rc 3600

musil text editor

MUSIL TEXT EDITOR

The technical information in this document, while correct at the time of publication, is liable to change without notice.

A/S REGNECENTRALEN

Documentation Department

Revision 1
July 1975
RCSL No.
42 - i 0276

TABLE OF CONTENTS

INTRODUCTION

The MUSIL Text Editor can be used to modify, up-date, or create MUSIL programs while sitting at the operator's console of the RC 3600. The editing procedure can be carried out on individual characters, on strings of characters, or on program segments.

The handiest device to use with Text Editor are the F 13 or F 15 Alphanumeric Display/ Keyboard, because they are fast and quiet, and deletions and changes are easiest to observe when using these devices, but the F 12 KSR Teletype is also preferred by many because of the permanent record it gives of the editing process. The F 14 Silent Printer/ Keyboard combines quiet operation with a permanent record.

Text Editor allows you to edit MUSIL programs with input from the card reader, paper tape reader, magnetic tape unit, or cassette tape unit and output to the paper tape punch, magnetic tape unit, cassette tape unit, line printer, or serial printer.

Text Editor can modify a program text either at the line or the character level. It does this by searching either for a string of characters or for an implicit line number. Text Editor provides these implicit line numbers, and also an implicit character pointer to the current character.

There are three types of Text Editor commands: input commands that put the user program -or a part of the user program - into the edit buffer, commands that modify the contents of the edit buffer, and commands that output the modified contents of the edit buffer.

At each point of the editing procedure there is a character pointer (CP) that points to the position that currently is available for operating on.

Each line that is read into the edit buffer is assigned an implicit sequential line number, beginning with 1, that is updated as the editing procedure progresses. Text Editor defines a line as text ending with a carriage return. When the user requests a line number, Text Editor calculates the line number by counting carriage returns from the beginning of the buffer.

All text is assumed to be in ASCII, and in the parity that is normal for the hardware used, see Introduction to MUSIL, for normal device parities.

The Logic of Text Editor

Text Editor views all input to it as a continuous stream of characters. This stream of characters is considered to be segmented into pages. A "page" is defined as a stream of characters up to a form feed character or to the end of input. Each page is segmented into lines. A "line" is defined as a stream of characters up to and including a carriage return.

In theory a page may be up to 4750 characters in length, but in practice no page may be this long, for within the 4750 characters there must be room for the Text Editor commands.

The editing process takes place in three steps:

1. read a page into the edit buffer,
2. edit this page,
3. output this page, as modified.

When Text Editor has been loaded, it prints an * (asterisk), and the user may begin editing. If at any point the capacity of the edit buffer is exceeded, then the appropriate error message will be printed.

The character pointer (CP) should be thought of as placed between two characters. Inserted characters will be placed between these characters. Operations on the current character will occur on the character to the right of the CP.

The Use of Special Keys

The Escape Key, ESC, is used for two purposes:

1. Striking ESC twice initiates processing of the command(s) just typed.
2. Striking ESC once after the argument to a command code delimits that argument from whatever may be typed after it.

When ESC is struck, a $ (dollar sign) is displayed.

Each command consists of one or two letters or a special graphic. This is the "command code". Some commands allow you to place a number in front of them. These numbers must be decimal integers between zero and ± 2047. Some command codes may be followed by a string argument.

Each command has the form:

n code string $

where $ represents the ESC key, and n and/or string may be absent, depending on the specific case.

To initiate processing of this command, the ESC key is pressed once more, if the command ends with a dollar sign, or twice if it does not.

Several commands may be written one after the other before processing is initiated. In this case we have a string of commands, such as

n code string$n code string$...........$

To execute this command string, the ESC key must be struck once more. Remember, that processing begins upon the reception of a sequence of two ESCs. An accidental third ESC will have no ill effects.

Carriage return, CR, will have no effect if struck within a command string, as long as it struck only between commands, and not in the middle of a command. If CR is struck within a string of characters, then Text Editor will assume that CR is part of this string.

RUBOUT is used to delete the last character input. On a teletype it is represented by a back arrow. See the Operators Guide for its effect on other devices. Repeated RUBOUTs delete as many characters as there are RUBOUTs, from right to left, until a "terminator" is reached. A "terminator" is any character whose ASCII code is below octal 32, such as ESC, new line, tabulation, form feed, etc. In practice the deletion process usually goes to the CR that signals the end of the preceeding line.

