1976

The Terminal Process

Kevin E. Kolis

John B. Lohse

Peter Radtke

Robert Sedlemeyer

Ronald D. Bedford

Report Number:
76-201

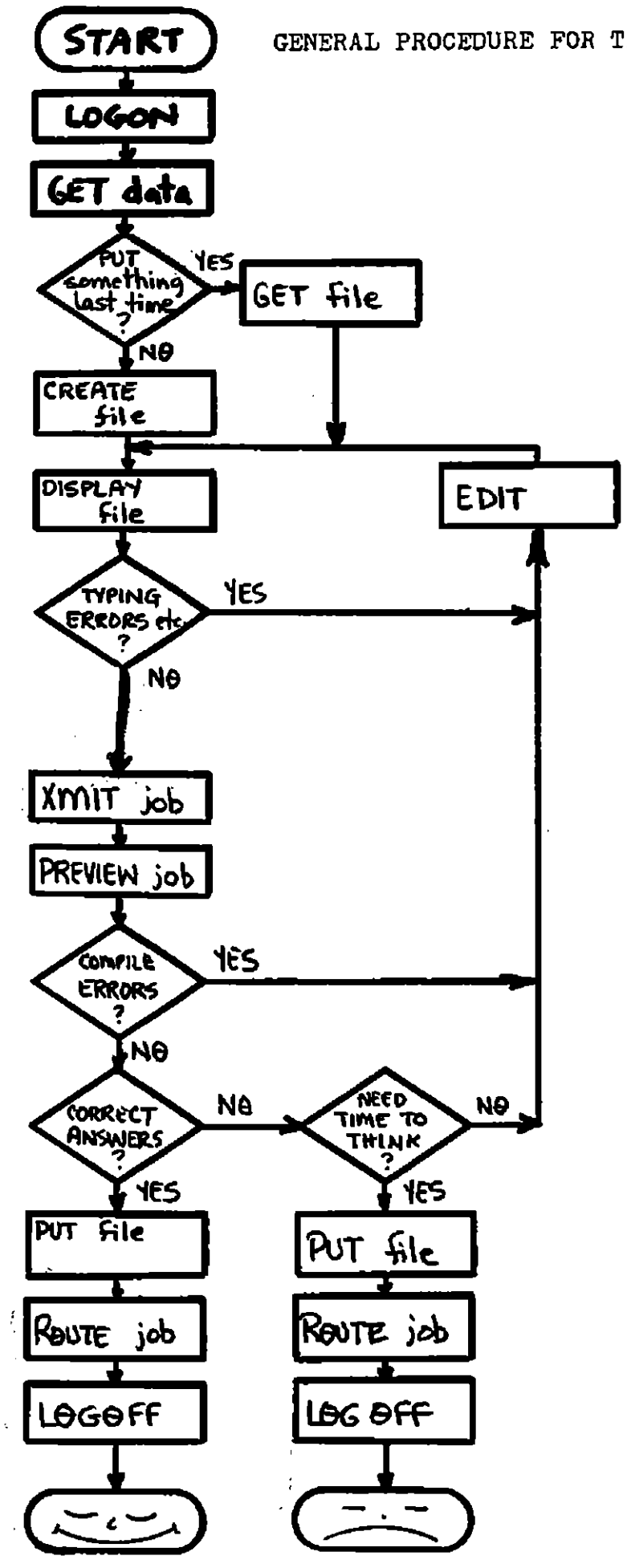
OCTOBER 1976

CSD-TR 201

THE TERMINAL PROCESS

BY

KEVIN E. KOLIS
JOHN B. LOHSE
PETER RADTKE
ROBERT SEDLMEYER
RONALD D. BEDFORD



INDEX

A QUICK REFERENCE GUIDE TO THE SYNTAX OF THESE COMMANDS IS ON THE BACK PAGE!

INTRODUCTION

This document is designed for New Users of the Terminal System (NUTS). It will step through the process of creating files, editing files, and running programs from the Purdue Remote On-line Console System (PROCSY).

