);
end; (ident2)

rel3 = record
BNUMBER : string(8);
OCCSTAT : string(4);
CURUSE : string(70);
ACCESS : string(4);
CNST : string(20);
FEDFUND : string(30);
NRSTAT : string(4);
SF : integer4;
STORIES : string(16);
end; (ident3)

rel4 = record
BNUMBER : string(8);
RNAME : string(40);
RTITLE : string(30);
RORGAN : string(30);
RSTREET : string(30);
RCITY : string(20);
RSTATE : string(4);
RZIP : string(10);
RPHONE : string(14);
ENTDATE : string(8);
end; (ident4)

rel6 = record
BNUMBER : string(8);
ENTDATE : string(8);
BNAME : string(40);
ONAME : string(40);
OSTREET : string(30);
OCITY : string(20);
OSTATE : string(4);
OZIP : string(10);
OPHONE : string(14);
END; (follow)

rel12 = record
BNUMBER : string(8);
ENTDATE : string(8);
TRDATE : string(8);
TRLOC : string(30);
TRTYPE : string(90);
MASTYPE : string(24);
PROD1 : string(60);
PROD2 : string(60);
MATCHUNG : string(4);
CHNGDES : string(180);
AREADES : string(180);
LAPSETM : string(10);
REPTRT : string(4);
REPDATE : string(8);
FUTTRT : string(4);
FUTDES : string(180);
COMMENTS : string(180);
end; (follow2)
BNUMBER : string(8);
ENTDATE : string(8);
RNAME : string(40);
RORGAN : string(30);
RTITLE : string(30);
RSTREET : string(30);
RCITY : string(20);
RSTATE : string(4);
RZIP : string(10);
RPHONE : string(14);
end; [follow3]
rel 7 = record
BNUMBER : string(8);
MASTYPE : string(24);
NUMBER : INTEGER4;
TRTTYPE : string(90);
RUNUM : INTEGER4;
BEGDATE : string(8);
ENDDATE : string(8);
TESTED : string(4);
TESTDES : string(180);
PROD1 : string(60);
PROD2 : string(60);
PROD3 : string(60);
TEMP : string(10);
HUMIDITY : string(10);
PRECIP : string(10);
AREATRT : string(30);
APPEAR : string(180);
end; [treatant]
rel 8 = record
BNUMBER : string(8);
STUDYLOC : string(30);
MASTYPE : string(24);
NUMBER : INTEGER4;
MASORRY : string(24);
MASCOUR : string(24);
MASF1N : string(24);
MASURE : string(24);
MASCND : string(24);
MASCOR : string(24);
MASDENS : string(10);
MASPOROS : string(10);
MASTREN : string(10);
ADJMAT1 : string(30);
ADJMAT2 : string(30);
ADJMAT3 : string(30);
MORTTYPE : string(10);
MORTCOLOR : string(10);
MORTSOFT : string(4);
JOINTYPE : string(10);
JOINTDEP : string(16);
MORTCOND : string(10);
MORTANAL : string(4);
end;[study1]
rel 11 = record
BNUMBER : string(8);
MASTYPE : string(24);
NUMBER : INTEGER4;
UNTRETCD : string(60);
COATCOND : string(60);
MOISTPRB : string(60);
TRT1 : string(30);
TRTDATE1 : string(8);
TRT2 : string(30);

end; (study2)
rel9 = record
  BNUMBER : string(8);
  CONSTDT : string(8);
  ARCH : string(34);
  BUILDER : string(34);
  ALTDATE1 : string(8);
  ALTARCH1 : string(34);
  ALTBLD1 : string(34);
  ALTDATE2 : string(8);
  ALTARCH2 : string(34);
  ALTBLD2 : string(34);
  ALTDATE3 : string(8);
  ALTARCH3 : string(34);
  ALTBLD3 : string(34);
  STRUCSYS : string(90);
  ROOFTYPE : string(30);
  ROOFMAT : string(30);
  DRAINTYP : string(30);
  DRAINCON : string(10);
end; (oati)

end; [mat1]
rel10 = record
  BNUMBER : string(8);
  MASTYPE1 : string(24);
  MASQRRY1 : string(24);
  MASCOUR1 : string(24);
  MASFIN1 : string(24);
  MASUSE1 : string(24);
  MASLOC1 : string(24);
  MASCOND1 : string(24);
  MASCOLR1 : string(24);
  MASTYPE2 : string(24);
  MASQRRY2 : string(24);
  MASCOUR2 : string(24);
  MASFIN2 : string(24);
  MASUSE2 : string(24);
  MASLOC2 : string(24);
  MASCOND2 : string(24);
  MASCOLR2 : string(24);
  MASTYPE3 : string(24);
  MASQRRY3 : string(24);
  MASCOUR3 : string(24);
  MASFIN3 : string(24);
  MASUSE3 : string(24);
  MASLOC3 : string(24);
  MASCOND3 : string(24);
  MASCOLR3 : string(24);
end; (matt)

