MUSS
VOLUME 3

DOCUMENTATION FACILITIES

This section provides facilities for text processing, drawing flowcharts and automatic program generation from encoded flowchart descriptions.

UNIVERSITY OF MANCHESTER

1. Volume 3 CONTENTS

MUSS

DOC011

MANCHESTER UNIVERSITY - CONFIDENTIAL

ISSUE 8

1. DOCUMENTATION IMPLEMENTATION DESCRIPTION

Section 1 Version 1
Section 1.1 Descriptive Text Generation

General Description

This section provides a set of simple word processing facilities. The main one TEXT produces well formatted layout from an encoding that intersperses warning sequences with the text.

2. Interfaces

The interface assumed by this section is the MUSS library i.e., it requires commands for manipulating input and output streams and for character input output.

2.1. Hardware Interface

Various output devices suitable for document production are assumed. These must all have a full visual character set. Lineprinters and terminals are suitable but more sophisticated printers such as Diablo and Dot Matrix are also catered for. These must have the following additional characteristics.

variable character size
fractional LF
fractional reverse LF
underlining

2.2. Software Interface

1) TEXT (INPUT.FILE,OUTPUT.FILE,OUTPUT.DEVICE)

This procedure copies text from an input stream generating a layout suited to the specified type of output device.

2) SPELL (DICTIONARY.FILE.NAME,TEXT.FILE.NAME)

This procedures checks all words in the text file against all words in the dictionary file, producing a list of all words not found in the dictionary on the current file, which may be printed, edited or added to the dictionary as appropriate. SPELL has not yet been implemented.

3. Implementation

3.1. Outline of Operation

The basic mode of operation of TEXT involves copying words from input text to output, starting newlines/newpages as necessary to avoid splitting words across lines and spacing the words across a line to right justify them. If a warning character ('@') is encountered in the input text, the characters which follow are decoded causing a modification to the basic mode of operation. Facilities controlled in this way include indentation, margin and tabulation settings, underlining, centring, forced newlines/pages and gaps for diagrams etc. Tables and diagrams can be built up by this means. In addition library procedures are called directly when "@*" is detected, allowing flowcharts for example to be generated within a document.

3.2. Data Structures

3.2.1. TEXT

DEVNAMES	a vector of 32-bit words containing the names of the output devices.
LSIZES	a vector of integers specifying the number of characters on a line for each device.
PSIZES	a vector of integers specifying the number of lines/page for each device.
SPECIAL	a vector of bytes indicating, for each device, whether it has special hardware characteristics, e.g. 'DS', 'VSP'.
INITSEQ	a vector of bytes used to initialise DIABLO type devices.
HDFORM	a vector of bytes describing the structure of the numbers 0 – 9 and letters A -> Z as produced by the HEADER facility (@H).
BUFF	a vector of bytes which is used to buffer each line of output.

TABLIST	a vector of integers indexed by LPOS specifying, for each output character position, the number of spaces to be output in order to advance to the next tabulation stop.
PAGEMSG	a byte vector used to hold the general heading output at the top of each new page.
PAGEMSGZ	an integer holding the number of bytes in the heading
CHAPSTR	a byte vector holding the current chapter heading
CHAPTER	an integer holding the chapter number.
BUFFPTR	an integer indexing the output buffer BUFF, specifying the position of the last character placed in the buffer.
LPOS	an integer specifying the current character position on the output device, equivalent to BUFFPTR.
DEVNO	an integer indexing the current device, as specified in DEVNAMES.
ULSW	a bit significant word indicating the current underline status. Bit 0 = character underline. Bit 1 = word underline. Bit 2 = line underline.
INSTREAM	an integer specifying the input stream associated with the input text.
OUTSTREAM	an integer specifying the output stream associated with the output text.
OLD.INSTREAM	an integer specifying the input stream selected prior to entering TEXT.
OLD.OUTSTREAM	an integer specifying the output stream selected prior to entering TEXT.
CONTSTR	an integer specifying the output stream number for the contents file.
INDSTR	an integer specifying the output stream number for the index file.
PAGE.GAP	an integer specifying the number of blank lines to be output at the start of a page.
BGAP	an integer variable giving the number of lines to be output at the start of a block.
SGAP	an integer variable giving the number of newlines to be output at the start of a section.
BIND	an integer variable giving the number of spaces of indentation to be output at the start of a new paragraph.
MARGIN	an integer variable giving the number of spaces to be output for the margin.
RMARGIN	an integer specifying the number of spaces in the right hand margin.
NDENT	an integer variable giving the number of spaces to be output for indentation following centring.
LSIZE	an integer variable initialised from LSIZES[DEVNO] giving the number of characters on a line.

PSIZE	an integer variable initialised from PSIZES[DEVNO] giving number of lines on a page.
SPEC.DEV	an integer variable, initialised from SPECIAL[DEVNO], indicating whether output device has special characteristics. bit 0 indicates variable space capability, bit 1 indicates underline capability, bit 2 indicates half line feed capability.
HIGHLIGHT	an integer flag which when positive indicates highlighting of the current line.
PCOUNT	an integer giving the page count.
LCOUNT	an integer giving the number of lines output on the current page.
SSW	an integer indicating that a section heading has been specified. Reset by @B.
DATE	an integer which indicates whether or not the date should appear at the foot of each page.
INDENT	an integer used to hold the current indentation position.
HDCHAR	a byte holding the character to be used in generating headings.
CH	a byte variable used to hold the character currently being processed.
AT	a byte variable used to hold the character currently being used as the warning sequence character. (@ is the default).
UP	a byte variable used to hold the character being used as the superscript character. (Default {).
DOWN	a byte variable used to hold the character being used as the subscript character. (Default }).
TABCH	a byte variable used to hold the character being used as the tabulation character. (Default %).
USER.FORM	an integer indicating whether the current line is user formatted (bit 0) and whether it is a heading (bit 1).
U.F.DIAG	an integer indicating the number of lines still required for outputting a user formatted diagram.
DISPLNS	an integer specifying the number of lines of displaced space/text awaiting output. A positive value indicates displaced space, a negative one displaced text.
SAVPOS1, 2, 3	32 bit integers used to hold values of the input stream pointer – used during storage & output of displaced text.
SAVPL1, 2, 3	as above but holding the page, line number of the input stream.

3.3. Special Notes

None.

4. Compile Jobs

The jobs presented here comprise two compiles for the VAX and one for the PDP11. The VAX jobs compile all the documentation software (TEXT,SPELL,LIST.INDEX,LIST.DICT,LIST.MOD,FLIP,DRAW) together while the PDP11 job cross-compiles only (on VAX for PDP11) the elements contained in this module (TEXT,SPELL,LIST.INDEX,LIST.DICT,LISTMOD). The first of the VAX jobs allows cross-compilation from one version of the system to the other, the second will produce a private library called DOCV in the directory of the calling process for test and development purposes.

4.1. Cross-compile Job for VAX.

This job allows all the documentation software (TEXT,SPELL,LIST.INDEX, LIST.DICT,LISTMOD,FLIP,DRAW) to be cross-compiled from one version to the other. The '?' symbol specified in some of the filenames is a substitute for the version number being compiled to, and will be altered by the actual compile jobs under MUSM to a 1 or 2 prior to entry.

```
::BEGIN COMPDOC
DEFINEOUTPUT 0 DOC3?LOG %200
LIB MUTLX3
LIB MUSLX
LIB LIB02X3
LIB VADIR?
ED DOC021/MU6S
(FD/::MU6 /)
E
FLIP 0 1
DOC02
**DI 4 0
MUSL 0 DOC3? %405 16
*TLSEG 0 %0 %7FEA0000 %F(8) 0;
*TLSEG 1 %0  0 %F(7)D 0;
*TLSEG 2 %0 %3B0000 %F(7)D 0;
*TLSEG 3 %0 %3B0000 %F(7)D 0;
*TLSEG 4 %0 %3B0000 %F(7)D 0;
*TLLOAD 1 5;
*TLLOAD 2 3;
*TLLOAD 3 2;
*TLLOAD 4 6;
*INFORM %2400;
```

```
**SELECTINPUT 4
ENDINPUT 4
ED DOC031/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC03
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC041/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC04
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC051/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC05
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC061/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC06
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC071/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC07
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC081/MU6S
(FD/::MU6 /)
E
```

```
FLIP 0 1
DOC08
**DI 4 0
MUSL 0 0 %C00
**SELECTINPUT 4
ENDINPUT 4
ED DOC091/MU6S
(FD/::MU6 /)
E
FLIP 0 1
DOC09
**DI 4 0
MUSL 0 0 %C00
**SELECTINPUT 4
ENDINPUT 4
ED EDT021/MU6S
(FD/::MU6 /)
E
FLIP 0 1
EDT02
**DI 4 0
MUSL 0 0 %C00
**SELECTINPUT 4
ENDINPUT 4
ED DOC011/MU6S
(FD/::MU6 /)
E
FLIP 0 1
DOC01
**DI 4 0
MUSL 0 0 %800
**SELECTINPUT 4
ENDINPUT 4
MERGEDIR VADIR? DOC3? VAXDIR?
STOP
::END COMPDOC
```

4.2. Private Library Compile Job for VAX.

This compiles all the Documentation software (TEXT,SPELL,LIST.INDEX, LIST.DICT,LIST.MOD,FLIP,DRAW,DISPLAY,SCROLL,LEVEL,FLED,XED,VED) into a private library (DOCV) in the directory of the calling process.

```
:BEGIN COMPDOCP
DEFINEOUTPUT 0 DOCPLOG %200
ED DOC021/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC02
**DI 4 0
MUSL 0 DOCV %405 16
*TLSEG 0 %0 %1C0000 %200000 0;
*TLSEG 1 %0  0 %F(7)D 0;
```

```
*TLSEG 2 %0 %3B0000 %F(7)D 0;
*TLSEG 3 %0 %3B0000 %F(7)D 0;
*TLSEG 4 %0 %3B0000 %F(7)D 0;
*TLLOAD 1 5;
*TLLOAD 2 3;
*TLLOAD 3 2;
*TLLOAD 4 6;
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC031/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC03
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC041/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC04
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC051/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC05
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC061/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC06
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC071/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC07
```

```
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC081/MU6S
(FD/::MU6 /)
E
FLIP 0 1
DOC08
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC091/MU6S
(FD/::MU6 /)
E
FLIP 0 1
DOC09
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED EDT021/MU6S
(FD/::MU6 /)
E
FLIP 0 1
EDT02
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC011/MU6S
(FD/::MU6 /)
E
FLIP 0 1
DOC01
**DI 4 0
MUSL 0 0 %800
**SELECTINPUT 4
ENDINPUT 4
STOP
::END COMPDOCP
```

4.3. Compile Job for PDP11(on VAX).

This job compiles TEXT ,SPELL,LIST.INDEX, LIST.DICT and LIST.MOD for downloading to a PDP11. It produces a code file PDP16 and a monitoring file TEXTPDPLOG.

```
::BEGIN COMPTEXTPDP
DO 0 TEXTPDPLOG %200
```

```
LIB MUSLJ/MUSM
LIB MUTLJ/MUSM
CREATESEGMENT 57 %10000
CREATESEGMENT 58 %10000
CREATESEGMENT 59 %10000
LIB MURDJ/MUSM
LIB FINDN11/MUSM
CREATESEGMENT 20 %10000
LIBRARY PDPDIR/MUSM
ED DOC011/MU6S
(FD/::PDP /)
E
FLIP 0 1
DOC01
DI 4 0
TL 4
TLDIRECTORY 5
TLSEG %6000 %2000 32
TLSEG %100 %1FA0 -1
TLSEG %6000 %8000 -2
TLSEG %6000 %8000 -2
TLLOAD 1 5
TLLOAD 2 2
TLLOAD 3 6
MUSL
*INIT 5;
**SELECTINPUT 4
ENDINPUT 4
CHANGESIZE 32 %6000
FILE PDP16 0 32
COPYFILE 0 STRO*
STOP
::END COMPTEXTPDP
```

