P.R.L.
TECHNICAL NOTE

TN-89-64

DATA ENTRY PACKAGE FOR DEC-SYSTEM

Jeena Thomas and D.R.Kulkarni

March 1989

PHYSICAL RESEARCH LABORATORY
AHMEDABAD-9

DOCUMENT CONTROL AND DATA SHEET

1. REPORT NO. : PRL : T8-89-64

2. TITLE AND SUBTITLE : "DATA ENTRY PACKAGE FOR DEC-SYSTEM"

3. REPORT DATE: MARCH, 1989

4. TYPE OF REPORT: TECHNICAL NOTE

5. PAGES :

6. PRICE : UNPRICED

7. NO. OF REFERENCES:

8. AUTHORS : JEENA THOMAS AND D.R.KULKARNI

9. a) PURPOSE : DATA ENTRY OPERATION ON DEC-SYSTEM

b) USEFUL FOR : FOR ANY INTENSIVE DATA PROCESSING WORK ON DEC-SYSTEM

10. ORIGINATING UNITS/DIVISIONS AND ADDRESS:

Computer Centre
Physical Research Laboratory
Navrangpura
Ahmedabad 380009

11. SPONSORING AGENCY : NIL

12. ABSTRACT : A DATA ENTRY PACKAGE HAS BEEN DEVELOPED FOR THE DEC-1091 COMPUTER SYSTEM. THE PACKAGE WHICH HAS BEEN CODED IN PASCAL MAKES USE OF VARIOUS FEATURES OF VT100 TERMINAL. THE PACKAGE ACCEPTS THE STRUCTURE OF THE USER-DEFINED RECORD BEFORE ACTUAL DATA-ENTRY PROCESS IS STARTED. THE COMPLETE RECORD STRUCTURE APPEARS ON THE SCREEN IN A SUITABLY DESIGNED FORM ALONG WITH THE BRACKETTED BLANK SPACE CORRESPONDING TO FIELD LENGTHS. THE SALIENT FEATURE OF THE PACKAGE IS THAT THE DATA ARE IMMEDIATELY VALIDATED AS THEY ARE BEING ENTERED. FURTHER THE PACKAGE OFFERS MANY OTHER OPERATIONAL FACILITIES THAT ENHANCES ITS UTILITY AND EFFECTIVENESS.

13. KEYWORDS : DATA ENTRY, DATA VERIFICATION, SCREEN DESIGN, VT100

14. DISTRIBUTION STATEMENT : DISTRIBUTION UNLIMITED

15. SECURITY CLASSIFICATION : Unclassified

DATA ENTRY PACKAGE FOR THE DEC-SYSTEM

Jeena Thomas and D.R.Kulkarni

ABSTRACT

An extremely user-friendly, screen-oriented data entry package has been developed for the DEC-1091 computer system under the operating system TOPS-10. The package which has been coded in Pascal makes use of various features of VT100 terminal. The package accepts the structure of the user-defined record before actual data-entry process is started. During data-entry session, the complete record structure appears on the screen in a suitably designed form along with the bracketted blank space corresponding to field lengths. Besides being flexible and convenient for inputting the data, the salient feature of the package is that the data are immediately validated as they are being entered. Further the package offers many other operational facilities that enhances its utility and effectiveness.

Data Entry Package for the DEC-system

1. INTRODUCTION

In an intensive data processing environment, a data entry (DE) package plays a very crucial roll. We describe in this note, a DE package developed on the system DEC 1091. The package has been coded in an ISO (International Standard Organization) Pascal consisting of 2500 lines. The package uses various features of the VT100 terminal to increase its effectiveness and user-friendliness. The salient features of the package are as follows:

1. *It enables inputting the data in an extremely convenient and user-friendly manner.*

2. *It validates the data as they are being entered. It may be worth noting that, while entering the data, user need not bother about the format specification of the data field. He should only see that he does not exceed the maximum length specification of the field conveniently indicated on the screen by square brackets. The package consists of three modules.*

    1. *VT100 Module (69 procedures)*
    2. *Record Structure Module (11 procedures)*
    3. *Data Entry Module (13 procedures)*

The VT100 module consists of procedures which can set its features dynamically during execution. This package can be used for other type of terminals as well provided the VT100 module is suitably replaced for the required terminal. The second

module is used to accept the information regarding the structure of the user-defined record. This information is being used by the third module which is the main data entry module. The following sections describes these modules in detail.

II. VT100 MODULE

The VT100 terminal is a simple device to operate. This is basically a typewriter that uses a video screen instead of paper and communicates with a computer. The VT100 terminal is both an input and output device for the computer.