In the examples given throughout this document, the characters typed by the user are underlined. Those characters not underlines are typed by the terminal.

Three important things should be noted by NUTS before attempting to use PROCSY:

1. The RETURN key must be pressed after the typing of each line.
2. To erase an entire line, press the RUBOUT key instead of RETURN. The system will respond with ^R and skip to the next line.
3. To erase individual characters, type the percent sign, '%'. One character is erased from each '%' typed starting with the last character.

   Ex: FORNT%%MAT yields FORMAT.

LOG-ON

This is the process of initiating a session at the terminal. First, flip the power switch to 'ON', and the 'ONLINE-OFF' switch to 'ONLINE'. To log-on to the system, press the 'CTRL' key, then while this key is depressed, type 'B'.

The terminal will then respond with some information, and ask for your account number, user ID, password, and the system desired. When asked for system, you should type PIRATE. Note that as a security measure, when you type your password, it is not printed.

Here is an example of the log-on procedure:

```
TCB L157  21.36.49   02/29/76.   FULL DUPLEX
ACCOUNT?  77510
USER ID?  FBR
PASSWORD?
SYSTEM?  PIRATE
  PIRATE VER. 5.11
+++
```

The terminal types '+++' as a prompt for your commands to the terminal.

LOG-OFF

To end a session at the terminal, enter the command:

```
+++LOG
```

The terminal will then respond with some accounting information, and request that you turn off the terminal (Please Do).

CREATE....CREATING A FILE

Creating a file from the terminal is analogous to punching a deck of cards from the keypunch. This "deck" or "file" is identified by a 'filename' which you assign at the time the file is created. To create a file, type (after the +++ prompt) the command:

```
+++CREATE,filename
```

where 'filename' is a name of your choice which may contain up to 7 characters. You must use this filename whenever you wish to reference the file. You will then be prompted with a line number. Enter the file, one line at a time, pressing 'RETURN' at the end of each line.

In the example which follows, we will create a file called KEY which will contain the control cards and the FORTRAN for a simple program.

```
+++CREATE,KEY
1.000=77510,FBR.
2.000=MNF.
3.000=#EOR
4.000=C
5.000=C      KEVIN E. KOLIS AND JOHN B. LOHSE
6.000=C
7.000=       PRINT   500
8.000=   500 FORMAT('1',1X,'X',2X,'SQUARE ROOT',1X,'SQUARE',2X,
9.000=      1        'CUBE',/)
10.000=      5 READ 501,X
                .
                .
                .
18.000=        STOP
19.000=        END
20.000=#EOR
21.000=#STOP
+++
```

Note several things about creating any file:

1) Do not include a password card (again, for security).

2) #EOR is used to indicate a 7/8/9 multipunch.

3) The same FORTRAN conventions apply, whether you punch cards or use a terminal (i.e. FORTRAN statements start in column 7, don't type past column 72, etc.).

4) #STOP is used to indicate that you are finished creating the file - it has no connection with the FORTRAN "STOP" statement. After pressing 'RETURN', another +++ prompt is issued.

5) If you cannot finish typing in your whole program during one session at the terminal do the following:
   a. Type #STOP
   b. PUT filename (see page 4)
   c. Now your file is saved and you may LOG OFF.
   d. When you return, GET filename (see page 5)
   e. Enter the editor (see page 7)
   f. Finish typing in your program using the INSERT command. (see page 8)

DISPLAY...DISPLAYING A FILE

It is often desirable to have the terminal type out the contents of a file. For example, it is usually a good idea to display a file after creating or editing. This aids in the early detection of typing errors, or omissions. The general form of the command to display a file is:

+++DISPLAY,filename

For example, the command

+++DISPLAY,KEY

will cause the entire file KEY, which we just created, to be typed out by the terminal. A file which has been obtained from PFILES (see the section on RETRIEVING A FILE for information on getting files) can, of course, be displayed in a similar fashion. After a file has been displayed, another +++ prompt is issued.