4.4. Private Library Compile of FLIP for VAX.

This compiles FLIP only into a private library (DOCFLIP) in the directory of the calling process.

```
:BEGIN COMPFLIP3
DEFINEOUTPUT 0 FLIP3LOG %200
ED DOC021/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC02
**DI 4 0
MUSL 0 DOCFLIP3 %405 16
*TLSEG 0 %0 %1C0000 %200000 0;
*TLSEG 1 %0  0 %F(7)D 0;
*TLSEG 2 %0 %3B0000 %F(7)D 0;
*TLLOAD 1 5;
*TLLOAD 2 3;
*INFORM %2400;
**SELECTINPUT 4
```

```
ENDINPUT 4
ED DOC031/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC03
**DI 4 0
MUSL 0 0 %800
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
STOP
::END COMPFLIP3
```

4.5. Cross-compile of FLIP for MC68000 on VAX.

This compiles FLIP,INPUT.TITLE and INPUT.CHART for MC68000 in a file called FLIP5 producing a log-file FLIP5LOG.

```
::BEGIN COMPFLIP5 FOR MC68000
DEFINEOUTPUT 0 FLIP5LOG %200
LIB MUTLX5/MUSM
LIB MUSLX/MUSM
LIB LIB02X5/MUSM
LIB VLIBDIR5/MUSM
ED DOC021/MU6S
(FD/::MU6 /)
E
FLIP 0 1
DOC02
**DI 4 0
MUSL 0 FLOP5 %705 16
**TLSEG 0 %0 %00140000 %F(8) 0
**TLSEG 1 %0 0 %F(7)D 0
**TLSEG 2 %0 %C0000 %F(7)D 0
**TLLOAD 1 5
**TLLOAD 2 3
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC031/MU6S
(FD?::MC68000?)
E
FLIP 0 1
DOC03
**DI 4 0
MUSL 0 0 %B00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
TRANDIR FLOP5 FLIP5 0 %4000
DEL FLOP5
DEL FLOP500
STOP
```

```
::END COMPFLIP5
```

4.6. Cross–compile of TEXT for MC68000 on VAX.

This compiles TEXT,SPELL,LIST.DICT and LIST.INDEX for MC68000 in a file called TEXT5 producing a log–file TEXT5LOG.

```
::BEGIN COMPTEXT5 FOR MC68000
DEFINEOUTPUT 0 TEXT5LOG %200
LIB MUTLX5/MUSM
LIB MUSLX/MUSM
LIB LIB02X5/MUSM
LIB VLIBDIR5/MUSM
ED DOC011/MU6S
(FD/::MC68000 /)
E
FLIP 0 1
DOC01
**DI 4 0
MUSL 0 TOXT5 %305 16
**TLSEG 0 %0 %00144000 %F(8) 0
**TLSEG 1 %0  0 %F(7)D 0
**TLSEG 2 %0 %C0000 %F(7)D 0
**TLSEG 3 %0 %C0000 %F(7)D 0
**TLLOAD 1 5
**TLLOAD 2 2
**TLLOAD 3 6
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
TRANDIR TOXT5 TEXT5 0 %4000
DEL TOXT5
DEL TOXT500
STOP
::END COMPTEXT5
```

4.7. Cross–compile of DRAW for MC68000 on VAX.

This compiles DRAW(for LPT and PLT only) and FLED for MC68000 in a file called DRAW5 producing a log–file DRAW5LOG.

```
::BEGIN COMPDRAW5 FOR MC68000
DEFINEOUTPUT 0 DRAW5LOG %200
LIB MUTLX5/MUSM
LIB MUSLX/MUSM
LIB LIB02X5/MUSM
LIB VLIBDIR5/MUSM
LIB EDT25/MUSM
ED DOC021/MU6S
(FD/::MU6 /)
E
FLIP 0 1
DOC02
```

```
**DI 4 0
MUSL 0 DRAW5 %705 16
**TLSEG 0 %0 %00118000 %F(8) 0
**TLSEG 1 %0  0 %F(7)D 0
**TLSEG 2 %0 %C0000 %F(7)D 0
**TLSEG 3 %0 %C0000 %F(7)D 0
**TLSEG 4 %0 %C0000 %F(7)D 0
**TLLOAD 1 5
**TLLOAD 2 3
**TLLOAD 3 2
**TLLOAD 4 6
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC041/MU6S
(FD?::MC68000 ?)
E
FLIP 0 1
DOC04
**DI 4 0
MUSL 0 0 %F00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC051/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC05
**DI 4 0
MUSL 0 0 %F00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC071/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC07
**DI 4 0
MUSL 0 0 %F00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC081/MU6S
(FD?::MC68000 ?)
E
ED
C/PS VEDIT/D/PS/I/LS/
E
FLIP 0 1
DOC08
**DI 4 0
MUSL 0 0 %800
*INFORM %2400;
```

```
**SELECTINPUT 4
ENDINPUT 4
TRANDIR DRAW5 DRAW5 0 %4000
STOP
::END COMPDRAW5 FOR MC68000
```

4.8. Private Library Compile Job for Interactive Draw on VAX.

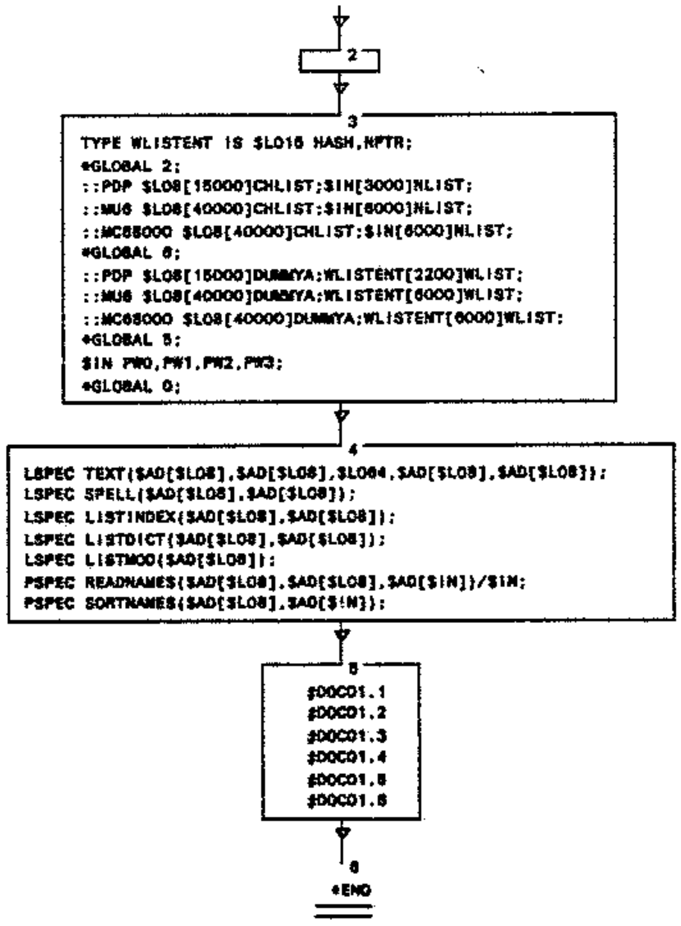
This compiles all the Interactive Documentation software (DISPLAY,SCROLL,LEVEL,FLED,XEDIT,VEDIT) into a private library (IDRAWP) in the directory of the calling process.

```
:BEGIN COMPIDRAWP
DEFINEOUTPUT 0 IDRAWPLOG %200
ED DOC021/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC02
**DI 4 0
MUSL 0 IDRAWP %405 16
*TLSEG 0 %0 %1C0000 %200000 0;
*TLSEG 1 %0  0 %F(7)D 0;
*TLSEG 2 %0 %3B0000 %F(7)D 0;
*TLSEG 3 %0 %3B0000 %F(7)D 0;
*TLSEG 4 %0 %3B0000 %F(7)D 0;
*TLLOAD 1 5;
*TLLOAD 2 3;
*TLLOAD 3 2;
*TLLOAD 4 6;
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC041/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC04
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC071/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
DOC07
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED DOC081/MU6S
```

```
(FD/::MU6 /)
E
FLIP 0 1
DOC08
**DI 4 0
MUSL 0 0 %C00
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
ED EDT021/MU6S
(FD?::MU6 ?)
E
FLIP 0 1
EDT02
**DI 4 0
MUSL 0 0 %800
*INFORM %2400;
**SELECTINPUT 4
ENDINPUT 4
STOP
::END COMPIDRAWP
```

FLOWCHARTS
DOC011

```
                              1
:: MU6 IMPLEMENTATION;
$PS DRAW($AD[$LO8].$AD[$LO8].$IN.$LO64);
$LS CURRENT.INPUT()/$IN; $LS CURRENT.OUTPUT()/$IN;
$LS END.INPUT($IN,$IN);$LS END.OUTPUT($IN,$IN);
$LS DEFINE.INPUT($IN,$AD[$LO8],$IN,$IN)/$IN;
$LS DEFINE.OUTPUT($IN,$AD[$LO8],$IN,$IN,$IN,$IN)/$IN;
$LS SELECT.INPUT($IN);
$LS SELECT.OUTPUT($IN);
$LS INCH()/$IN;
$LS NEXTCH()/$IN;
$LS INI()/ADDR;
$LS INBACKSPACE($IN);
$LS OUTCH($IN);
$LS NEWLINES($IN);
$LS SPACES($IN);
$LS OUTI($IN.$IN);
$LS CAPTION($AD[$LO8]);
$LS PPCCMD();
$LS BREAK.OUTPUT($IN);
$LS I.POS()/$IN32;
$LS I.BPOS()/$IN32;
$LS SET.I.BPOS($IN32,$IN32);
$LS ISIZE()/$IN;
$LS TIME.AND.DATE();
$LS IENQ()/$IN;
$LS OUTLINENO($IN32);
$LS INSTR($AD[$LO8])/$IN;
$LS MAP($IN,$IN,$IN);
$LS CREATE.SEGMENT($IN,$AD);
$LS RELEASE.SEGMENT($IN);
MODULE(TEXT,SPELL,LISTINDEX,LISTDICT,LISTMOD);
```



```
2

3
TYPE WLISTENT IS $LO16 HASH,NPTR;
*GLOBAL 2;
::PDP $LO8[15000]CHLIST;$IN[3000]NLIST;
::MU6 $LO8[40000]CHLIST;$IN[6000]NLIST;
::MC68000 $LO8[40000]CHLIST;$IN[6000]NLIST;
*GLOBAL 6;
::PDP $LO8[15000]DUMMYA;WLISTENT[2200]WLIST;
::MU6 $LO8[40000]DUMMYA;WLISTENT[6000]WLIST;
::MC68000 $LO8[40000]DUMMYA;WLISTENT[6000]WLIST;
*GLOBAL 5;
$IN PW0,PW1,PW2,PW3;
*GLOBAL 0;

4
LSPEC TEXT($AD[$LO8],$AD[$LO8],$LO64,$AD[$LO8],$AD[$LO8]);
LSPEC SPELL($AD[$LO8],$AD[$LO8]);
LSPEC LISTINDEX($AD[$LO8],$AD[$LO8]);
LSPEC LISTDICT($AD[$LO8],$AD[$LO8]);
LSPEC LISTMOD($AD[$LO8]);
PSPEC READNAMES($AD[$LO8],$AD[$LO8],$AD[$IN])/$IN;
PSPEC SORTNAMES($AD[$LO8],$AD[$IN]);

5
$DOC01.1
$DOC01.2
$DOC01.3
$DOC01.4
$DOC01.5
$DOC01.6

6
*END
```