Attached to this terminal there is a main keyboard like a typewriter and a numeric keypad. In addition to the usual typewriter keys on the main keyboard, the VT100 keyboard has other keys and indicators used to generate control sequences, cursor control commands and to show the current terminal status. The keys in the numeric keypad generates the same characters as the corresponding numeric key on the main keyboard. However, these keys can also be interpreted by the host computer as special function keys if programmed suitably.

The VT100 terminal has a large number of optional built-in functions which can be selected by the user. There is also a provision to store the selected features in its non-volatile memory either permanently or temporarily. The selection and storage of built-in terminal features is performed in a special mode of operation called SET-UP mode. When you enter the SET-UP mode, the status of the features stored is shown on the screen.

SET-UP mode provides two brief summaries of the current feature status. The first presentation SET-UP A displays the location of the tab stops set in the terminal and a visual ruler which numbers each character position on the line. The second- SET-UP B gives the status of other terminal features such as scroll (jump or smooth), autorepeat(on or off), screen (dark or light background), cursor (block or underlined), keyclick, margin bell, wraparound, new line, interlace, bits per character etc.

The VT100 is capable of displaying either 80 or 132 characters per line. In the 80 character per line mode, screen is 80 character wide by 24 lines high and in the 132 character per line mode, the screen is 132 character wide by 14 lines high. The lines of the VT100 terminal can be scrolled up or scrolled down to make space for new lines at the bottom or top of the screen.

We have described above how the features of the terminal can be set up mannually. However, it is possible to set them even through programming so that they can be changed by the application program dynamically during execution as desired. We have developed the Pascal procedures to set most of the terminal features. The list of these procedures has been given in appendix. The detailed description can be found in the VT100 user guide (1).

The features of the terminal can be set through programming by sending the appropriate escape sequences which are all preceeded by what is called 'Control Sequence Introducer(CSI)'. The CSI, which itself is a sequence of characters, affects the interpretation of a limited number of contiguous characters that follow it. In the VT100 the CSI is ESC[. The control sequence for each setting is given in the bracket along with the procedure name in the appendix.

III. RECORD STRUCTURE MODULE

For any data entry package, it is necessary beforehand to know the structure of the record defined by the user. The main program 'Structure.Pas' has been developed to accept the structure of a user-defined record. This program generates a file 'X.FDT' containing the information regarding the structure of the record to be used by the user. X is any user-supplied file name and FDT is its fixed system-assigned extension. The structure of any record is, in general, given in terms of following parameters, followed by their type and size of character strings.

i) The title of the record (max.25 alphabetic characters).

ii) Total number of fields in the record (max.2 numeric characters).

iii) Label of each field (max.15 alphabetic characters).

iv) The maximum length of the data contained in each field (max. three numeric characters). The present version of PASCAL allows a character string of max. 120 characters.

v) Code for the data in each field (1 alphabetic character).

vi) The range of values of the data in the field. This is given in terms of two fields each of maximum 8 characters. The type of the characters will depend on the code of the field. This field is optional.

The program accepts the data code for the fields as given below:

A - For alphabetic data

N - For numeric data

R - For real data

I - For integer data

M - For data in alpha-numeric and special characters.

In order to facilitate the inputting of the information regarding the structure of the record by the user, the program displays a suitable form on the screen containing the above parameters. The user can input the information for all the parameters pertaining to each field. After completing the information for a given field, the cursor goes back to the parameter (iii), so that you can feed the information for the next field. In this way you can feed the information for all the fields in the record. The salient feature of the program is that as soon as you enter the information for the parameter, it is validated before it is written on the file. There is also a provision that in case you put incorrect information, you can go back to the beginning of the parameter by just putting a '$' sign. This can be done both before and after pressing the enter key. Irrespective of the result of validation, the user gets one more chance to reenter the same data by pressing '$' and 'return' keys. If he does not wish to change the entry, he may press the 'return' key twice to move the cursor to the next parameter. As the cursor moves to the beginning of the next parameter, a message appears below the form which indicates the type of the characters and the number of characters to be entered in that parameter. In case the entry is invalid, the message 'invalid data' appears below the form and the cursor goes back to the beginning of the same parameter. When you want to input the code of the field, the list of the valid codes is displayed below the form for the convenience of the user.