CONST
  compress = chr(15);
  fora feed = chr(12);
  noco = chr(18);
  esc = chr(27);

VAR
  rbs: INTEGER;  (RBASE status indicator)
function vdoxqq(areg:word):word;

function null(varname: pointer): boolean;

procedure rbstrti;

procedure rbstat (vars rbs: integer);

procedure rbopen (vars dbname: name9);

procedure rbfind (vars path: integer; vars renname: names);

procedure rbwhe (vars path: integer; vars att, ops: names;
values: pointer; vars andor: shortnames;
vars nboo: integer);

procedure rbsort (vars path: integer; vars att: names;
vars natt, direction: integer);

procedure rbget (vars path: integer; tuple: pointer);

procedure rbput (vars path: integer; tuple: pointer);

procedure rbdel (vars path: integer);

procedure rbload (vars path: integer; tuple: pointer);

procedure rbclose;

procedure rbuser (vars pwnname: names);

procedure rbset (vars set1: names; vars set2: char);

procedure rbwhr2 (vars path: integer; vars att, ops: names;
values: pointer; vars andor: shortnames;
vars nboo: integer);

procedure intmov (pstr: pointer; vars start, length: integer;
vars rimint: integer4);

procedure Inbc(pstr: painter; vars start, nchar, last: integer);

procedure intcod (pstr: pointer; vars count: integer;
vars rimint: integer4; vars fillchar: char);

procedure date(var d: string);

procedure str (string1: names; pos1: integer in char: integer;
vars string2: Istring; pos2: integer);

procedure time(var t: string);

procedure script;

procedure spchin(vars ch: integer);

procedure conrdy(vars chr, state: integer);

procedure rbform (vars path: integer; vars form: names; buff: pointer);

procedure rbdraw (vars path: integer; buff, raw: pointer; vars funct: integer);
procedure clscrn;
procedure moveto (vars line, col: integer);
procedure conout (str: pointer; vars count: integer);
procedure bline (vars linen: integer);
procedure beep;
procedure bright;
procedure clear_screen(ch:cl:dl:word);
procedure blink;
procedure under_line;
procedure normal;
procedure cancel;
procedure reverse;
procedure scr_loc(rowicol:integer;chiatt:char);
procedure cur_pos(rowicol:integer);
procedure scroll_up(urucilr:1c:integer);
procedure scroll_dn(urucilr:1c:integer);
procedure rbcheck(operation:names;p:integer);
procedure search;
procedure blist;

begin
end;
procedure util_menu;
begin
  j := 0;
  repeat
    clear_screen(0,0,24,79);
    cur_pos(2,18):writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
    cur_pos(4,15):writeln('Programmed for the Preservation Assistance Division');
    cur_pos(5,27):writeln('U.S. National Park Service');
    bright;
    cur_pos(7,28):writeln('UTILITY MENU');
    cur_pos(9,28):writeln('PUTILITY MENU');
    normal;
    cur_pos(11,28):bright:write('C'):normal:write('convert Short to Long Form');
    cur_pos(12,28):bright:write('D'):normal:write('delete a Short or Long Form');
    cur_pos(13,28):bright:write('R'):normal:write('return to Main Menu');
    cur_pos(14,28):bright:write('E'):normal:write('');
    cur_pos(15,28):bright:write('T'):normal:write('');
    cur_pos(16,28):bright:write('I'):normal:write('');
    cur_pos(17,28):bright:write('A'):normal:write('
');
    cur_pos(22,29):write('Enter the option desired: 1');
    repeat cur_pos(22,55):write('); cur_pos(22,56):readln(fmen1);
    case fmen1 of
      'C','c': begin i := 1; convert := 1; end;
      'D','d': begin i := 1; delete_bldg := 1; end;
      'R','r': begin i := 1; j := 1; end;
      'I','I': begin i := 1; j := 0; end;
      'A','A': begin i := 1; j := 1; end;
      otherwise
        begin
          blink; cur_pos(23,26):write('INVALID ENTRY -- Please Re-enter');
          normal;
          i := 0;
          cycle;
        end;
    end;
    until i = 1;
    until j = 1;
  end;
procedure edit_menu;
begin
  j := 0;
  repeat
    clear_screen(0,0,24,79);
    cur_pos(2,18):writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
    cur_pos(4,15):writeln('Programmed for the Preservation Assistance Division');
    cur_pos(5,27):writeln('U.S. National Park Service');
    bright;
    cur_pos(7,30):writeln('EDIT MENU');
    cur_pos(9,30):writeln('');
    normal;
    cur_pos(11,28):bright:write('B'):normal:write('building data (Part I)');
    cur_pos(12,28):bright:write('M'):normal:write('material data (cont I)');
    cur_pos(13,28):bright:write('S'):normal:write('study area data (Part II)');
    cur_pos(14,28):bright:write('T'):normal:write('treatment data (Part III)');
    cur_pos(15,28):bright:write('E'):normal:write('entry data');
    cur_pos(16,28):bright:write('F'):normal:write('follow-up data');
    cur_pos(17,28):bright:write('P'):normal:write('return to Main Menu');
    cur_pos(22,29):write('Enter the option desired: 1');
    repeat cur_pos(22,55):write('); cur_pos(22,56):readln(fmen1);
    case fmen1 of
      'B','b': begin i := 1; end;
      'M','m': begin i := 1; end;
      'S','s': begin i := 1; end;
      'T','t': begin i := 1; end;
      'E','e': begin i := 1; end;
      'F','f': begin i := 1; end;
      'P','p': begin i := 1; end;
      otherwise
        begin
          blink; cur_pos(23,26):write('INVALID ENTRY -- Please Re-enter');
          normal;
          i := 0;
          cycle;
        end;
    end;
    until i = 1;
  end;
begin i := 1; b_ask := 0; end;
'S'/'s': begin i := 1; b_ask := 0; end;
'F'/'f': begin i := 1; b_ask := 0; end;
'R'/'r': begin i := 1; j := 1; end;
otherwise
begin
  blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
  normal;
  i := 0;
  cycle;
end;
end;
end;