Example:	start :	FIRSTTT↑
	action:	two RUBOUTs
	result :	FIRST↑

The arrows indicate the position of the CP.

Tabs are simulated with spaces, that is, they appear as spaces on the operator device. On the output they appear as the TAB character followed by the RUBOUT character. Predefined TAB positions occur at columns 1, 9, 17, 25, etc. TAB positions cannot be redefined by the user. The judicious use of TABs makes a program attractive and easy for you and others to read.

NULL and LINE FEED will be ignored on input, but a line feed will be provided for every CR sent to the output device.

When the CTRL, Control Key, is pressed together with a character, then the ASCII sum of the combination is input.

Pressing the CTRL key together with H will cause the last character of the command just input to be deleted, and the deleted character will be echoed on the console device. As opposed to RUBOUT, CTRL H can delete terminators, as well as any other character.

Pressing CTRL L will insert a form feed character at the current position of the CP. It will be displayed as↑ L on the operator device.

Parity Errors

Parity is always checked on input from paper tape, where even parity is assumed. If an input character contains a parity error, then the following message will be delivered.

PARITY ERROR IN LINE n

If the user then examines that line, the character in error will appear as a /.

The occurrence of a parity error will cause the execution of some commands to come to a halt. These commands are

E, N, H, R, Q.

Beginning the Editing Procedure

Before editing can take place the machine must of course be in operation and the`necessary programs will have to be loaded. Consult the Operator's Guide for loading procedures.

Specifically, before beginning editing

The operating system must have been autoloaded,
All necessary driver programs must have been loaded,
The MUSIL interpreter must be loaded,
Text Editor must be loaded, and
The MUSIL program (if any) must be ready for input.

After program loading Text Editor takes command, clears its buffers, and is in control. This is indicated by the printing of an *.

Device Handling

When loading has been accomplished and Text Editor has assumed control, the initial asterisk delivered, and the user is ready to begin, Text Editor must be told which physical devices to use.

GR inputfilename$$	get for reading

will open the file with the specified name, after first closing a currently open input file, if any. If the file does not exist, then the message

NO SUCH FILE

will be displayed.

GW outputfilename$$	get for writing

will open an output file with name specified. If the file does not exist, then

NO SUCH FILE

will appear.

For files that, like magnetic tape, have set position possibilities, write

```
        GR inputfilename: xx$$
or      GW outputfilename:xx$$
```

where xx is the file number.

The input and output file names available with Text Editor are

Paper tape reader	$PTR
Paper tape punch	$PTP
Line printer	$LPT
Serial printer	$SP
Card reader	$CDRC
Magnetic tape unit 0	$MT0
Magnetic tape unit 1	$MT1
Magnetic tape unit 2	$MT2
Magnetic tape unit 3	$MT3
Cassette tape unit 0	$CT0
Cassette tape unit 1	$CT1

where the $ is a true dollar sign, and NOT the escape key.

To append multiple input files to produce a single output file, use the GR command for each input file, but declare only one output file. Remember that GR will close the current input file before opening a new one.

Device Errors

If a device error occurs, the message

```
devicename ERROR errornumber
CORRECT ERROR WRITE START/STOP
```

will be displayed.

To correct the error, consult the Operator's Guide for the error number. After you have made the correction, press RETURN. The operation at which the error occurred will repeat itself automatically.

If you cannot correct the error, write STOP and press RETURN. The program will then ignore the command that gave rise to the error and will prompt for a new command.

For example, you have put in

```
GW$LPT$$
```

and

```
LPT ERROR 21
CORRECT ERROR WRITE START/STOP
```

appears. This means that the line printer is off-line. To correct the situation, put the line printer on-line and press RETURN. Text Editor will now automatically execute

```
GW$LPT$$.
```

Input Commands

Y read a page

The next page of the user program is read into the edit buffer. The form feed at the end of the page will be read, but not stored in the edit buffer. The CP will be positioned to just before the first character in the edit buffer. If the edit buffer is exceeded, the following message will be printed:

BUFFER IS FULL-Y OR A INPUT IS TERMINATED

If this message is received, then a part of the current page is in the edit buffer. To continue the editing process, the user must either read out the buffer or delete something from it. The rest of the page can then be read in.