PUT.....SAVING A FILE

Ordinarily, at the end of a session at the terminal (that is, after logging-off) all the files that have been created during that session are lost. However, many times there is a need to save files which have been created, for use at a future session. For example, you may decide to give up on a program and to try again at a later time; or you may require the use of a correct program sometime in the future.

A file can be stored into the PFILES storage area associated with your three letter ID by using the command:

+++PUT,filename

A message will be printed by the terminal saying that the file was saved, and giving the length of the file.

To save the file KEY, which we created earlier, the following is used:

+++PUT,KEY

GET.....RETRIEVING A FILE

Once a file has been saved in your PFILES storage area, it can be retrieved at a later terminal session by typing the command:

+++GET,filename

For example, if we had earlier saved the file KEY, we could retrieve it by using the following

+++GET,KEY

When a file is saved in someone else's storage area, you must also specify that user's ID. This would be the case when an instructor saves the data file for your assignment. For example, assume that the data file for the sample program given earlier had been saved by your instructor (whose user ID is FEO), and that the name of the file is KPDATA. To retrieve this file for use, the following command would be needed:

+++GET,KPDATA//FEO

Of course, you would be informed of the proper ID and filename to specify in order to obtain your instructor's data file.

After any of the above GET commands is entered, the terminal responds with a message giving the length of the file, followed by a +++ prompt. Once a file has been retrieved, it can be used in all subsequent commands in exactly the same manner one would use a file created during the current terminal session. NOTE: It is only necessary to GET a file once during each terminal session.

XMIT.....SUBMITTING A JOB

The real usefulness of the terminal lies in the ability to actually run programs from the terminal. In order to run your job from the terminal, you must have available two files. The first file will contain the control cards for the job (without the PASS=password. card), followed by #EOR, then the FORTRAN program, and then another #EOR. The second file will contain the data for that program. These files must have been obtained earlier in the session by either CREATE (if the file was a new file that session, as it would be during the first terminal session with a new program) or GET (if one or both of the files had been previously saved in PFILES). The job may then be sent for execution via the XMIT command. The form of this command that we will use is as follows:

+++XMIT,jobname,file1,file2

where file1 is the file containing control cards and FORTRAN program, and file2 is the file containing the data.

This command causes your job to execute exactly as if you had combined together into one deck, the cards corresponding to file1 (file with control cards and program) and file2 (file with data), and then read this deck into the card reader. The "jobname" is a name of your choice containing up to four characters. The jobname serves to uniquely identify your job (the green-striped card serves this purpose when you are running programs from cards). You must pick a different "jobname" each time you wish to run your program.

Recall that the filename used in creating the file containing the control cards and program for the sample program was KEY. The data was obtained from PFILES on a separate file called KPDATA. The job could then be executed with the command:

+++XMIT,KP1,KEY,KPDATA

The terminal would then respond with:

JOB "KP1" SENT

PREVIEW...VIEWING JOB OUTPUT

After submitting a job to be executed, one would like to see the output from the terminal. The command for this is:

+++PREVIEW,jobname

where 'jobname' is the same name used in the XMIT command when the job was sent for execution. To view the output from the sample program which we sent under the jobname KP1:

+++PREVIEW,KP1

The terminal would then respond typing out first the "dayfile", a history of the execution of the job (that information appearing on the first page of your printed output), and then the "output", consisting of the program listing followed by the output produced by the program. A note about your job's output--the output will be single spaced regardless of the carriage control used. The carriage control has no effect when output is printed at the terminal, though it will work as expected if you have the job printed on the printer (see ROUTE below).

If you attempt to PREVIEW your job before it has completed execution, the terminal will periodically type a line giving the status of the job until execution is complete. After the job has completed, the terminal will then respond with the dayfile and output as stated above.

ROUTE....PRINTING A JOB

Unlike running programs from cards, when a job is sent from the terminal a printed listing of the program is not automatically produced. This, of course, is not necessary since the output can be examined by using the PREVIEW command. Eventually, however, the time will come when a printed listing of the job is desired, either because the results are correct and the program is done, or the error is baffling and time to think is needed. In any case, to produce a printed listing, use the command:

+++ROUTE,jobname

where again 'jobname' is the jobname used when the job was sent for execution with the XMIT command.