procedure add_menu;
begin
  j := 0;
  repeat
    clear_screen(0,0,24,79);
    cur_pos(2,18); writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
    cur_pos(4,15); writeln('Programmed for the Preservation Assistance Division');
    cur_pos(5,27); writeln('U.S. National Park Service');
    bright;
    cur_pos(7,31); writeln('A D D M E N U');
    normal;
    cur_pos(11,28); writeln('Census form');
    cur_pos(12,28); writeln('Follow-up form');
    cur_pos(13,28); writeln('Short form');
    cur_pos(22,29); write('Enter the option desired: ');
    repeat cur_pos(22,55); write('');
      cur_pos(22,56); readln(fmen1);
    case fmen1 of
      'C'/'c': begin i := 1; add_census; j := 1; end;
      'F'/'f': begin i := 1; fmen1 := ''; add_follow; j := 1; end;
      'S'/'s': begin i := 1; add_short; j := 1; end;
      'R'/'r': begin i := 1; j := 1; end;
    otherwise
      begin
        blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
        normal;
        i := 0;
        cycle;
      end;
    end;
  until i = 1;
  until j = 1;
end;

procedure report_menu;
begin
  j := 0;
  repeat
    clear_screen(0,0,24,79);
    cur_pos(2,18); writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
    cur_pos(4,15); writeln('Programmed for the Preservation Assistance Division');
    cur_pos(5,27); writeln('U.S. National Park Service');
    bright;
    cur_pos(7,31); writeln('R E P O R T M E N U');
    normal;
    cur_pos(11,28); writeln('Enter the option desired: ');
    repeat cur_pos(11,55); write('');
      cur_pos(11,56); readln(fmen1);
    if fmen1 = '' then
      begin
        blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
        normal;
        i := 0;
        cycle;
      end;
    end;
end;
cur_pos(12,28); bright; write('F'); normal; write('Follow-up Report');
cur_pos(13,28); bright; write('B'); normal; write('Building List');
cur_pos(14,28); bright; write('R'); normal; write('Return to Main Menu');
cur_pos(22,29); write('Enter the option desired: '); repeat cur_pos(22,55); write(''); cur_pos(22,56); readln(fmen);
case fmen of
  'C': begin i := 1; comp; j := 0; end;
  'F': begin i := 1; print_follow; j := 0; end;
  'B': begin i := 1; blist; j := 0; end;
  'R': begin i := 1; j := 1; end;
otherwise
  begin
    blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
    normal;
    i := 0;
    cycle;
  end;
end;
until i = 1;
until j = 1;
end;
procedure query_menu;
begi
  j := 0;
  repeat
    clear_screen(0,0,24,79);
    cur_pos(2,18); writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
    cur_pos(4,15); writeln('Programmed for the Preservation Assistance Division');
    cur_pos(5,27); writeln('U.S. National Park Service');
    bright;
    cur_pos(7,29); writeln('');
    cur_pos(8,29); writeln('QUERY');
    normal;
    search;
    j := 1;
    until j = 1;
end;
procedure exit;
begi
  i := 1; j := 1; k := 1
end;
procedure main_menu;
begi
  j := 0;
  repeat
    clear_screen(0,0,24,79);
    cur_pos(2,18); writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
    cur_pos(4,15); writeln('Programmed for the Preservation Assistance Division');
    cur_pos(5,27); writeln('U.S. National Park Service');
    bright;
    cur_pos(7,29); writeln('');
    cur_pos(8,29); writeln('MAIN MENU');
    normal;
    cur_pos(11,23); bright; write('A'); normal; write('Add information to CENSUS database');
    cur_pos(12,23); bright; write('D'); normal; write('dit information in CENSUS database');
    cur_pos(13,23); bright; write('T'); normal; write('rint standard CENSUS reports');
    cur_pos(14,23); bright; write('Q'); normal; write('uery the CENSUS database');
    cur_pos(15,23); bright; write('U'); normal; write('se CENSUS database Utilities');
    cur_pos(16,23); bright; write('X'); normal; write('xit to DOS');
    cur_pos(22,29); write('Enter the option desired: '); repeat cur_pos(22,55); write(''); cur_pos(22,56); readln(fmen);
case fmen of
  'A': begin comp; j := 0; end;
  'D': begin blist; j := 0; end;
  'T': begin print_follow; j := 0; end;
  'Q': begin search; j := 1; end;
  'U': begin exit; j := 1; end;
  'X': begin exit; j := 1; end;
'E', 'e': begin open_db; edit_menu; close_db; i := 1; end;
'P', 'p': begin open_db; report_menu; close_db; i := 1; end;
'O', 'q': begin open_db; query_menu; close_db; i := 1; end;
'U', 'u': begin open_db; util_menu; close_db; i := 1; end;
'X', 'x': begin open_db; exit; close_db; i := 1; k := 1; end;
otherwise
    begin
        blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
        normal;
        i := 0;
        cycle;
        end;
end;
until i = 1;
until k = 1;
end;
unit crep2(b_ask: print_follow; comp);
procedure b Ask;
procedure print_follow;
procedure comp;