When the next page is to be input from the console device, then the Insert command is used instead of the Y command。

A append a page

The next page is added to the present contents of the edit buffer. The CP will be positioned just before the first character of this new page. If the buffer is full before the command is given, or if it fills up while the command is being executed, then

BUFFER IS FULL-Y OR A INPUT IS TERMINATED

will appear.

T display contents of edit buffer on console device

The CP is not affected. The entire contents of the edit buffer will appear on the display device. If the user wants to see only a part of the contents of the edit buffer, then to stop the execution of a current T command, press the RETURN key.

nT display n lines of edit buffer

will display n lines of the edit buffer, starting from the current position of the CP, and CP will <u>not</u> be moved.

Commands to the Character Pointer

B place CP at beginning of edit buffer

The CP is moved to just before the first character in the edit buffer. If an argument is used with this command, then the argument will be ignored.

nJ jump CP to line n

The CP will be moved to just before the first character of the n-th line of the edit buffer.

L move CP to beginning of line

Moves the CP to before the first character of the current line.

nL move CP n lines

The CP will move n lines from its current position and then place itself just before the first character of the new line:

for $n>0$,	move forward past n carriage returns
for $n<0$,	move backwards past $|n| + 1$ carriage returns and forward one character
for $n=0$,	place CP to just before the first character of the current line (equivalent to L)

If n is so large that it is attempted to move the CP past the limits of the edit buffer contents, then the command will result in the CP being placed after the last character of the buffer (if n is positive), or before the first character of the buffer (if n is negative).

nM move CP n characters

This moves the CP with respect to individual characters.

for $n>0$,	CP is moved n character positions forward
for $n<0$,	CP is moved $|n|$ character positions backwards
for $n=0$,	there is no effect on CP

Z move CP to end of edit buffer

CP ends up just after the last character of the edit buffer.

Example of Commands to the CP

△ represents a blank space
(n) represents the implicit line number
↙ represents Carriage Return

In the edit buffer originally is

```
(1)TABLE△=△#25△56△43↙(2)54△97△56↙(3)00△55△97↙
                          ↑CP
```

Subsequently:

```
Command               Result
   B$$    (1)TABLE△=△#25△56△43↙(2)54△97△56↙(3)00△55△97↙
             ↑CP
  3J$$    (1)TABLE△=△#25△56△43↙(2)54△97△56↙(3)00△55△97↙
                                              ↑CP
 -1L$$    (1)TABLE△=△#25△56△43↙(2)54△97△56↙(3)00△55△97↙
                                  ↑CP
   Z$$    (1)TABLE△=△#25△56△43↙(2)54△97△56↙(3)00△55△97↙
                                                       ↑CP
-15M$$    (1)TABLE△=△#25△56△43↙(2)54△97△56↙(3)00△55△97↙
                                     ↑CP
   L$$    (1)TABLE△=△#25△56△43↙(2)54△97△56↙(3)00△55△97↙
                                  ↑CP
  2J$$    (1)TABLE△=△#25△56△43↙(2)54△97△56↙(3)00△55△97↙
                                  ↑CP
 50M$$    (1)TABLE△=△#25△56△43↙(2)54△97△56↙(3)00△55△97↙
                                                       ↑CP
```

The same effect can be obtained by writing the commands as a single command string, thus:

```
B$3J$ – 1L$Z$ – 15M$L$2J$50M$$
```

Text Modification Commands

These commands often contain arguments (operands). Each argument is delimited by $. Thus, commands with arguments end with one $, as shown below, and to initiate execution only one more $ is needed, for a sum of $$.

Cstring$_1$\$string$_2$\$ search for and replace string

Will cause Text Editor to search forward (only) for (only) the first occurence of string$_1$ and replace it with string$_2$. If the end of the edit buffer is reached before the string is found, then

STR NOT FOUND

will be displayed and the CP will be placed before the first character of the edit buffer, allowing the user to search the rest of the buffer for the string by repeating the command. If the string is found, then the CP is placed just after the last character of string$_2$.

To simply delete a string, the C command can be used with an empty string$_2$, that is, we type

Cstring$ delete string

The CP will be placed at the point of the deletion.

Istring$ insert string

The string is inserted at the current position of the CP. The CP is then moved to just after the inserted string.

nD delete characters

This command has the following effect

for $n > 0$, delete n characters to the right of CP
for $n < 0$, delete $|n|$ characters to the left of CP
for $n = 0$, ignore command.