After this command is entered, the terminal will respond giving the number of the folder in which the output can be found (with any luck). The output can then be picked up in the basement of the Math-Science building (regardless of the location of the terminal) exactly as if the job had been sent using cards. To print the sample program (jobname KP1):

```
+++ROUTE,KP1
  FBRKP1   =   FBRO417  AT MATH
```

Note that the initial three letters correspond to the ID under which the user logged-on. If you wish to pick up your output at ENAD type

```
+++ROUTE,KP1,TO,ENAD
```

(You may wish to save your file at this time (see PUT). To log-off, type LOG).

EDIT.....THE EDITOR — AN INTRODUCTION

The editor is one of the most powerful tools that the terminal user can employ. Without an editor, the user would be forced to re-CREATE a file each time an error was discovered. By using an editor, however, it is quite easy to correct a typing error, to insert a line which was omitted or to delete a line.

ENTERING THE EDITOR

The general form of the command which allows you to edit a file is:

```
+++EDIT,filename
```

From this point on you will be prompted with a "#" in order to remind you that you are using the editor. Only editor commands may be entered after a # prompt. None of the previously mentioned commands may be used until after you exit the editor.

EXITING THE EDITOR

When you have finished editing a file, you exit the editor by using the editor command:

```
#STOP
```

At this point you will be prompted with a +++ , and you may once again make use of any of the terminal commands described earlier.

USING THE EDITOR (AN EXAMPLE FOLLOWS)

To have a single line PRINTED:

```
#line-number PRINT
```

TO CHANGE some set of characters in a line, first have the line printed using the PRINT command above, then enter:

#CHANGE/bad stuff/good stuff/

The new version of the line is then automatically printed.

TO DELETE a line, first have the line printed, then enter:

#DELETE

TO INSERT some lines, first have the line printed which the insertion is to follow, then enter:

#INSERT

you will then be given a line number. From this point on, you can enter lines just as if you were creating a file. When you have finished inserting the lines type #.

To REPLACE a line first have the line which is to be replaced printed then enter:

#REPLACE

The old line will be deleted and you will be given a new line number. Now you should proceed as if you were inserting lines with the INSERT command.

AN EXAMPLE OF USING THE EDITOR

Assume we have just DISPLAYed the file KEY, and the following was printed:

```
+++DISPLAY,KEY
1.000=77510,FBR.
2.000=MNF.
3.000=#EOR
4.000=C
5.000=C     KEVIN E. KOLIS AND JOHN B. LOHSE
6.000=C
7.000=      PRINT 500
8.000=  500 FORMAT('1',1X,'X',2X,'SQUARE ROOT',1X,'SQUARE',2X
9.000=     1       'CUBE',/)
10.000=  5   READ  501,X
11.000=  501 FRMT (F3.1)
12.000=      IF,
13.000=      IF (EOG,9  99,100
14.000=  10  XRT = X**0.5
15.000=      XSQ = X**2
16.000=      XCUB= X**3
17.000=      PRINT 502,X,XRT,XSQ,XCUB
18.000=  502 FORMAT(' ',F3.1,4X,F5.3,4X,F5.2,2X,F6.2)
19.000=      GOTO 5
20.000=   99 STOP
21.000=      END
22.000=#EOR
```

You should note that several things are "incorrect"

1. The word FORMAT is misspelled in line 11.000
2. Line 12.000 was erroneously entered and should be deleted
3. Several comment cards are to be added
4. Line 13.000 has several errors.

We will rectify these errors by using the editor.