Figure 1 Y.DOC01110 6 August 1982

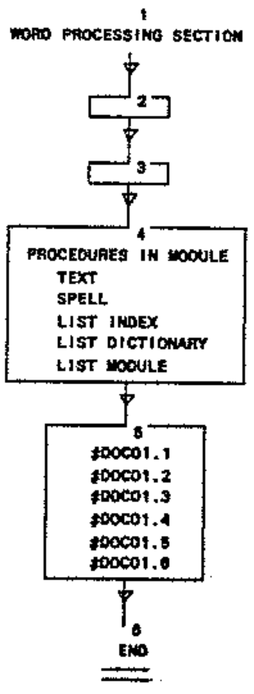
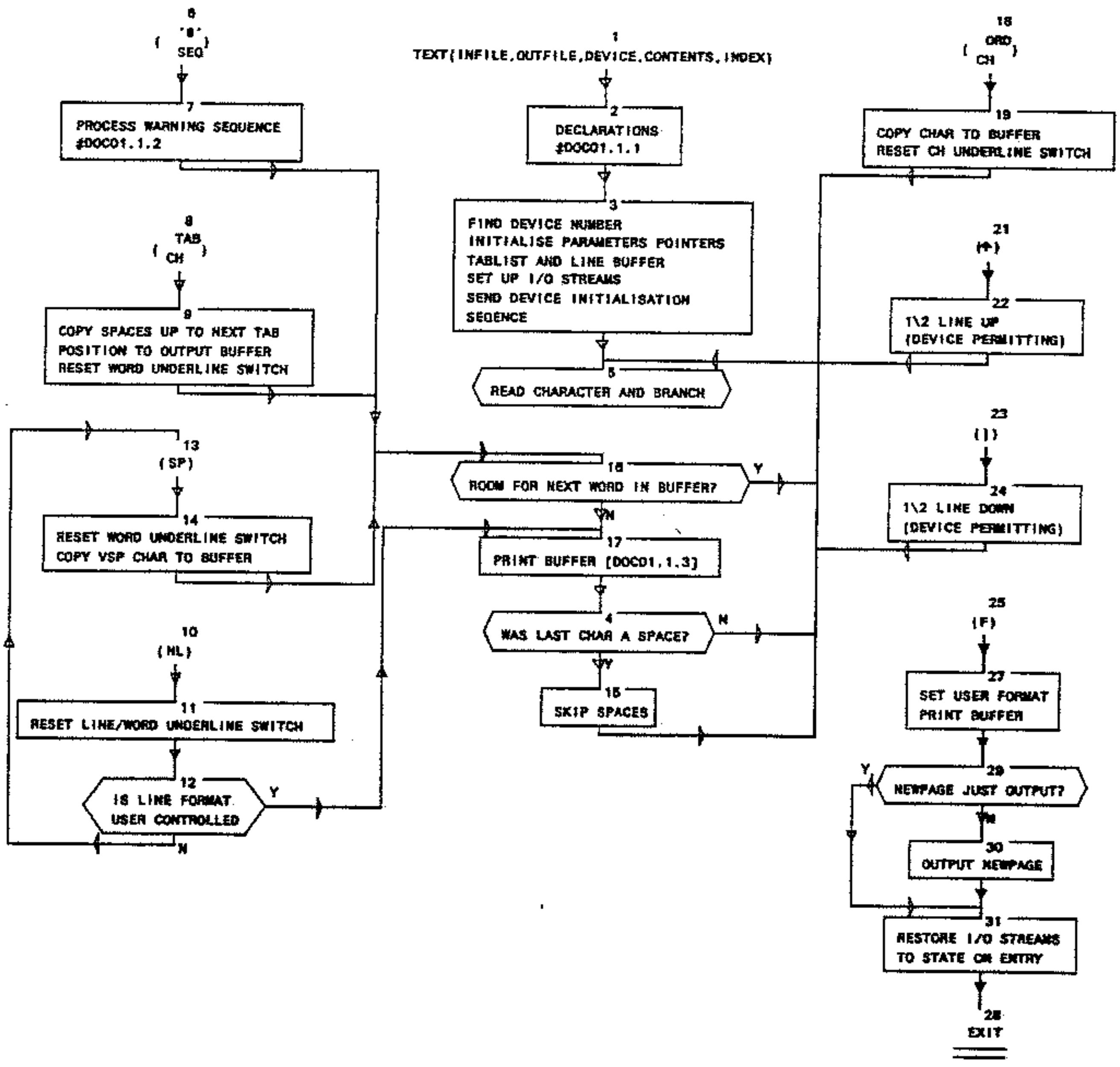
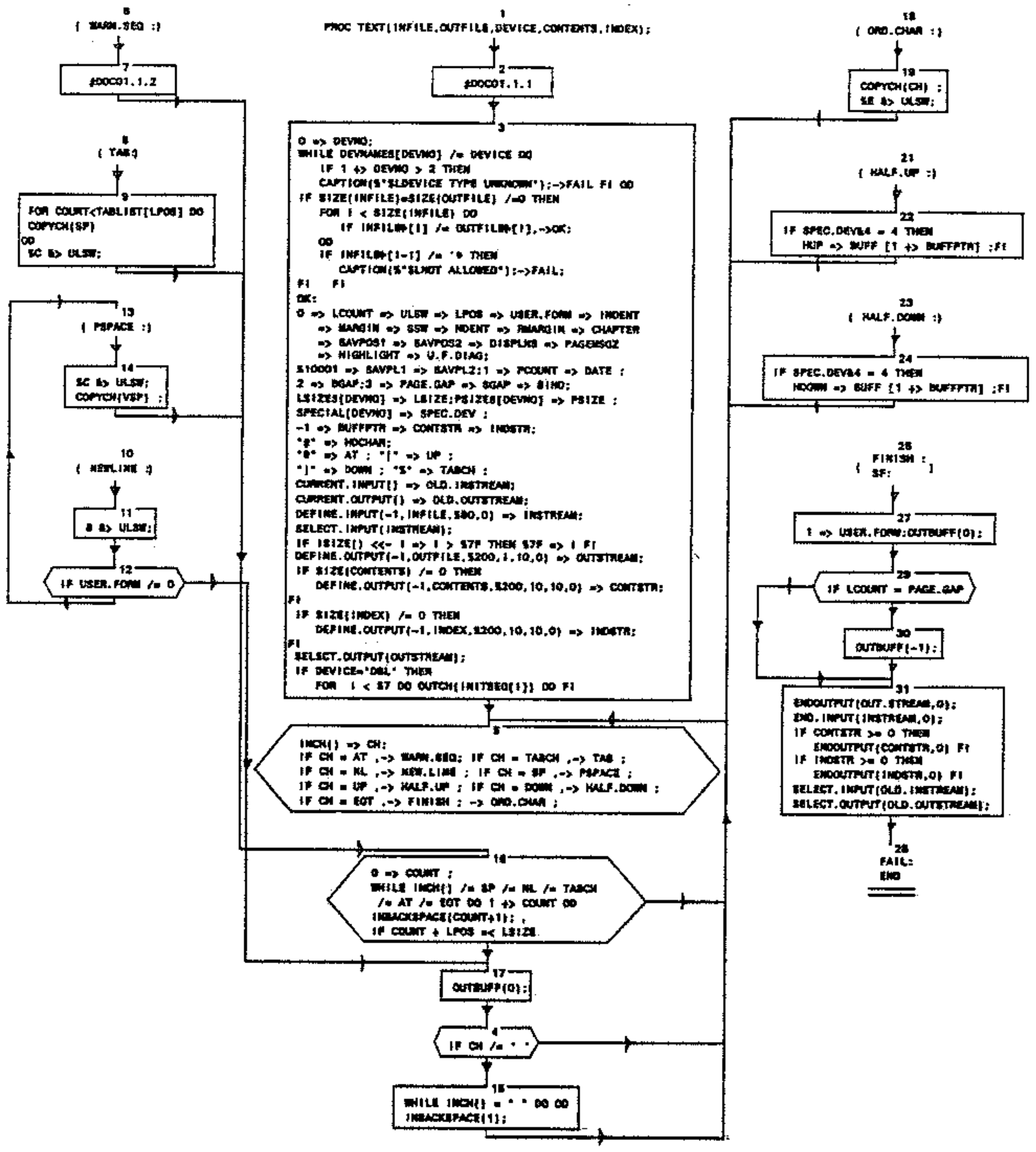


Figure 1 Y.DOC01100 8 August 1982



MANCHESTER UNIVERSITY - CONFIDENTIAL

Figure 2 Y.DOCO1100 6 August 1982



```
1
PROC TEXT(INFILE,OUTFILE,DEVICE,CONTENTS,INDEX);

2
#DOC01.1.1

3
0 => DEVNO;
WHILE DEVNAMES[DEVNO] /= DEVICE DO
   IF 1 +> DEVNO > 2 THEN
   CAPTION(%"$LDEVICE TYPE UNKNOWN");->FAIL FI OD
IF SIZE(INFILE)=SIZE(OUTFILE) /=0 THEN
   FOR I < SIZE(INFILE) DO
      IF INFILE^[I] /= OUTFILE^[I],->OK;
   OD
   IF INFILE^[I-1] /= '9 THEN
      CAPTION(%"$LNOT ALLOWED");->FAIL;
FI   FI
OK:
0 => LCOUNT => ULSW => LPOS => USER.FORM => INDENT
   => MARGIN => SSW => NDENT => RMARGIN => CHAPTER
   => SAVPOS1 => SAVPOS2 => DISPLNS => PAGEMSGZ
   => HIGHLIGHT => U.F.DIAG;
%10001 => SAVPL1 => SAVPL2;1 => PCOUNT => DATE ;
2 => BGAP;3 => PAGE.GAP => SGAP => SIND;
LSIZES[DEVNO] => LSIZE;PSIZES[DEVNO] => PSIZE ;
SPECIAL[DEVNO] => SPEC.DEV ;
-1 => BUFFPTR => CONTSTR => INDSTR;
"#" => HDCHAR;
"@" => AT ; "[" => UP ;
"]" => DOWN ; "$" => TABCH ;
CURRENT.INPUT() => OLD.INSTREAM;
CURRENT.OUTPUT() => OLD.OUTSTREAM;
DEFINE.INPUT(-1,INFILE,%80,0) => INSTREAM;
SELECT.INPUT(INSTREAM);
IF ISIZE() <<- I => I > %7F THEN %7F => I FI
DEFINE.OUTPUT(-1,OUTFILE,%200,I,10,0) => OUTSTREAM;
IF SIZE(CONTENTS) /= 0 THEN
   DEFINE.OUTPUT(-1,CONTENTS,%200,10,10,0) => CONTSTR;
FI
IF SIZE(INDEX) /= 0 THEN
   DEFINE.OUTPUT(-1,INDEX,%200,10,10,0) => INDSTR;
FI
SELSCT.OUTPUT(OUTSTREAM);
IF DEVICE='DBL' THEN
   FOR  I < %7 DO OUTCH(INITSEQ(I)) OD FI

5
INCH() => CH;
IF CH = AT ,-> WARN.SEQ; IF CH = TABCH ,-> TAB ;
IF CH = NL ,-> NEW.LINE ; IF CH = SP ,-> PSPACE ;
IF CH = UP ,-> HALF.UP ; IF CH = DOWN ,-> HALF.DOWN ;
IF CH = EOT ,-> FINISH ; -> ORD.CHAR ;

6
( WARN.SEQ :)

7
#DOC01.1.2

8
( TAB:)

9
FOR COUNT<TABLIST[LPOS] DO
COPYCH(SP)
OD
%C &> ULSW;

13
( PSPACE :)

14
%C &> ULSW;
COPYCH(VSP) ;

10
( NEWLINE :)

11
8 &> ULSW;

12
IF USER.FORM /= 0

16
0 => COUNT ;
WHILE INCH() /= SP /= NL /= TABCH
 /= AT /= EOT DO 1 +> COUNT OD
INBACKSPACE(COUNT+1); ,
IF COUNT + LPOS =< LSIZE

17
OUTBUFF(0);

4
IF CH /= " "

15
WHILE INCH() = " " DO OD
INBACKSPACE(1);

18
( ORD.CHAR :)

19
COPYCH(CH) ;
%E &> ULSW;

21
( HALF.UP :)

22
IF SPEC.DEV&4 = 4 THEN
   HUP => BUFF [1 +> BUFFPTR] ;FI

23
( HALF.DOWN :)

24
IF SPEC.DEV&4 = 4 THEN
   HDOWN => BUFF [1 +> BUFFPTR] ;FI

26
( FINISH :
  SF:        )

27
1 => USER.FORM:OUTBUFF(0);

29
IF LCOUNT = PAGE.GAP

30
OUTBUFF(-1);

31
ENDOUTPUT(OUT.STREAM,0);
END.INPUT(INSTREAM,0);
IF CONTSTR >= 0 THEN
   ENDOUTPUT(CONTSTR,0) FI
IF INDSTR >= 0 THEN
   ENDOUTPUT(INDSTR,0) FI
SELECT.INPUT(OLD.INSTREAM);
SELECT.OUTPUT(OLD.OUTSTREAM);

28
FAIL:
END
```