The CP is not moved.

nK delete n lines

n lines are deleted from the edit buffer, thus:

for $n > 0$, delete n lines forward from the CP
for $n < 0$, delete $|n| + 1$ lines backwards from the CP
for $n = 0$, delete everything before the CP, until a carriage return is reached.

After command execution, the CP will be placed just after the last deleted character.

Sstring$ search for string

The editor searches forward until the <u>first</u> occurence of the string, and positions CP to just after this first occurence. If the end of the edit buffer is reached before the string is found, then

STR NOT FOUND

is displayed and CP will be placed just before the first character of the edit buffer, enabling the rest of the buffer to be searched by a repetition of the command.

Nstring$ search, output, and read

This command causes Text Editor to search forward in the edit buffer for the string. If the end of the buffer is reached before the string is found, then Text Editor reads in the next page of the input until the buffer is filled again, having output the first buffer contents. This process continues until the first occurence of the string is found. If the end of the input is reached before then,
then

STR NOT FOUND

is displayed. The CP is placed just before an empty edit buffer, and all input is transferred to the output file.

The N command can be used for copying magnetic or cassette tapes. It will stop execution only when the string is found or a double file mark is encountered. Single filemarks will, moreover, be copied. If the string to be searched for is given in such a way that it does not exist on the tape, then N can be used to copy the tape, including its single file marks.

Tapes copied in such a way will not have double file marks on them. These must be put in by the execution of a GC command, described in the section on Output Commands.

The Macro Command

Text Editor provides for the definition and execution of a macro command. A macro command is defined as a single command which contains an arbitrary command string. Once defined, the macro command is retained by Text Editor for future execution by means of a single symbol. Text Editor allows for the definition of only one macro at a time.

To specify the contents of a macro command, issue the command

XMcommand$_1$command$_2$ .... command$_n$ create macro

where "command" has the format

n codestring$

The command

X execute macro

will cause the macro to be executed as defined.

nX execute macro n times

will cause the macro to be executed a number of times in succession. Note that

```
X$$, 0X$$, 1X$$
```

are all equivalent, and will cause the macro to be executed once.

A defined macro may be deleted at any time by issuing the command

XD delete macro

If a macro is recursive in execution, then the message

```
MACRO CONTENTS X IN COMMAND
```

will be displayed.

If the macro is not properly defined, then the message will be

```
MACRO UNDEFINED
??
the command in error
```

If an error occurs during the execution of the macro, then the string containing the illegal command will appear.

```
STRING NOT FOUND
??
the string
```

is what will appear when searching for a non-existent string.

Example of Text Modification Commands

△ represents a blank
(n) represents the implicit line number
↙ represents a Carriage Return

In the edit buffer originally is

```
(1)BEGIN↙(2)△△△△IF△IN.ZMODE= 0△THEM△OPEN(IN,1);↙
  ↑CP
```

Subsequently:

```
Command                        Result
CTHEM$THEN$$    (1)BEGIN↙(2)△△△△IF△IN.ZMODE=0△THEN△OPEN(IN,1);↙
                                                  ↑CP
CIN,1$$         (1)BEGIN↙(2)△△△△IF△IN.ZMODE=0△THEN△OPEN( );↙
                                                         ↑CP
IINN.1$$        (1)BEGIN↙(2)△△△△IF△IN.ZMODE=0△THEN△OPEN(INN.1);↙
                                                             ↑CP
-3M$$           (1)BEGIN↙(2)△△△△IF△IN.ZMODE=0△THEN△OPEN(INN.1);↙
                                                          ↑CP
3D$$            (1)BEGIN↙(2)△△△△IF△IN.ZMODE=0△THEN△OPEN(IN);↙
                                                          ↑CP
B2M$$           (1)BEGIN↙(2)△△△△IF△IN.ZMODE=0△THEN△OPEN(IN);↙
                     ↑CP
1K$$            (1)BE△△△△IF△IN.ZMODE= 0△THEN△OPEN(IN);↙
                     ↑CP
0K$$            (1)△△△△IF△IN.ZMODE=0△THEN△OPEN(IN);↙
                   ↑CP
2M$$            (1)△△△△IF△IN.ZMODE=0△THEN△OPEN(IN);↙
                     ↑CP
-1K$$           (1)△△IF△IN.ZMODE= 0△THEN△OPEN(IN);↙
                   ↑CP
IBEGIN$$        (1)BEGIN△△IF△IN.ZMODE=0△THEN△OPEN(IN);↙
                        ↑CP
```