begin
end;

{$include: 'c.pas'}

implementation of crept;

uses census;

procedure b_ask; extern;

procedure bstring(instring: string100; vars outstr1, outstr2, outstr3, outstr4: string50);

var
  tempstring : string50;
  strlen : integer;
  i, index : integer;
  stpt : integer;
  outstr : array [1..4] of string50;

begin
  if null(adstring) then begin
    outstr1 := ')
    outstr2 := ')
    outstr3 := ')
    outstr4 := ')
  end else begin
    for index := 1 to 4 do
      for i := 1 to 50 do outstr[index][i] := ' ';
    for i := 1 to 50 do tempstring[i] := instring[i];
    stpt := 0;
    for index := 1 to 4 do begin
      strlen := scaneq(-10,' ',tempstring,50);
      if (strlen <> -10) or (strlen 0 50) then begin
        for i := 1 to (50+strlen) do outstr[index][i] := tempstring[i];
        if strlen 0 0 then for i := (50+strlen+1) to 50 do outstr[index][i] := ";
      end else
        write('error in string search ', index:2);
    stpt := stpt + 50 + strlen;
    for i := 1 to 50 do tempstring[i] := instring[stpt+i];
    end (for)
    outstr1 := outstr[1];
    outstr2 := outstr[2];
    outstr3 := outstr[3];
    outstr4 := outstr[4];
  end;
end;

procedure part_one_long;
var
  margin : integer;