MANCHESTER UNIVERSITY - CONFIDENTIAL

Figure 2 Y.DOC01110 6 August 1982

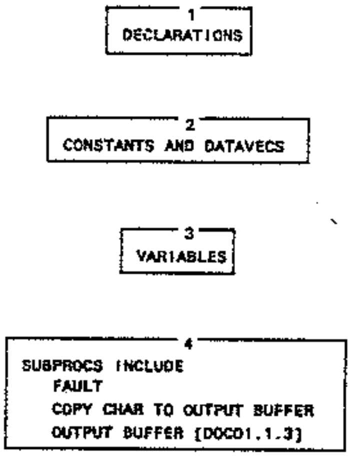


Figure 3 Y.DOC01100 6 August 1982

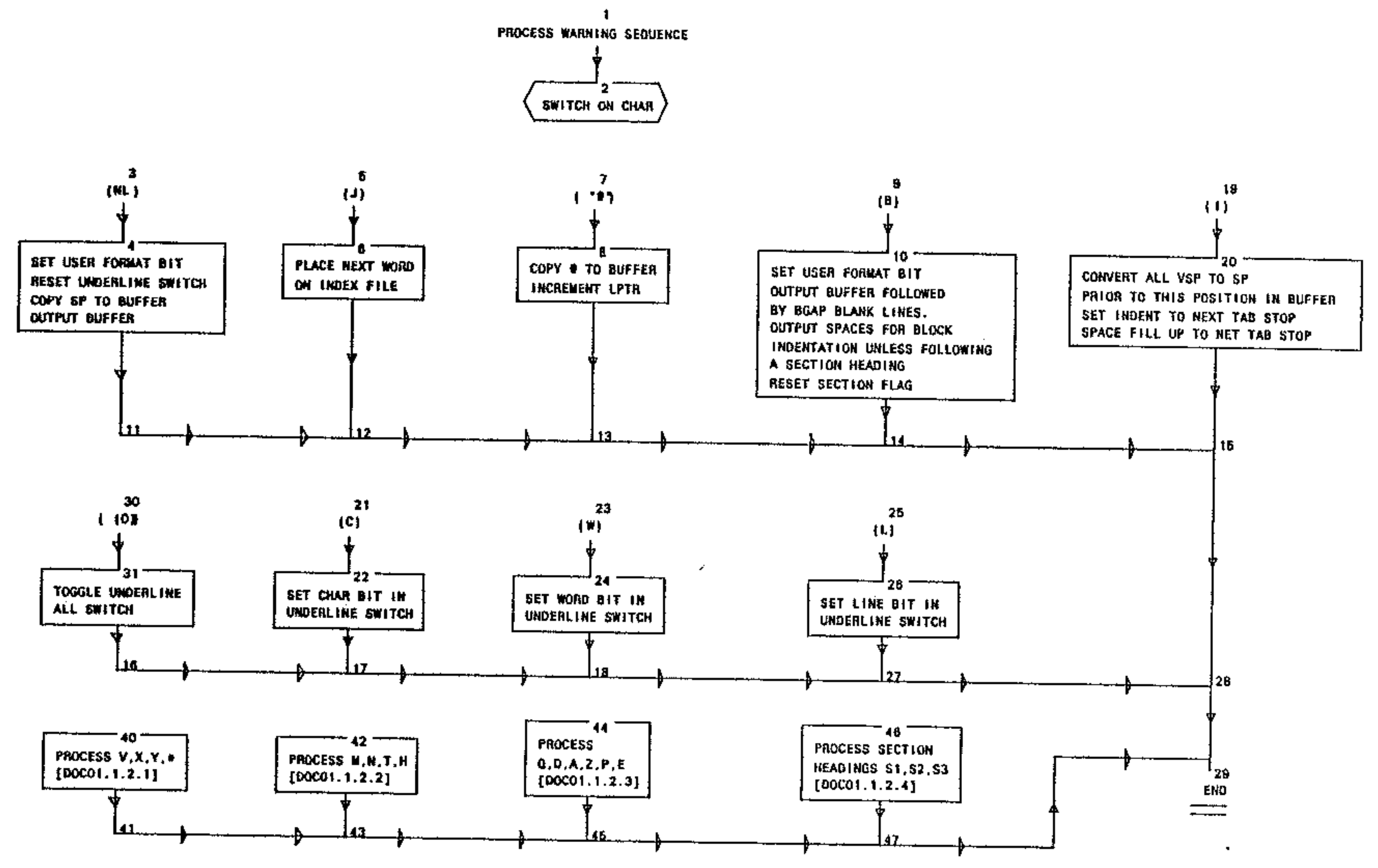
```
1
::DECLARATIONS
```

```
2
LITERAL/LOGICAL8 SP=$20,NL=$A,EOT=4,BS=8,
HUP=$80,HDOWN=$81,VSP=$C0,UL=$5F,HTAB=8 ;
DATAVEC DEVNAMES(LOGICAL32)
"DBL" "LPT" 0 END
DATAVEC LSIZES(INTEGER)
70  70   70  END
DATAVEC PSIZES(INTEGER)
60  60  80  END
DATAVEC SPECIAL(LOGICAL8)
7 0 0 END
DATAVEC INITSEQ(LOGICAL8);
$98 $82 $98 $9F $88 $98 $9E $88
$98 $88 $8F $98 $88 $98 $88 $89
$98 $81 $98 $88 $A3 $98 $81 $98
$88 $A0 $98 $81 $98 $88 $87 $98
$81 $98 $88 $C1 $98 $81 $98 $89
$C8 $98 $81 $98 $88 $D6 $98 $81
$98 $88 $DF $98 $81 $98 $C2 $80 $83
END
DATAVEC HDFORM(LOGICAL8);
8 $78 $8C $94 $A4 $C4 $78
6 $20 $60 $A0 $20 $20 $F0
8 $78 $84 $8 $70 $80 $FC
8 $FC $8 $38 $4 $84 $78
8 $18 $28 $48 $88 $FC $8
8 $FC $80 $F8 $4 $84 $78
8 $78 $84 $80 $F8 $84 $78
8 $FC $4 $8 $10 $20 $40
8 $78 $84 $78 $84 $84 $78
8 $78 $84 $7C $4 $4 $4
8 $30 $48 $84 $FC $84 $84
8 $F8 $84 $F8 $84 $84 $F8
8 $78 $84 $80 $80 $84 $78
8 $F8 $84 $84 $84 $84 $F8
8 $FC $80 $F0 $80 $80 $FC
8 $FC $80 $F0 $80 $80 $80
8 $78 $84 $80 $9C $84 $78
8 $84 $84 $FC $84 $84 $84
7 $F8 $20 $20 $20 $20 $F8
8 $FC $10 $10 $10 $90 $60
8 $84 $88 $F0 $90 $88 $84
7 $80 $80 $80 $80 $80 $F8
8 $84 $CC $B4 $84 $84 $84
8 $84 $C4 $A4 $94 $8C $84
8 $78 $84 $84 $84 $84 $78
8 $F8 $84 $F8 $80 $80 $80
8 $70 $88 $88 $A8 $98 $7C
8 $F8 $84 $F8 $90 $88 $84
8 $78 $80 $78 $4 $84 $78
7 $F8 $20 $20 $20 $20 $20
8 $84 $84 $84 $84 $84 $78
8 $84 $84 $84 $84 $48 $30
8 $84 $84 $84 $84 $CC $84
8 $84 $48 $30 $30 $48 $84
7 $88 $50 $20 $20 $20 $20
8 $FC $8 $10 $20 $40 $FC
END
```

```
3
LOGICAL8[380] BUFF,TABLIST ;
LOGICAL8[32] PAGEMSG ;
LOGICAL8[128] CHAPSTR;
INTEGER BUFFPTR,LPOS,DEVNO,ULSW,INSTREAM,OUTSTREAM,CONTSTR,CNT,
OLD.INSTREAM,OLD.OUTSTREAM,PAGE,GAP,BGAP,SGAP,BIND,INDSTR,HIGHLIGHT,
MARGIN,LSIZE,PSIZE,PCOUNT,LCOUNT,CH,SSW,DATE,INDENT,CHAPTER,SAVIND,
I,J,K,COUNT,NDENT,RMARGIN,DISPLNS,USER,FORM,PAGEMSGZ,U.F.DIAG,TEMP;
INTEGER32 SAVPL1,SAVPL2,SAVPL3,SAVPOS1,SAVPOS2,SAVPOS3;
LOGICAL SPEC.DEV,AT,UP.DOWN,TABCH,NOCHAR ;
```