Example of Macro Commands

△ represents a blank

(n) represents the implicit line number

↙ represents a carriage return

In the edit buffer originally is

```
(1)TABLE △ = △ # 25 △ 56 △ 43↙(2)54 ↑△ 97 △ 56↙(3)00 △ 56 △ 56↙
                                    CP
```

Subsequently :

we define the macro

```
XMN56$L$C56$51$$
```

then occurs :

```
command                           Result

2X       (1)TABLE △ = △ # 25 △ 56 △ 43↙(2)↑54 △ 97 △ 56↙(3)00 △ 56 △ 56↙
                                          CP

         (1)TABLE △ = △ # 25 △ 56 △ 43↙(2)54 △ 97 △ 51↙(3)00 △ 51 ↑△ 56↙
                                                                  CP
```

Output Commands

F form feed

This command outputs a form feed character to the output file, along with 10 zeroes before and after the form feed character. These zeroes will be ignored when the file is later read. When the output file is later read, and if it is to be used with a printer, then the occurrence of the form feed character will cause the printer to begin a new page.

nF n inches of leader output

The Editor will output n inches of leader, up to 100 inches. If n is greater than 100, then 100 inches will be output. In the case of paper tape this means that n inches of blank tape will be punched. 0F$$ causes the same effect as F$$. Neither the F nor the nF command has any effect on the CP.

P output buffer

The entire edit buffer is output, followed by a form feed preceded and followed by ten zeroes.

nP output n lines

Starting from the CP, n lines are output, plus a form feed, as above. If the end of the text in the buffer is reached before n lines are output, then output is terminated, and with a form feed.

PW output buffer, no form feed

The entire edit buffer is output and no form feed is inserted.

nPW output n lines, no form feed

Starting from the CP, n lines are output, but no form feed. If n is too big, output stops when the end of text in the edit buffer is reached.
The output commands have no effect on the CP. In counting the number of lines output, the part of the current line after the CP is counted as the first line.

E output buffer and remaining input

The current contents of the edit buffer plus the remaining input contained in the input device will be output.

R output and read page

One page of the edit buffer is output and the next page is read in. This command is equivalent to writing PY

nR output and read

The equivalent of a combination of P and Y commands, written n(PY), n pages are output and n pages are read into the edit buffer. 0R and 1R are equivalent.

↑ I insert tabulation

Insert a tabulation character in the output. Printing or display of the output would proceed as though spaces had been inserted until the next tab position. The CP will move to the next of the present tab positions.

When editing is completed, the output file must then be closed.

GC close output file

will close the output file. This command will not force the output of the last page. This must have been done previously by the execution of a P or an E command.

When using the N command for copying tapes, the execution of the GC command subsequently will have the effect of putting two file marks at the end of the output tape.

The output file buffer can be emptied only be using the GC or the H command.

H home, restart

activates Text Editor and causes it to give the asterisk prompt, after first emptying and closing all buffers. H is logically equivalent to E$GC.

Special Commands

: print number of lines

The number of lines that are in the edit buffer will be displayed.

. CP line number

The number of the line the CP is pointing to will be displayed.

= print number of characters

The number of characters that are in the edit buffer will be displayed.

Error Messages

BUFFER CAPACITY EXCEEDED DURING COMMAND INPUT, COMMAND IS TERMINATED	Command string exceeds the capacity of edit buffer.
BUFFER IS FULL - Y OR A INPUT IS TERMINATED	During a read, buffer capacity is exceeded. Part of a page has been read in.
PARITY ERROR IN LINE n	During a read, a parity error occurred in line n. When examined, the character in error will be replaced by a /.
STR NOT FOUND	Unsuccessful string search.
?? COMMAND STRING	Editor cannot understand the command. It displays the command it cannot understand plus the commands that follow it in the command string.
NO SUCH FILE	File name does not exist.
NO INPUT FILE	Input file has not been declared.
NO OUTPUT FILE	Output file has not been declared.
WRONG FILE NUMBER	Fileno = 0 on magnetic tape or cassette tape during a GR or a GW command.
MACRO CONTENTS X IN COMMAND	The macro is recursive.
MACRO UNDEFINED	No macro has been declared, or the macro has been improperly defined.
device name ERROR error number CORRECT ERROR WRITE STOP/START	A device error has occurred. Consult Operator's Guide for error number.