begin
  margin := 5;
  rbfind(path[2], relname[2]);
  rbwher(path[2], attlist[1], wherop[1], ads where, wherao[1], natt);
end;
<table>
<thead>
<tr>
<th>PART I</th>
<th>BACKGROUND INFORMATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME/LOCATION</td>
<td></td>
</tr>
<tr>
<td>Historical Name of Structure</td>
<td>1. rowl.bname:40, rowl.bnumber:7</td>
</tr>
<tr>
<td>Contemporary Name of Structure (if different)</td>
<td>2. rowl.alname:40</td>
</tr>
<tr>
<td>State</td>
<td>rowl.bstate:2, rowl.npsreg:2</td>
</tr>
<tr>
<td>NPS Region</td>
<td></td>
</tr>
<tr>
<td>OWNERSHIP/MANAGEMENT</td>
<td></td>
</tr>
<tr>
<td>Ownership</td>
<td>3. row2.oship:8</td>
</tr>
<tr>
<td>Name of Primary Owner(s)</td>
<td>4. row2.oname:40, row2.ophone:14</td>
</tr>
<tr>
<td>Street Address</td>
<td>5. row2.astreet:30, row2.acity:20, row2.ostate:2, row2.azip:10</td>
</tr>
<tr>
<td>City</td>
<td>row2.acity:20, row2.ostate:2, row2.azip:10</td>
</tr>
<tr>
<td>UTM Coordinates</td>
<td>row2.utsr:30, row2.acity:20, row2.ostate:2, row2.azip:10</td>
</tr>
<tr>
<td>USE/OCCUPANCY/STATUS</td>
<td></td>
</tr>
<tr>
<td>Building Manager/Superintendent/Project Architect</td>
<td>row2.manager:40, row2.mphone:14</td>
</tr>
<tr>
<td>Street Address</td>
<td>row2.mstreet:30, row2.mcity:20, row2.mstate:2, row2.mzip:10</td>
</tr>
<tr>
<td>City</td>
<td>row2.mcity:20, row2.mstate:2, row2.mzip:10</td>
</tr>
</tbody>
</table>
if null(ads row3.access) then row3.access := 1;
if null(ads row3.cnst) then row3.cnst := 1;
if null(ads row3.fedfund) then row3.fedfund := 30;
if null(ads row3.nrstat) then row3.nrstat := 4;
writeln(cl, ':margin, '11. row3.occstat:4);
writeln(cl, ':margin, Percent Occupied');
writeln(cl, ':margin, row3.curuse:65);
writeln(cl, ':margin, Present Use');
writeln(cl, ':margin, row3.access:4);
writeln(cl, ':margin, Accessible');
writeln(cl, ':margin, row3.cnst:20);
writeln(cl, ':margin, Type of Construction Work in progress or pending');
writeln(cl, ':margin, row3.fedfund:30);
writeln(cl, ':margin, Explain how federal funds are involved in this project');
writeln(cl, ':margin, row3.nrstat:4);
writeln(cl, ':margin, Designation/ National Historic Register');
writeln(cl, ':margin);
write(chform_feed);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl, ':margin, BUILDING MATERIAL DATA');
writeln(cl, ':margin, BUILDING DESCRIPTION');
writeln(cl, ':margin, Dates of construction');
writeln(cl, ':margin, BUILDING MATERIAL DATA');
writeln(cl, ':margin, (cont) BUILDING MATERIAL DATA');
writeln(cl, ':margin, BUILDING DESCRIPTION');
writeln(cl, ':margin, Dates of construction');
if null(ads row9.constdt) then row9.constdt := 1;
if null(ads row9.arch) then row9.arch := 1;
if null(ads row9.builder) then row9.builder := 1;
if null(ads row9.altdate1) then row9.altdate1 := 1;
if null(ads row9.altarch1) then row9.altarch1 := 1;
if null(ads row9.altbldr1) then row9.altbldr1 := 1;
if null(ads row9.altdate2) then row9.altdate2 := 1;
if null(ads row9.altarch2) then row9.altarch2 := 1;
if null(ads row9.altbldr2) then row9.altbldr2 := 1;
if null(ads row9.altdate3) then row9.altdate3 := 1;
if null(ads row9.altarch3) then row9.altarch3 := 1;
if null(ads row9.altbldr3) then row9.altbldr3 := 1;
write(chform_feed);
writeln(cl, ':margin, Year | Architect | Builder/contractor |');
write(chform_feed);
write(chform_feed);
write(chform_feed);
write(chform_feed);
write(chform_feed);
writeln(cl, ':margin, BUILDING DESCRIPTION');
if null(ads row3.sf) then row3.sf := 0;
write(chform_feed);
writeln(cl, ':margin, Size - Height (approx. feet or stories) | Square feet (approx.)');
write(chform_feed);
writeln(cl, ':margin, Type of Wall Structural System (ex: load bearing, veneer, etc.)');
write(chform_feed);
writeln(cl, ':margin, Roof type and drainage system');
if null(ads row9.rooftype) then row9.rooftype := 0;
if null(ads row9.roofmat) then row9.roofmat := 0;
if null(ads row9.draincon) then row9.draincon := 0;
write(chform_feed);
writeln(cl, 'Roof material: ', row9.roofmat:30); writeln(cl, 'Drainage type: ', row9.draintype:30); writeln(cl, 'Drainage condition: ', row9.draincon:10); writeln(cl, 'Type(s) of Masonry Used in Building');
if null(row10.mastype1) then row10.mastype1 := 
if null(row10.masqrry1) then row10.masqrry1 := 
if null(row10.masqrry1) then row10.masqrry1 := 
if null(row10.mascour1) then row10.mascour1 := 
if null(row10.masfin1) then row10.masfin1 := 
if null(row10.masuse1) then row10.masuse1 := 
if null(row10.masloc1) then row10.masloc1 := 
if null(row10.mascond1) then row10.mascond1 := 
if null(row10.mastype2) then row10.mastype2 := 
if null(row10.masqrry2) then row10.masqrry2 := 
if null(row10.mascour2) then row10.mascour2 := 
if null(row10.masfin2) then row10.masfin2 := 
if null(row10.masuse2) then row10.masuse2 := 
if null(row10.masloc2) then row10.masloc2 := 
if null(row10.mascond2) then row10.mascond2 := 
if null(row10.mastype3) then row10.mastype3 := 
if null(row10.masqrry3) then row10.masqrry3 := 
if null(row10.mascour3) then row10.mascour3 := 
if null(row10.masfin3) then row10.masfin3 := 
if null(row10.masuse3) then row10.masuse3 := 
if null(row10.masloc3) then row10.masloc3 := 
if null(row10.mascond3) then row10.mascond3 :=
writeln(cl, 'Masonry type #1 | Masonry type #2 | Masonry type #3');
close(c1);
CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