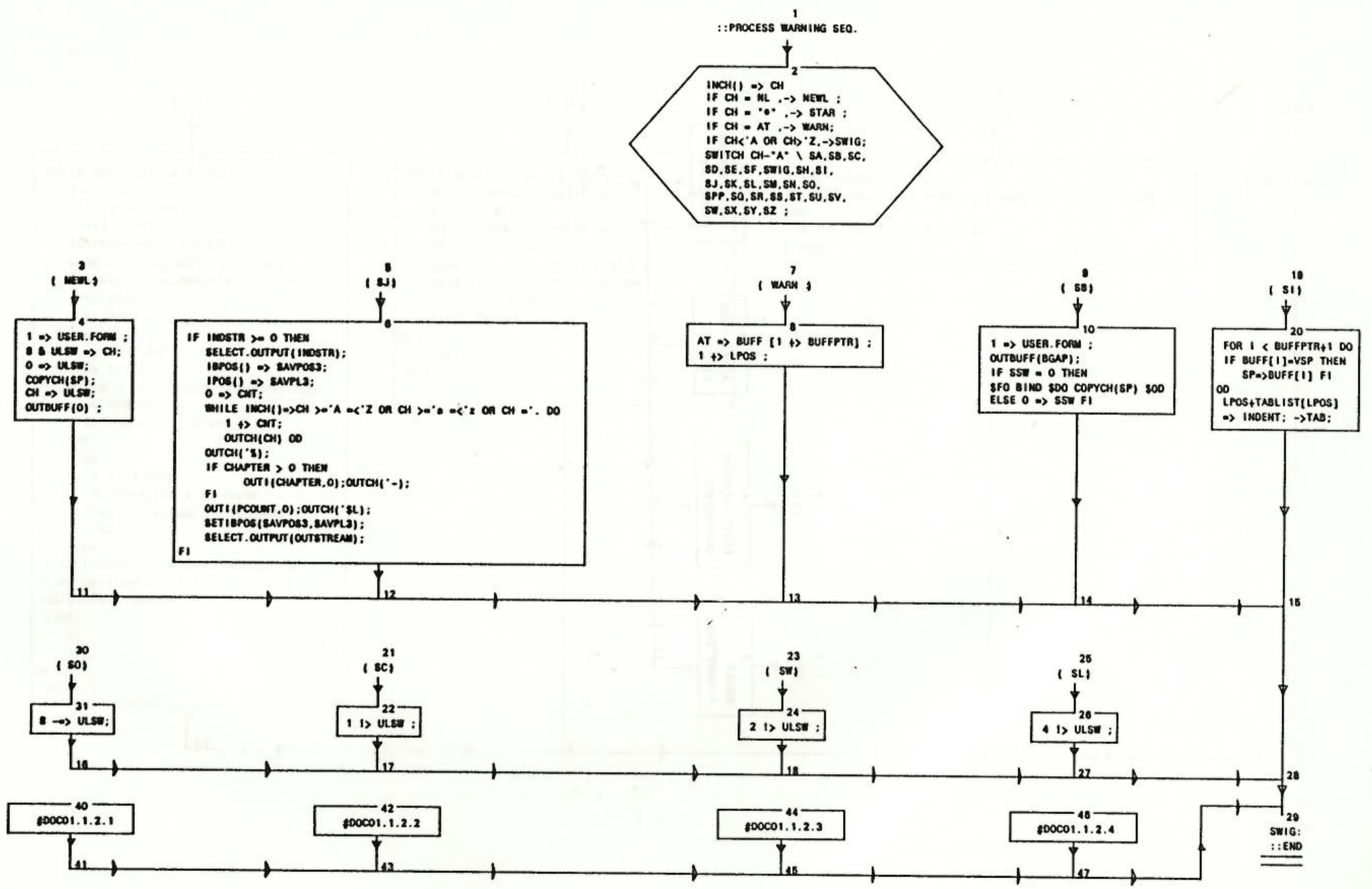
```
4
PSPEC FAULT(INTEGER);
PSPEC COPYCH(LOGICAL8);
PSPEC OUTBUFF(INTEGER);

PROC FAULT(N) ;
NEWLINES(0)
FOR 8 DO OUTCH("*") OD
OUTI(N,5) ;
END

PROC COPYCH(CHAR) ;
   CHAR => BUFF [1 +> BUFFPTR] ;
   1 +> LPOS
IF ULSW /= 0 THEN
   IF CHAR = VSP THEN
      SP => BUFF [BUFFPTR]
   FI
   IF SPEC.DEVS2 /= 0 THEN
   BS => BUFF [1 +> BUFFPTR] ;
   UL => BUFF [1 +> BUFFPTR] ;
FI FI
END
#DOC01.1.3
```

Figure 3 Y.DOC01110 6 August 1982



14/06/82

MANCHESTER UNIVERSITY - CONFIDENTIAL



```
INCH() => CH
IF CH = NL ,-> NEWL ;
IF CH = "*" ,-> STAR ;
IF CH = AT ,-> WARN;
IF CH<'A OR CH>'Z,->SWIG;
SWITCH CH-"A" \ SA,SB,SC,
SD,SE,SF,SWIG,SH,SI,
SJ,SK,SL,SM,SN,SO,
SPP,SQ,SR,SS,ST,SU,SV,
SW,SX,SY,SZ ;
```

```
1 => USER.FORM ;
8 & ULSW => CH;
0 => ULSW;
COPYCH($P);
CH => ULSW;
OUTBUFF(0) ;
```

```
IF INDSTR >= 0 THEN
   SELECT.OUTPUT(INDSTR);
   IBPOS() => SAVPOS3;
   IPOS() => SAVPL3;
   0 => CNT;
   WHILE INCH()=>CH >='A =<'Z OR CH >='a =<'z OR CH ='. DO
      1 +> CNT;
      OUTCH(CH) OD
   OUTCH('$);
   IF CHAPTER > 0 THEN
         OUTI(CHAPTER,0);OUTCH('-);
   FI
   OUTI(PCOUNT,0);OUTCH('$L);
   SETIBPOS(SAVPOS3,SAVPL3);
   SELECT.OUTPUT(OUTSTREAM);
FI
```

```
AT => BUFF [1 +> BUFFPTR] ;
1 +> LPOS ;
```

```
1 => USER.FORM ;
OUTBUFF(BGAP);
IF SSW = 0 THEN
$FO BIND $DO COPYCH(SP) $OD
ELSE 0 => SSW FI
```

```
FOR I < BUFFPTR+1 DO
IF BUFF[I]=VSP THEN
   SP=>BUFF[I] FI
OD
LPOS+TABLIST[LPOS]
=> INDENT; ->TAB;
```

```
8 -=> ULSW;
```

```
1 !> ULSW ;
```

```
2 !> ULSW ;
```

```
4 !> ULSW ;
```

Figure 4 Y.DOCO1110 6 August 1982

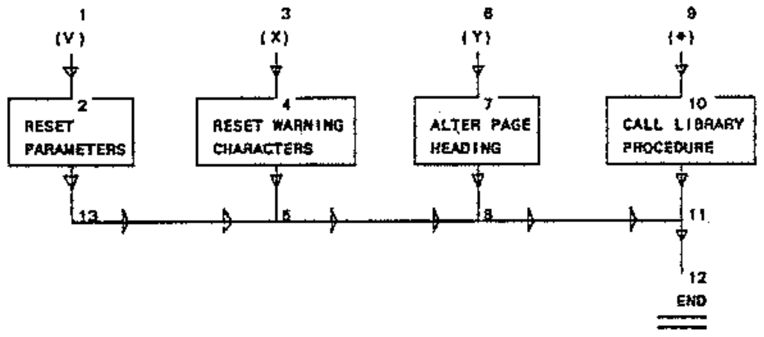
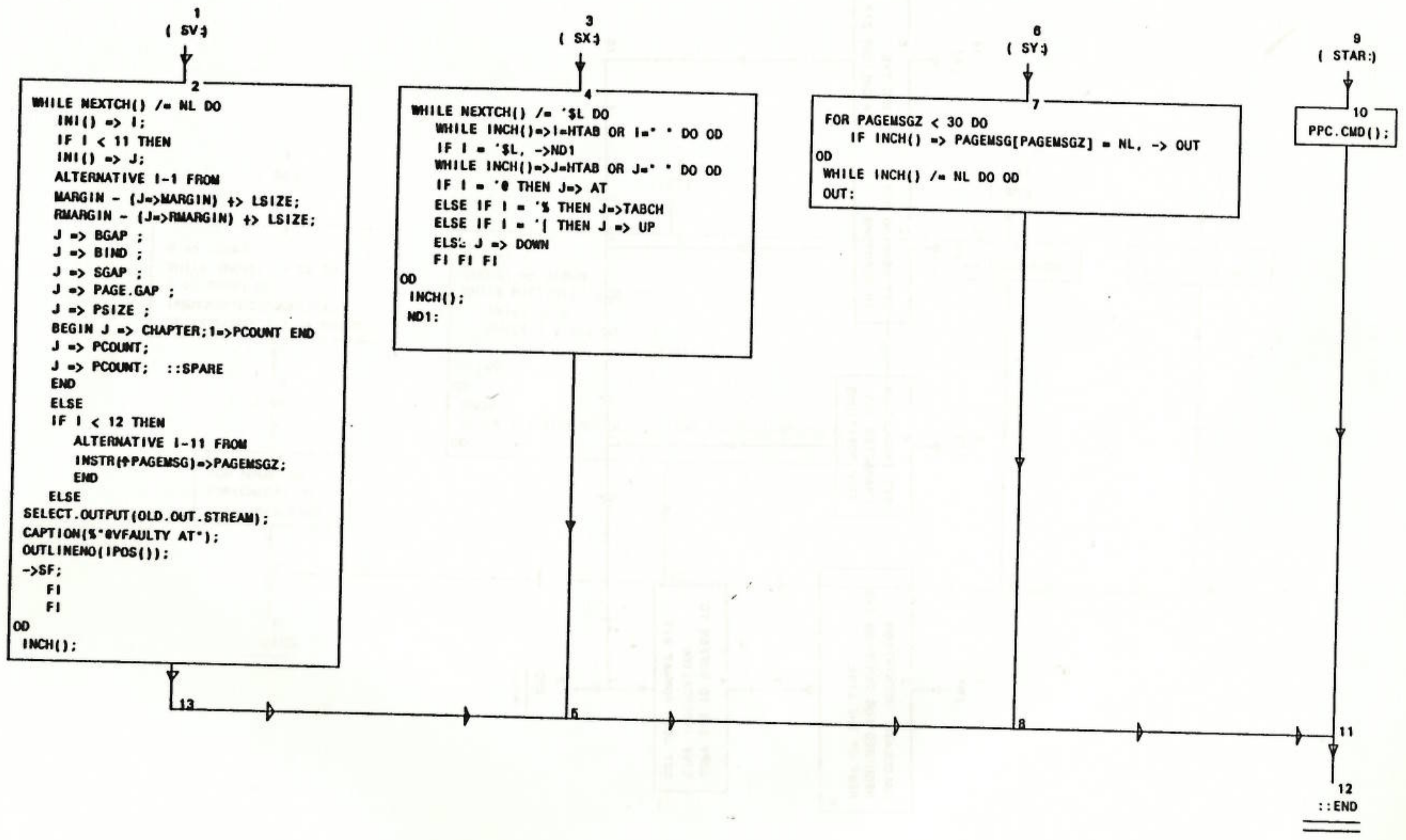