As far as validation is concerned, the program validates the data according to its type of the character string specified for each parameter. For all codes the input is accepted in terms of characters. After due validation, the same character string is converted into integer numbers for the parameter (ii) and (iv). It may be noted that embedded blanks in the data entry of integer or real fields are considered invalid. The values in the range field will be, obviously, validated according to the code of the field. It is also ensured that the first value of the range is smaller than the second value. In case, the field code is R, the range values can be real numbers. The real data can be expressed both with or without exponentiation. This program has following procedures:

1. *Structure-Display (stadd):- To display the record structure.*
2. *Alpha-Valid (string,valid):- To validate the alphabetic data.*
3. *Num-Valid (string,valid):- To validate numeric data.*
4. *Mix-Valid(string,valid):- To validate character data.*
5. *Int-Valid (string,valid):- To validate integer data.*
6. *Real-Valid(String,valid):- To validate real data without exponentiation*
7. *Exp-Valid (string,valid):- To validate real data with exponentiation.*
8. *All-Blank(string,kmax,index):- To check if all the characters in the field are blank*
9. *Get-File(string):- To input the name of the output file by the user with a fixed extension FDT*
10. *Brack-Space(brack,nspace):- It types a given character n times.*
11. *Line(symbol ,nsymbol):- It types a given character n times with an Eoln procedure.*

Besides these procedures the program also makes use of various VT100 procedures given in appendix to invoke different features.

IV. DATA ENTRY MODULE

A main data entry program 'Problem.Pas' has been developed which reads the information from the file 'X.FDT' generated by the program 'structure.Pas' and displays a data entry form for the user-record on the screen using the information in the file. To be specific, the data entry form will display all the field labels in the user record followed by the number of bracketted blank spaces equal to the length of the corresponding fields. In case, the length of the field is so large that it can not be accommodated in a single line, it is appropriately split into two lines automatically. The program will also try to accommodate two fields in the same line if it is possible. It may, however, be noted that the order in which the fields will appear in the form is the same as supplied by the user. The spacing between the two fields in the form can be either single or double as specified by the user. The screen is sufficiently large for most of the records encountered in routine problems. However since the screen space is finite, it is possible that a record containing a large number of fields having very large field lengths may not be accommodated on a single screen. Below the form, there will appear four lines of messages giving the instructions to fill the form. This program generates a problem data file 'Y.DAT', where Y is any user - defined file name for output and DAT is its fixed system-assigned extension.

The operation of inputting the data is more or less similar to the one described in the case of the program 'Structure.Pas'. The user can enter the data in each field according to the message that appears below the form indicating the type of the character and the maximum number of characters in the field. The entered data will be validated immediately according to the code of the field before writing on the file. It is worth mentioning that even the blank data entry for any field is considered as invalid data entry. If you want to change the entry, you can go back to the beginning of the field, by just putting a '$' sign. The message to this effect appears permanently below the form. The user can go to the beginning of the next field, by pressing the return key twice. If the entered value is invalid, a message 'invalid data' appears below the form and the cursor goes back to the beginning of the same field. In the case of fields having a range parameter, it is ensured that the values entered should lie in the range specified. After inputting the first record successfully, the cursor goes back to the first field of the record automatically to facilitate the entry of the next record. As in the case of the program 'Structure.Pas', all information is accepted in terms of characters and then converted into real or integer values, if necessary, depending on the code.

Before you start entering the data, the user is supposed to provide a file name to store the data. If you stop the data entry session in the middle, you can start it once again

so that the new entries will be appended to the old file. In other words, a new file name needs to be given in the beginning of the session and the same is appended whenever the session is restarted. When all the records are entered, the session can be terminated by putting an '&' sign as the first character of the first field. On termination, appropriate message will appear below the form. This program uses the following procedures:

1. Record-Display (spacing, stadd, splitno):-
   To display the form consisting of all field labels in the record with proper spacing between the fields alongwith the bracketted space corresponding to field length.

2. Get-Outfile (string):-
   To input the name of the output file with a fixed extension DAT.

3. Put-Message (i, fld.no, fld-length, fld-code):-
   To display proper messages below the form to facilitate proper data entry.

Besides these procedures, this program also makes use of the validation and other miscellaneous procedures given in the earlier section and some of the VT100 procedures described in APPENDIX.

V. SUMMARY

A user-friendly, screen-oriented data entry package has been developed in ISO Pascal on the system DEC-1091 under the operating system TOPS-10. It may be worth noting that no such

package has been available on the system at present. The salient fectures of thc package are

1. Extreme convenience in inputting the data as the full record is presented as a form on the screen.
2. Eachafield label is displayed alongwith the bracketted blank space equal to its length.
3. The cursor moves from one field to another automatically.
4. The display of suitable instructions to input the data for each field in the form.
5. The validation of each field including the range check. In fact the package indicates the invalid entry and offers an apportunity to reenter the data again.
6. The opportunity to confirm the entry of the data at every field level.

Reference: VT100 USER GUIDE (Digital Equipment Corporation) 3rd Edition June 1981.