SHORT REPORT

National Historic Landmark: row1.nhl
Federal Property: row1.fedown
Tax Act Project: row1.tra

PART I SHORT BACKGROUND INFORMATION

NAME/LOCATION
Historical Name of Structure: row1.altname
Contemporary Name of Structure (if different): row1.bname
State: row1.bstate
NPS Region: row1.npsreg
County: row1.bcounty
City: row1.bcity
Street Address (ex. 440 G Street NW): row1.bstreet
Zip Code: row1.bzip
N.R. number: row2.namer
UTM Coordinates: row2.utm

OWNERSHIP/MANAGEMENT
Name of Primary Owner(s): row2.oname
Phone #: row2.ophone

USE/OCCUPANCY/STATUS
Designation/ National Historic Register:

BUILDING DESCRIPTION
Dates of construction:
Year
Architect
Builder/contractor

close(c1);
end;
procedure part_two_long;
var margin : integer;
s1 : string50;
s2 : string50;
s3 : string50;
s4 : string50;
begin
  s1 := ';
s2 := ';
s3 := ';
s4 := ';
  rbget(path[3],ads row11);
  margin := 5;
  assign(cll'Iptl:');
  rewrite(cll); if null(ads row8.mastype) then row8.mastype := ';
  if null(ads row8.masqrry) then row8.masqrry := ';
  if null(ads row8.mascour) then row8.mascour := ';
  if null(ads row8.masfin) then row8.masfin := ';
  if null(ads row8.masuse) then row8.masuse := ';
  if null(ads row8.mascond) then row8.mascond := ';
  if null(ads row8.mascolor) then row8.mascolor := ';
  if null(ads row8.studyloc) then row8.studyloc := ';
  if null(ads row8.masdens) then row8.masdens := ';
  if null(ads row8.masporos) then row8.masporos := ';
  if null(ads row8.masstren) then row8.masstren := ';
  if null(ads row8.adjam1) then row8.adjam1 := ';
  if null(ads row8.adjam2) then row8.adjam2 := ';
  if null(ads row8.adjam3) then row8.adjam3 := ';
  if null(ads row8.morttype) then row8.morttype := ';
  if null(ads row8.mortcolor) then row8.mortcolor := ';
  if null(ads row8.mortsoft) then row8.mortsoft := ';
  if null(ads row8.jointype) then row8.jointype := ';
  if null(ads row8.jointdep) then row8.jointdep := ';
  writeln(cl,':margin,"PART II MASONRY STUDY AREA"');
  writeln(cl,':margin,"22. Location ",row8.studyloc:30," (ex: Center, NW side etc.)"');
  writeln(cl,':margin,"24./25. Geological Phy sical data"');
  writeln(cl,':margin,"26. Adjacent Non-masonry Materials"');
  writeln(cl,':margin,"Type ",row8.mastype:22," (1) ");
  writeln(cl,':margin,"Quarry ",row8.masqrry:22," (2) ");
  writeln(cl,':margin,"Coursing ",row8.mascour:22," (3) ");
  writeln(cl,':margin,"Finish ",row8.masfin:22," (4) ");
  writeln(cl,':margin,"Use ",row8.masuse:22," (5) ");
  writeln(cl,':margin,"Condition ",row8.mascond:22," (6) ");
  writeln(cl,':margin,"Hard or Soft: ",row8.mortsoft);
writein(cl, ':margin,[Color | row8.mascot:22],| Mortar condition: | row8.mortcond));
writein(cl, ':margin,|----------------| Mortar analysis (Y/N): | row8.mortanal));
writein(cl, ':margin,|28. ',row11.untretcd);
writein(cl, ':margin,| Existing Condition of Untreated Masonry (Supplemented with photographs));
writein(cl, ':margin,|29. ',row11.coatcond);
writein(cl, ':margin,| Existing Condition of Surface Coating (if applicable));
writein(cl, ':margin,|30. ',row11.moistprb,' 32. ',row11.specs);
writein(cl, ':margin,| Evidence of Moisture Problems (Supplement with photographs)  Specs Y/N');
writein(cl, ':margin,|31. Previous Treatment 	 Date 	 Previous Treatment 	 Date');
writein(cl, ':margin,|2,row11.trt1,' ',row11.trtdat1,' ',row11.trt3,' ',row11.trtdat3);
writein(cl, ':margin,|2,row11.trt2,' ',row11.trtdat2,' ',row11.trt4,' ',row11.trtdat4);
writein(cl, ':margin,|33. Describe Masonry Condition as a Result of Previous Treatment');
writein(cl, ':margin,|34. Describe Condition of Masonry requiring treatment prescribed in Part III');
writein(cl, ':margin,|35. Masonry type number: | row11.number:2);
if null(ads row8.masfin) then row8.masfin := ;
if null(ads row8.masuse) then row8.masuse := ';
if null(ads row8.mascond) then row8.mascond := ';
if null(ads row8.mascolr) then row8.mascolr := ';
writeln(cl, ' | Margin | Masonry type | ');
writeln(cl, ' | Margin | Type | row8.mastype:22 | ');
writeln(cl, ' | Margin | Quarry | row8.masonry:22 | ');
writeln(cl, ' | Margin | Coursing | row8.masour:22 | ');
writeln(cl, ' | Margin | Finish | row8.masfin:22 | ');
writeln(cl, ' | Margin | Use | row8.masuse:22 | ');
writeln(cl, ' | Margin | Condition | row8.mascond:22 | ');
writeln(cl, ' | Margin | Color | row8.mascolr:22 | ');
writeln(cl, ' | Margin | ');
close(c1);
end;