Figure 5 Y.DOC01100 6 August 1982

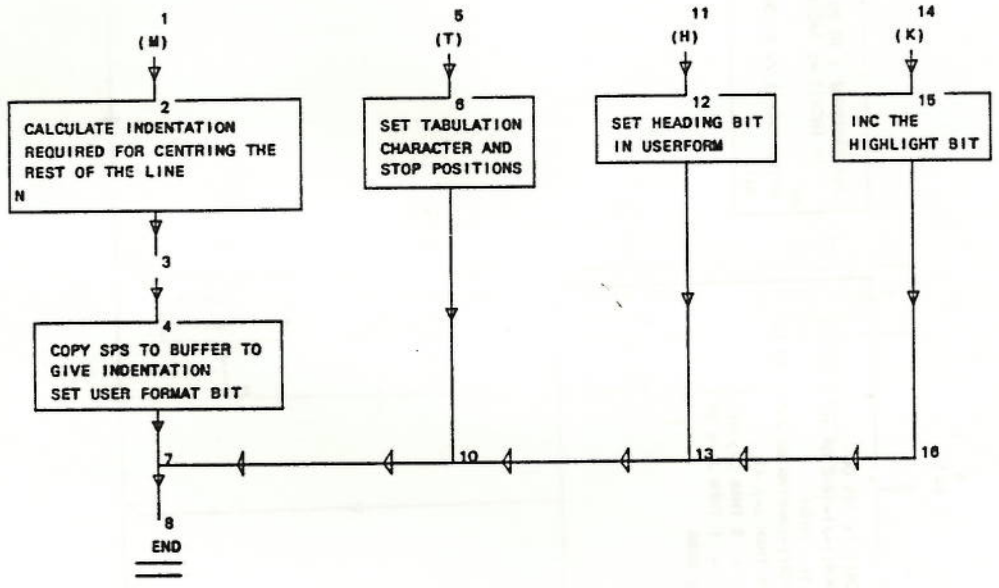


```
WHILE NEXTCH() /= NL DO
    INI() => I;
    IF I < 11 THEN
    INI() => J;
   ALTERNATIVE I-1 FROM
   MARGIN - (J=>MARGIN) +> LSIZE;
   RMARGIN - (J=>RMARGIN) +> LSIZE;
   J => BGAP ;
   J => BIND ;
   J => SGAP ;
   J => PAGE.GAP ;
   J => PSIZE ;
   BEGIN J => CHAPTER;1=>PCOUNT END
   J => PCOUNT;
   J => PCOUNT;  ::SPARE
   END
   ELSE
   IF I < 12 THEN
      ALTERNATIVE I-11 FROM
      INSTR(↑PAGEMSG)=>PAGEMSGZ;
      END
   ELSE
 SELECT.OUTPUT(OLD.OUT.STREAM);
 CAPTION(%"@VFAULTY AT");
 OUTLINENO(IPOS());
 ->SF;
   FI
   FI
OD
 INCH();
```

```
WHILE NEXTCH() /= '$L DO
   WHILE INCH()=>I=HTAB OR I=" " DO OD
   IF I = '$L, ->ND1
   WHILE INCH()=>J=HTAB OR J=" " DO OD
   IF I = '@ THEN J=> AT
   ELSE IF I = '% THEN J=>TABCH
   ELSE IF I = '[ THEN J => UP
   ELSE J => DOWN
   FI FI FI
OD
 INCH();
 ND1:
```

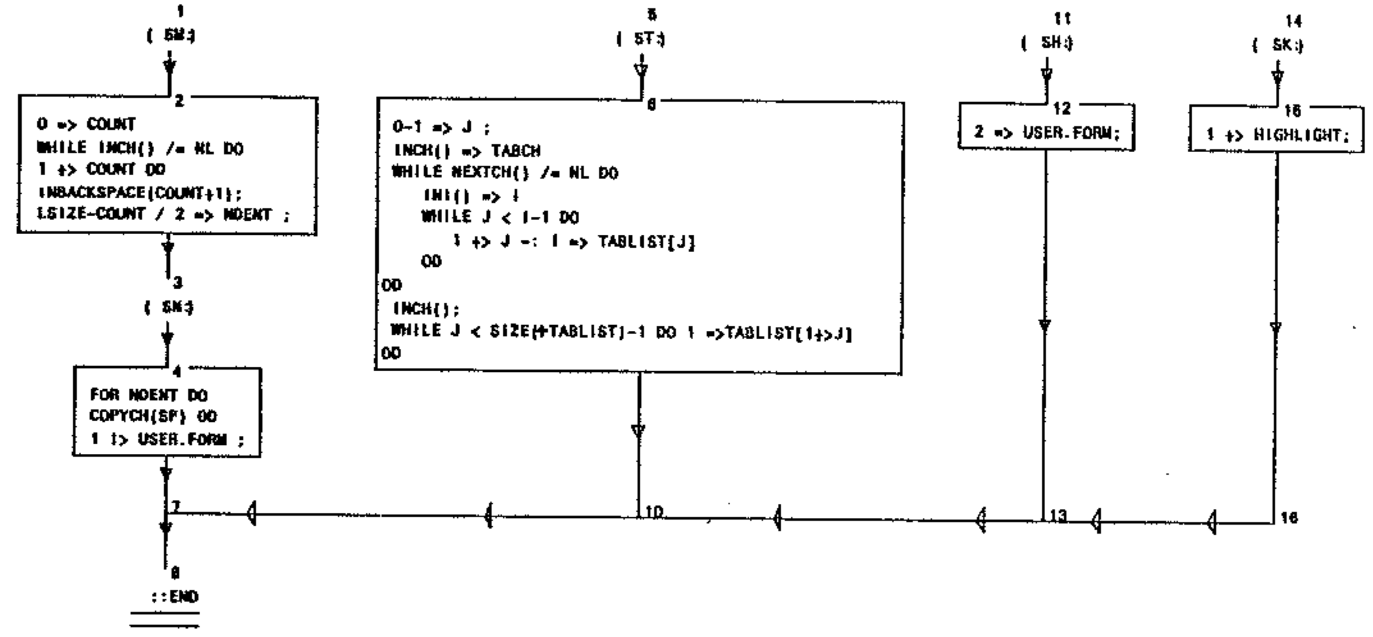
```
 FOR PAGEMSGZ < 30 DO
    IF INCH() => PAGEMSG[PAGEMSGZ] = NL, -> OUT
OD
 WHILE INCH() /= NL DO OD
 OUT:
```

Figure 5 Y.DOCO1110 6 August 1982



1
(M)
2
CALCULATE INDENTATION REQUIRED FOR CENTRING THE REST OF THE LINE
N
3
4
COPY SPS TO BUFFER TO GIVE INDENTATION
SET USER FORMAT BIT
5
(T)
6
SET TABULATION CHARACTER AND STOP POSITIONS
11
(H)
12
SET HEADING BIT IN USERFORM
14
(K)
15
INC THE HIGHLIGHT BIT
7
10
13
16
8
END

Figure 6 Y.DOCO1100 6 August 1982



```
0 => COUNT
WHILE INCH() /= NL DO
1 +> COUNT OD
INBACKSPACE(COUNT+1);
LSIZE-COUNT / 2 => NOENT ;
```

```
FOR NOENT DO
COPYCH(SP) OD
1 !> USER.FORM ;
```

```
 0-1 => J ;
 INCH() => TABCH
 WHILE NEXTCH() /= NL DO
    INI() => I
    WHILE J < I-1 DO
       1 +> J -: I => TABLIST[J]
    OD
OD
 INCH();
 WHILE J < SIZE(+TABLIST)-1 DO 1 =>TABLIST[1+>J]
OD
```

```
2 => USER.FORM;
```

```
1 +> HIGHLIGHT;
```

MANCHESTER UNIVERSITY - CONFIDENTIAL

Figure 6 Y.DOC01110 6 August 1982

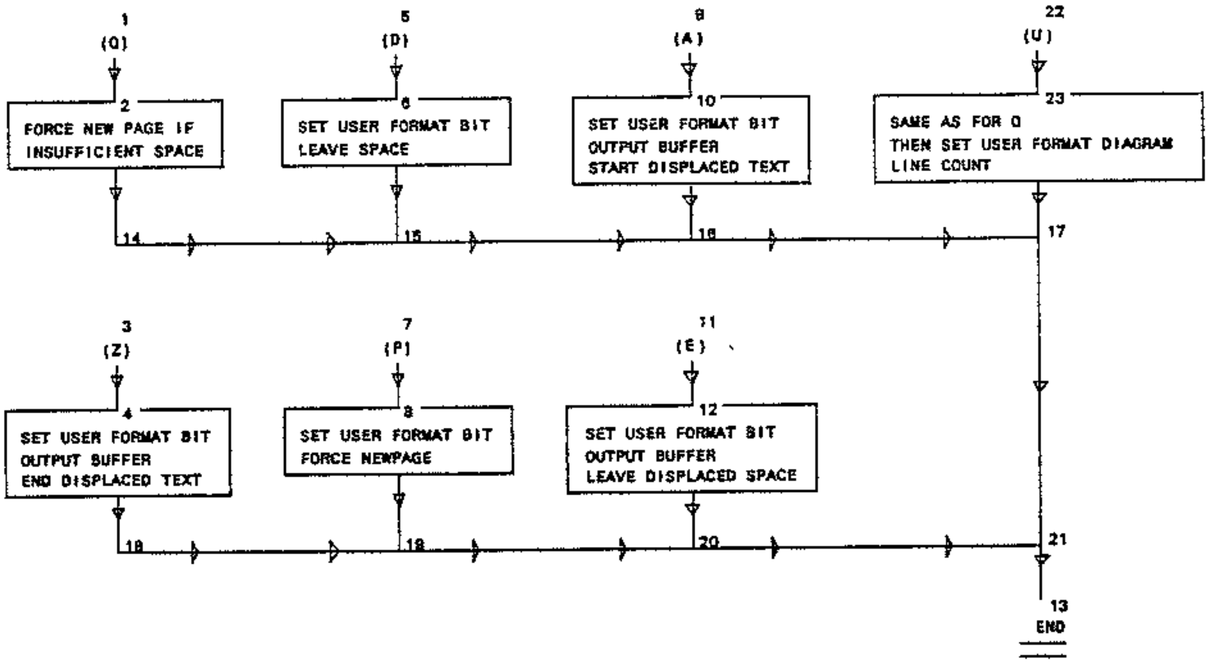
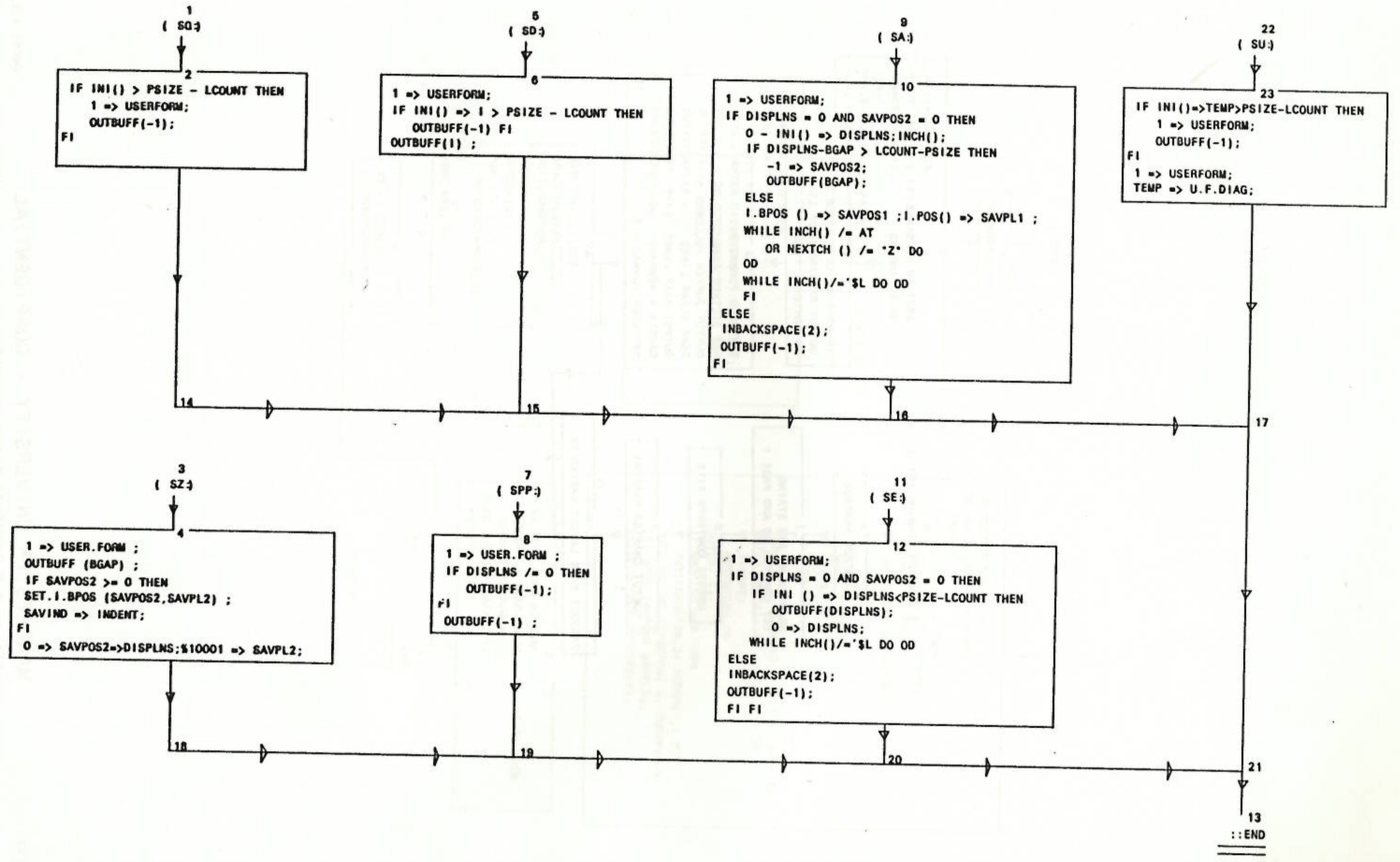