APPENDIX

PROCEDURES TO SET VT100 FEATURES

The list of Pascal Procedures to set various VT100 features is given below:

I. PROCEDURES FOR CURSOR MOVEMENTS

1. Up-Cursor (n) (Esc[nA) (default value:1)
2. Down-Cursor(n) (Esc[nB) (default value :1)
3. For-Cursor(n) (Esc[nC) (default value :1)
4. Back-Cursor(n) (Esc[nD) (default value:1)
5. Put-Cursor (n1,n2) (Esc[n1;n2H) (default value:1)
6. Down-Scroll (Esc D)
7. Up-Scroll (Esc M)
8. Next-Line (Esc E)
9. Save-Cursor (Esc 7)
10. Restire-Cursor (Esc 8)

II. PROCEDURES FOR ERASING

1. Cur-End-Erase (Esc[K) (default value:0)
2. Start-Cur-Erase (Esc[1K) (default value:0)
3. Line-Erase (Esc[2K) (default value:0)
4. Cur-Screen-Erase (Esc[J) (default value:0)
5. Screen-Cur-Erase(Esc[1J) (default value:0)
6. Erasre-Screen (Esc[2J) (default value:0)

III. PROCEDURES FOR LINE SIZE

1. Double-width (Esc#6)
2. Single-Ht-width (Esc#5)
3. Top-Double (Esc#3)
4. Bottom-Double (Esc#4)

The last two sequences, viz. Top-Double and Bottom-Double, must be used in pairs on adjacent lines and the same character output must be sent to both lines to form full double height characters.

IV. PROCEDURES FOR CHARACTER ATTRIBUTES

Without the Advanced Video Option (AVO), only one type of character attribute is possible as determined by the cursor selection. In this case specifying either the underscore or the reverse attribute will activate the currently selected attribute. This is because of the Hardware.

1. On-All (Esc[1;4;5;7m) (default value:0)
2. Off-All(Esc [0m) (default value:0)
3. On-Underscore[Esc(4m) (default value:0)
4. On-Reverse-Video(Esc[7m) (default value:0)

V. PROCEDURES FOR SETTING AND RESETTING THE FEATURES

1. New-Line(set mode) (Esc[20h)
2. Line-Feed(reset mode) (Esc[20l
3. Key-Application(set) (Esc[?1h)
4. Keymode-Cursor (reset) (Esc[?1l)

5. Col-132 (set) (Esc[?3h)

6. Col-80 (reset) (Esc[?3l)

7. Smooth-Scroll(set) (Esc[?4h)

8. Jump-Scroll (reset) (Esc[?4l)

9. Reverse-Screen(set) (Esc[?5h)

10. Normal-Video(reset) (Esc[?5l)

11. Relative-Origin (set) (Esc[?6h)

12. Absolute-Origin (reset) (Esc[?6l)

13. On-Wraparound (set) (Esc[?7h)

14. Off-Wraparound(reset) (Esc[?7l)

15. On-Autorepeat (set) (Esc[8h)

16. Off-Autorepeat (reset) (ESC[?8l)

17. On-Interlace (set) (Esc[?9h)

18. Off-Interlace(reset) (Esc[?9l)

19. Keypad-Application (set) (Esc=)

20. Key-Numeric (reset) (Esc>)

VI. PROCEDURES FOR CHARACTER SETS (GO AND G1 DESIGNATORS)

1. SGC-GO (Special Graphics) (Esc(0)

2. UK-GO (United Kingdom set) (Esc (A)

3. SGC-G1 (Esc)0)

4. USA-GO (Esc(B)

5. USA-G1 (Esc)B)

6. UK-G1 (Esc)A)

VII. MISCELLANEOUS PROCEDURES

1. One-Tab-Clear (esc[0g) (default value:0)
2. All-Tab-Clear (esc[3g) (default value:0)
3. Set-Tab (escH) (default value:0)
4. CPR (Cursor Position Report) (Esc[6n)
5. Scrolling (n1,n2) (Esc[n1;
6. Bell (Chr(7))
7. Reset (Esc c)
8. Test (Esc#8)
9. Test-Reset (Esc[2;1y)
10. Power-Test (Esc[2;2y)
11. Modem-Test (Esc[2;4y)
12. Select-Test (Esc[2;8y)
13. Clear-LED (Esc[0q) (default value:0)
14. Clear-L1 (Esc[1q) (default value:0)
15. Clear-L2 (Esc[2q) (default value:0)
16. Clear-L3 (Esc[3q) (default value:0)
17. Clear-L4 (Esc[4q) (default value:0)
18. Device-Attribute (Esc[0c) (default value:0)
19. Status (Esc[5n)