procedure part_three_long;

var  margin : integer;
s1,s2,s3,s4 : string50;
begin
assignic1,'pt1:';
rewrite(c1);
margin := 5;
if row7.number = row8.number then begin
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl,' 3) ',row7.prod3);
writeln(cl,' 38. Temperature ',row7.temp,' Humidity ',row7.humidity,' Precipitation ',row7.precip);
writeln(cl,' 39. Location of Treatment (define a limited area) ');
writeln(cl,' 40. Describe Immediate Post-Treatment Appearance of Masonry ');
bstring(row7.appear,s1,s2,s3,s4);
writeln(cl,' ',s1:50);
writeln(cl,' ',s2:50);
writeln(cl,' ',s3:50);
writeln(cl,' ',s4:50);
sl := 'Masonry type number: ',row7.number;
write(cl,' 50,'sl);
t := 
write(cl,' 51,'sl);
write(cl,' 52,'sl);
if null(ads.row7.prod1) then row7.prod1 := 1;
if null(ads.row7.prod2) then row7.prod2 := 1;
if null(ads.row7.prod3) then row7.prod3 := 1;

writeln(cl,"margin",8,'Name of Product(s) used if Applicable');
writeln(cl,"margin",1,'row7.prod1');
writeln(cl,"margin",2,'row7.prod2');
writeln(cl,"margin",3,'row7.prod3');

writeln(cl,"margin",9,'Extent of treatment (Area treated)');

writeln(cl,"margin",9,'Reason for the treatment');
bstring(row7.appear,s1,s2,s3,s4);
writeln(cl,"margin",50,'Masonry type number : ',row7.number:2);
writeln(cl,"margin",50,'Treatment number : ',row7.tnum:2);
if (n mod 2) <> 0 then writeln(cl,'form_feed');
end;
close(cl);
end;

procedure part_four_long;
begin
margin := 5;
assign(cl,'Iptl:');
rewrite(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl);
writeln(cl,"margin","RECORDERS INFORMATION");
writeln(cl);
write(cl,"margin",42,'NAME & ADDRESS OF RECORDER');
write(cl,"margin",42,row4.rname:40,row4.rphone:14);
write(cl,"margin",42,'Name (Last name first) Phone #');
write(cl,"margin",42,row4.rname:40,row4.rstreet:30,row4.rorgan:30);
write(cl,"margin",42,'Address Organization');
write(cl,"margin",42,row4.rcity:20,row4.rstate:2,row4.rzip:10);
write(cl,"margin",42,'City Zip');
write(cl,"margin",42);;
write(cl,"margin",42,'Date of Entry');
write(cl,"margin",42,row4.entdate:8);
write(cl,"margin",42);----- Refer any questions to ------);
write(cl,"margin",42,'Preservation Assistance Division (424) (');
write(cl,"margin",42,'National Park Service (');
write(cl,"margin",42,'P.O. Box 37127 (');
write(cl,"margin",42,'Washington, D.C. 20013-7127 (');
writeln(cl,' :margin,'RECORDER5 INFORMATION');
writeln(cl,' :margin,'NAME & ADDRESS OF RECORDER');
writeln(cl,' :margin,','10. ',row4.rname, ',row4.rphone);
writeln(cl,' :margin,' 	 Name (Last name first) 	 Phone #');
writeln(cl,' 	 :margin,' 	 Date of Entry');
writeln(cl,' :margin,' 	 > Refer any questions to ( 	 11);
writeln(cl,' 	 :margin,')
writein(chform_feed);
close(cl);
U. S. National Park Service:

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS:

FOLLOW-UP REPORT:

1. Name of Structure;
2. Name of Primary Owner(s) Phone #;
3. Treatment being monitored;
4. Name of Product(s) used if Applicable;
5. Has treatment changed appearance of area?