Figure 7 Y.DOC01100 6 August 1982



```
IF INI() > PSIZE - LCOUNT THEN
   1 => USERFORM;
   OUTBUFF(-1);
FI
```

```
1 => USERFORM;
IF INI() => I > PSIZE - LCOUNT THEN
   OUTBUFF(-1) FI
OUTBUFF(I) ;
```

```
1 => USERFORM;
IF DISPLNS = 0 AND SAVPOS2 = 0 THEN
   0 - INI() => DISPLNS;INCH();
   IF DISPLNS-BGAP > LCOUNT-PSIZE THEN
      -1 => SAVPOS2;
      OUTBUFF(BGAP);
   ELSE
   I.BPOS () => SAVPOS1 ;I.POS() => SAVPL1 ;
   WHILE INCH() /= AT
      OR NEXTCH () /= 'Z' DO
   OD
   WHILE INCH()/='$L DO OD
   FI
ELSE
INBACKSPACE(2);
OUTBUFF(-1);
FI
```

```
IF INI()=>TEMP>PSIZE-LCOUNT THEN
   1 => USERFORM;
   OUTBUFF(-1);
FI
1 => USERFORM;
TEMP => U.F.DIAG;
```

```
1 => USER.FORM ;
OUTBUFF (BGAP) ;
IF SAVPOS2 >= 0 THEN
SET.I.BPOS (SAVPOS2,SAVPL2) ;
SAVIND => INDENT;
FI
0 => SAVPOS2=>DISPLNS;%10001 => SAVPL2;
```

```
1 => USER.FORM ;
IF DISPLNS /= 0 THEN
   OUTBUFF(-1);
FI
OUTBUFF(-1) ;
```

```
1 => USERFORM;
IF DISPLNS = 0 AND SAVPOS2 = 0 THEN
   IF INI () => DISPLNS<PSIZE-LCOUNT THEN
      OUTBUFF(DISPLNS);
      0 => DISPLNS;
   WHILE INCH()/='$L DO OD
ELSE
INBACKSPACE(2);
OUTBUFF(-1);
FI FI
```

MANCHESTER UNIVERSITY - CONFIDENTIAL

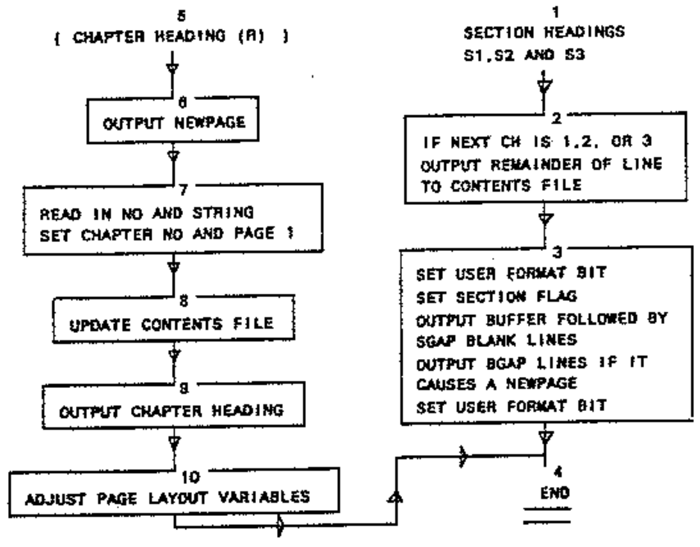
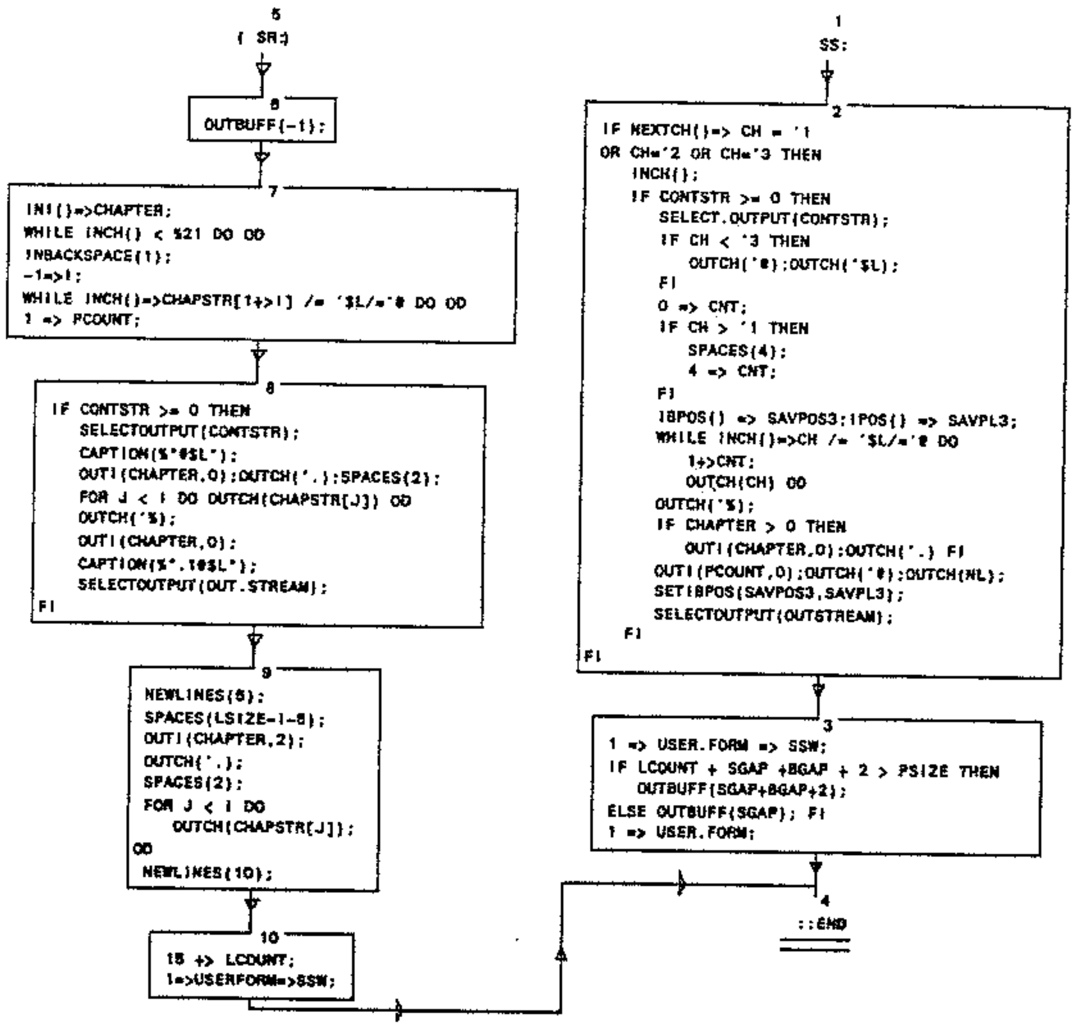


Figure 8 Y.DOC01100 6 August 1982



```
6
OUTBUFF(-1);
```

```
7
IN!()=>CHAPTER;
WHILE INCH() < %21 DO OD
INBACKSPACE(1);
-1=>I;
WHILE INCH()=>CHAPSTR[1+>I] /= '$L/='# DO OD
1 => PCOUNT;
```

```
8
IF CONTSTR >= 0 THEN
   SELECTOUTPUT(CONTSTR);
   CAPTION(%"#$L");
   OUTI(CHAPTER,0);OUTCH('.);SPACES(2);
   FOR J < I DO OUTCH(CHAPSTR[J]) OD
   OUTCH('%);
   OUTI(CHAPTER,0);
   CAPTION(%".1#$L");
   SELECTOUTPUT(OUT.STREAM);
FI
```

```
9
NEWLINES(5);
SPACES(LSIZE-I-5);
OUTI(CHAPTER,2);
OUTCH('.);
SPACES(2);
FOR J < I DO
   OUTCH(CHAPSTR[J]);
OD
NEWLINES(10);
```

```
10
15 +> LCOUNT;
1=>USERFORM=>SSW;
```

```
2
IF NEXTCH()=> CH = '1
OR CH='2 OR CH='3 THEN
   INCH();
   IF CONTSTR >= 0 THEN
      SELECT.OUTPUT(CONTSTR);
      IF CH < '3 THEN
         OUTCH('#);OUTCH('$L);
      FI
      0 => CNT;
      IF CH > '1 THEN
         SPACES(4);
         4 => CNT;
      FI
      IBPOS() => SAVPOS3;IPOS() => SAVPL3;
      WHILE INCH()=>CH /= '$L/='# DO
         1+>CNT;
         OUTCH(CH) OD
      OUTCH('%);
      IF CHAPTER > 0 THEN
         OUTI(CHAPTER,0);OUTCH('.) FI
      OUTI(PCOUNT,0);OUTCH('#);OUTCH(NL);
      SETIBPOS(SAVPOS3,SAVPL3);
      SELECTOUTPUT(OUTSTREAM);
   FI
FI
```

```
3
1 => USER.FORM => SSW;
IF LCOUNT + SGAP +BGAP + 2 > PSIZE THEN
   OUTBUFF(SGAP+BGAP+2);
ELSE OUTBUFF(SGAP); FI
1 => USER.FORM;
```