Command	Format	Meaning
A	A	Append a page
B	B	Place CP at beginning of edit buffer
C	$\text{Cstring}_1\$\text{string}_2\$$	Search for string$_1$ and replace it with string$_2$ (also used for string deletions)
D	nD	Delete n characters
E	E	Output buffer and remainder of input
F	F	Output a form feed
	nF	Output n inches of leader
GC	GC	Close output file
GR	GRinputfilename$	Get for reading
GW	GWoutputfilename$	Get for writing
H	H	Home, restart
H	CTRL H	Delete character
I	Istring$	Insert string
I	CTRL I	Insert tabulation
J	nJ	Jump CP to line n
K	nK	Delete n lines
L	L	Move CP to beginning of current line
	nL	Move CP n lines from current position
L	CTRL L	Insert form feed character
M	nM	Move CP n character positions
N	Nstring$	Search for string, if necessary through input and output result
P	P	Output edit buffer plus form feed
	nP	Output n lines plus form feed
PW	PW	Output edit buffer
	nPW	Output n lines
Q	Qstring$	Search for string, if necessary through input
R	R	Output edit buffer and read in next page
	nR	Output n pages and read in n pages
S	Sstring$	Search for string
T	T	Display contents of edit buffer
	nT	Display n lines of edit buffer
X	X	Execute macro
	nX	Execute macro n times
XD	XD	Delete macro
XM	(see page 10)	Create macro
Y	Y	Read a page
Z	Z	Place CP at end of edit buffer
=	=	Display number of characters in edit buffer
:	:	Display number of lines in edit buffer
.	.	Display the current line number

Appendix: Disc Cartridge Editor System

For those who wish to edit MUSIL source language programs to and/or from disc cartridge, a few special commands are provided in addition to the Text Editor commands already described.

In addition to the Text Editor commands to be found in the section Device Handling, the Disc Cartridge user must initialize the disc he will use. Initializing a disc consists of creating or up/dating the disc's catalog. If the user wishes to employ an empty disc, or to erase what had previously been written on the disc to be used, the command

GIDISC$$ get disc initialization and erase

will create a new catalog on the disc and erase whatever data had previously been written on it. This new catalog will be mirrored in core.

To use a disc without erasing previous information on it, use

GINIT$$ get disc initialization

which will read the present catalog on the disc into core and prepare the disc for use.

Once the disc has been made ready for use by one of the two above commands, the user might want to remove one of the files from the disc. In this case one writes

GKfilename$$ remove disc file

The keying of this command removes the named file from the disc and from its catalog. The file name must be at most 5 ASCII characters long, and the characters must be letters with ASCII value greater than 32_{10}.

If the RC 3600 system has more than one disc drive, then the system will always assume that the 'first' disc drive, DK0, is meant, unless the user specifies some other disc drive. To do this, write :n before the escape key, thus :

GKfilename:n$$

which will inform the system that drive n is meant. The same convention can be used with the disc initialization commands.

Example of Disc Cartridge Text Editor Commands

GIDISC$$	Initialize the first disc, erasing previous data.
GINIT:1$$	Initialize the disc unit 1 without erasing previous data.
GKTEST2:1$$	Remove the file called TEST2 from the disc unit 1.
GRTEST:1$$	Open (for reading) the file called TEST on the disc unit 1.
GWTEST2$$	Open (for writing) the file called TEST2 on the disc unit 0.
Y$$	Read a page (from the disc unit 1).

Disc Cartridge Error Messages

CATALOG IS FULL	No room for more catalog entries.
CATALOG ERROR	Catalog trouble or disc not connected.
DISC CORE IS FULL	No room for more data on the disc.
NAME TOO LONG	Filename has more than five characters.

There is room for over 1000 file entries in the disc's catalog.

To interpret error messages containing error numbers, consult the Disc Operating System Programmers' guide.

The user should remember that when discs are used the catalog processor and the catalog handler must be loaded, in addition to the device drivers and the editor program itself. As stated in the Disc Operating System Programmers' Guide, the catalog processor, CATP, must be loaded _before_ the catalog handler, CAT01.

The load sequence is, therefore,

Interpreter
Device drivers as needed
DP0
DP1 if needed
CATP
CAT01
Text Editor

Note that disc cartridge filenames do _not_ begin with a dollar sign.