FOLLOW-UP INSPECTION INFORMATION:

U. S. National Park Service:

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS:

FOLLOW-UP REPORT:

1. Name of Structure;
2. Name of Primary Owner(s) Phone #;
3. Treatment being monitored;
4. Name of Product(s) used if Applicable;
5. Has treatment changed appearance of area?
WRITELN(CL,'6. Visual description of treated area');
BSTRING(ROW12.AREADES,S1,S2,S3,S4);
WRITELN(CL,'7. Is another treatment type planned? ',ROW12.FUTRT,' Explain below.');
BSTRING(ROW12.FUTDES,S1,S2,S3,S4);
WRITELN(CL,'8. Time elapsed: ',ROW12.LAPSETM,' Will this treatment be repeated? ',ROW12.REPTRT,' If so when? ',ROW12.REPDATE);
WRITELN(CL,'9. Comments:');
BSTRING(ROW12.COMMENTS,S1,S2,S3,S4);
WRITELN(CL,'U. S. National Park Service');
WRITELN(CL,'CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
WRITELN(CL,'FOLLOW-UP REPORT');
FOLLOW-UP INSPECTION INFORMATION

11. NAME & ADDRESS OF RECORDER

Name (Last name first)          Phone #
Title
Address           Organization
City           State           Zip

Refer any questions to:
Preservation Assistance Division (424) 
National Park Service 
P.O. Box 37127
Washington, D.C. 20013-7127 
(202)343-9567

---

procedure print_follow
begin
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ t;
wherao[1] := 'AND ';
natt := 1;
b_ask;
clear_screen(20,0,24,79);
brfind(path[2],relname[6]);
brcheck('rbfind ',2);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
brstat(rbs);
if rbs -1 then begin
repeat
natt := 1;
bright;cur_pos(22,20);write('Prepare Printer for Follow-up Report');
cur_pos(23,25);write('Press any key to continue ');
spchin(ch);
brget(path[2],ads row6);
brstat(rbsl);
if rbsl -1 then print_follow_up;
until rbsl = -1;
end else begin
cur_pos(19,27);write('No FOLLOW-UP reports exist for this building');
end;
end;

procedure comp;
begin
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ ';
wherao[1] := 'AND ';
natt := 1;
clear_screen(20,0;24,79);
bright;cur_pos(22,20);write('Prepare Printer for Comprehensive Report');
cur_pos(23,25);write('Press any key to continue ');
spchint(ch);

rbfind(path[1],relname[1]);
rbccheck('rbfind ',1);
rwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
rbccheck('rbwher ',1);
rbget(path[1],ads row1);
if bnum[3] = 'C' then part_one_long else part_one_short;
rbfind(path[2],relname[8]);
rbccheck('rbfind ',2);
rwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
rbccheck('rbwher ',2);
rbfind(path[3],relname[11]);
rbccheck('rbfind ',3);
rwher(path[3],attlist[1],wherap[1], ads where,wherao[1],natt);
rbccheck('rbwher ',3);
repeat
  natt := 1;
  rbget(path[2],ads row6);
rbs2 := 0;
  repeat
    if bnum[3] <> 'C' then begin
      if bnum[3] = 'C' then part_two_long else part_two_short;
      natt := 1;
    end;
    rbfind(path[4],relname[7]);
rbccheck('rbfind ',4);
rwher(path[4],attlist[1],wherop[1], ads where,wherao[1],natt);
rbccheck('rbwher ',4);
    n := 0;
    repeat
      rbget(path[4],ads row7);
rbs2 := 0;
      repeat
        if bnum[3] <> 'C' then begin
          if bnum[3] = 'C' then part_three_long else part_three_short;
        end;
        until rbget = -1;
      end;
    end;
  end;
  until bnum[3] = 'C' then part_four_long else part_four_short;
end;
program CEN(input;output);

uses census;

procedure print_follow;extern;
procedure comp;extern;

VAR dwayne_null : names;
    space     : char;

begin
dbname := 'C:CENSUS '; rbstrt;
    blank := ' ';
dataerr := false;
edfunc := 0;
dwayne_null := 'NULL ';
    space := chr(32);
    rbopen (dbname);
    rbset(dwayne_null; space);
    path[1] := 1;
    path[2] := 2;
    path[3] := 3;
    path[5] := 5;
    relname[1] := 'IDENT1 ';
    relname[3] := 'IDENT3 ';
    relname[7] := 'TREATMNT';
    relname[8] := 'STUDY1 ';
    relname[9] := 'MAT1 ';
    relname[10] := 'MAT2 ';
    relname[12] := 'FOLLOW2 ';
    relname[13] := 'FOLLOW3 ';
date(todays_date);
time mark;
    sbrint;
caps_lock;
    clear_screen(2;0;23;79);
main_menu;
end.