The steps to follow in making these changes to the file KEY are as follows:

1. Enter the editor
2. Have line 11.000 printed
3. Change the characters FRMT to FORMAT
4. Have line 12.000 printed
5. Delete it
6. Have line 6.000 printed
7. Insert the comment cards
8. Have line 13.000 printed
9. Replace line 13.000
10. Exit the editor

The entire process looks like the following:

```
+++EDIT,KEY                                        entering editor

#11.000 PRINT
 11.000=  501   FRMT(F3.1)                         correcting misspelling

#CHANGE/FRMT/FORMAT/
11.000=  501 FORMAT(F3.1)

#12.000 PRINT
12.000=     IF,                                    deleting bad line
#DELETE

#6.000 PRINT                                       inserting lines
6.000=C
#INSERT

6.010=C        THE FOLLOWING FORTRAN PROGRAM INPUTS A NUMBER
6.020=C        CALCULATES ITS SQUARE ROOT, SQUARE, AND CUBE
6.030=C        THEN OUTPUTS THESE FOUR VALUES
6.040=C
6.050=#

#13.000 PRINT
13.000=       IF (EOG,9  99,100                    replacing bad line
#REPLACE
12.010=      IF(EOF,5)99,10
12.020=#

#STOP                                              exiting editor
+++
```

SOME ADDITIONAL COMMENTS ON USING THE EDITOR

1. To have more than one line printed while in the editor type the beginning line number, then PRINT, then a number indicating how many lines you want printed.

   #line-number PRINT number-of-lines

   So, if you wanted to see 14 lines of your file beginning with line number 37.000, type

   #37.000 PRINT 14

2. If you have made many changes to your file (and especially if you are using one of the TV-screen terminals) you may periodically want to look at the current version of your file. To view the entire file, type

   # ↑ PRINT !

   Note, on some terminals you must use ↑ PRINT ↓ instead.

3. If a line has been really messed up, you may find it expedient to replace the entire line rather than changing parts of it.

4. If you want to remove some characters from a line, use the CHANGE command with nothing between the last two slashes.

   example:

   ```
   #37.000 PRINT
   37.000=C   THIS IS A DUMB COMMENT CARD

   #CHANGE/DUMB//
   37.000=C   THIS IS A  COMMENT CARD
   ```

5. When you use the CHANGE command:

   #CHANGE/bad stuff/good stuff/

   the left-most occurence of 'bad stuff' is replaced by 'good stuff'. This may not be what is desired.

   example:

   ```
   #106.000 PRINT
   106.000=  200 FORMOT (I5)

   #CHANGE/O/A/
   106.000=  200 FARMOT (I5)
   ```

obviously not what was wanted. A better way (one that works!):

#106.000 PRINT
106.000= 200 FORMOT (I5)

#CHANGE/MOT/MAT/
106.000= 200 FORMAT (I5)

Moral: be careful

6. If you need to add statements at the beginning of the file:

   a. Have the first line of your file printed.
   b. Replace this line with all statements which should precede it.
   c. Retype the first line.

This concludes the guide for NEW USERS of the Terminal System. If anything is unclear, see your instructor. Furthermore, you are encouraged to use the terminal system for future CS assignments. Your instructor can also direct you to more detailed information on using the terminal system, if you feel really motivated.

QUICK REFERENCE GUIDE TO THE SYNTAX OF COMMANDS

TERMINAL COMMANDS

+++CREATE,filename — 7 characters max. (see page 3)

+++DISPLAY,filename (see page 4)

+++GET,filename — program file (see page 5)

or

GET,filename//user id — data file

+++LOG (see page 2)

+++PREVIEW,jobname (see page 6)

+++PUT,filename (see page 4)

+++EDIT,filename (see page 7)

+++ROUTE,jobname (see page 6)

+++XMIT,jobname,file1,file2 — 4 characters max, different each time; program file; data file (see page 5)

EDITOR COMMANDS

#CHANGE/bad stuff/good stuff/ (see page 8)

#DELETE (see page 8)

#INSERT (see page 8)

#REPLACE (see page 8)

#line-number PRINT (see page 7)

#STOP (see page 7)

Things to remember

1. Press RETURN after entering a line.
2. "%" erases the last character typed.
3. Press RUBOUT to erase an entire line.
4. Press CONTROL-B to start terminal session.