Figure 8 Y.DOCO1110 6 August 1982

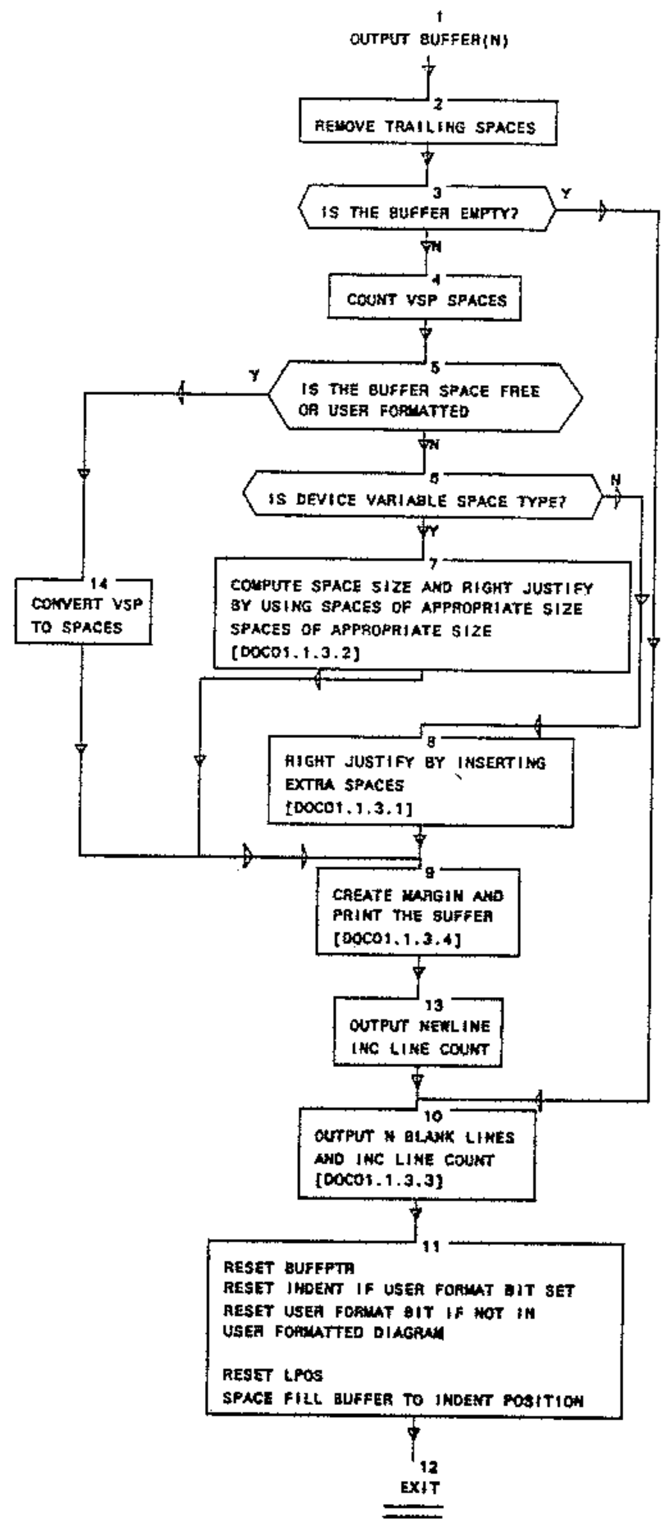
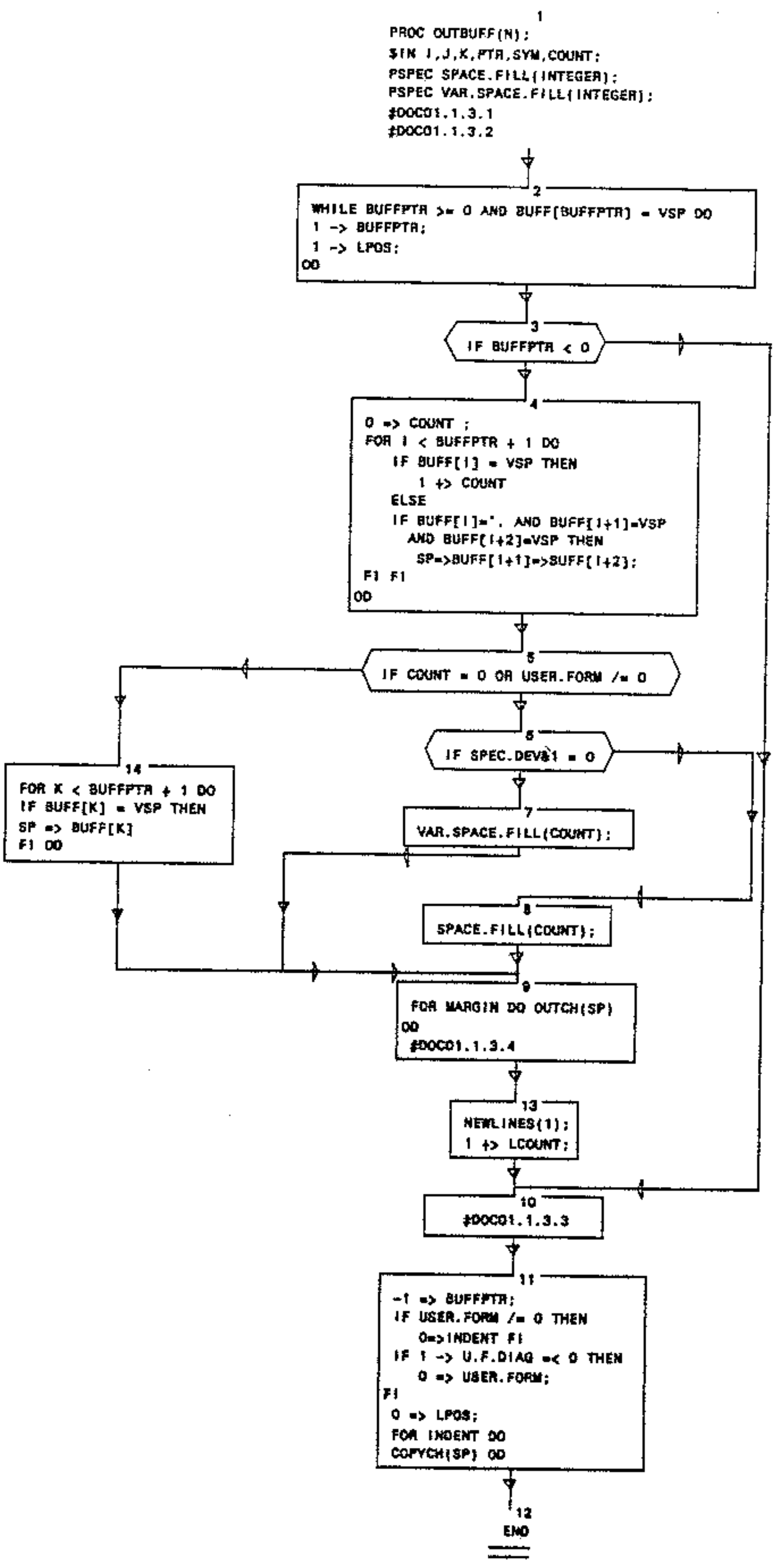


Figure 9 Y.DOC01100 6 August 1982



```
1
PROC OUTBUFF(N);
$IN I,J,K,PTR,SYM,COUNT;
PSPEC SPACE.FILL(INTEGER);
PSPEC VAR.SPACE.FILL(INTEGER);
#DOC01.1.3.1
#DOC01.1.3.2

2
WHILE BUFFPTR >= 0 AND BUFF[BUFFPTR] = VSP DO
 1 -> BUFFPTR;
 1 -> LPOS;
OD

3
IF BUFFPTR < 0

4
0 => COUNT ;
FOR I < BUFFPTR + 1 DO
   IF BUFF[I] = VSP THEN
      1 +> COUNT
   ELSE
   IF BUFF[I]='. AND BUFF[I+1]=VSP
     AND BUFF[I+2]=VSP THEN
      SP=>BUFF[I+1]=>BUFF[I+2];
 FI FI
OD

5
IF COUNT = 0 OR USER.FORM /= 0

6
IF SPEC.DEV&1 = 0

7
VAR.SPACE.FILL(COUNT);

8
SPACE.FILL(COUNT);

9
FOR MARGIN DO OUTCH(SP)
OD
#DOC01.1.3.4

13
NEWLINES(1);
1 +> LCOUNT;

10
#DOC01.1.3.3

11
-1 => BUFFPTR;
IF USER.FORM /= 0 THEN
   0=>INDENT FI
IF 1 -> U.F.DIAG =< 0 THEN
   0 => USER.FORM;
FI
0 => LPOS;
FOR INDENT DO
COPYCH(SP) OD

12
END

14
FOR K < BUFFPTR + 1 DO
IF BUFF[K] = VSP THEN
SP => BUFF[K]
FI OD
```

Figure 9 Y.DOC01110 6 August 1982

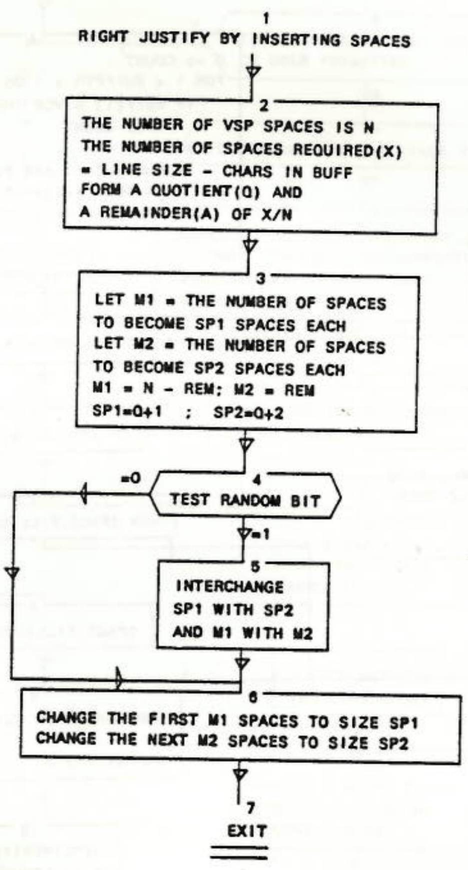
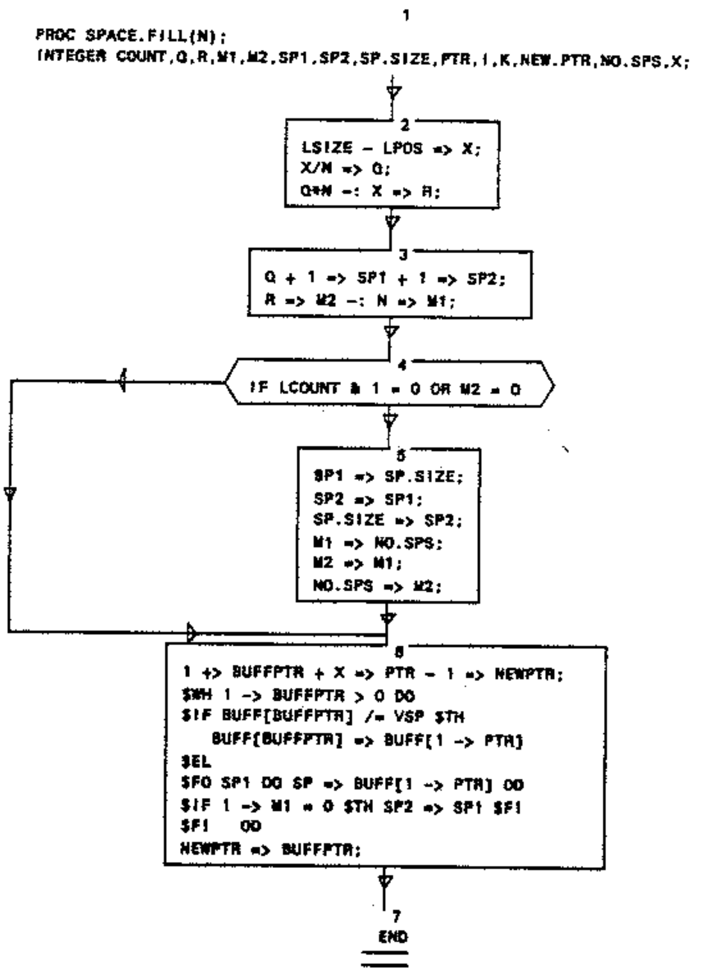

1
RIGHT JUSTIFY BY INSERTING SPACES
2
THE NUMBER OF VSP SPACES IS N
THE NUMBER OF SPACES REQUIRED(X)
= LINE SIZE - CHARS IN BUFF
FORM A QUOTIENT(Q) AND
A REMAINDER(A) OF X/N
3
LET M1 = THE NUMBER OF SPACES
TO BECOME SP1 SPACES EACH
LET M2 = THE NUMBER OF SPACES
TO BECOME SP2 SPACES EACH
M1 = N - REM; M2 = REM
SP1=Q+1 ; SP2=Q+2
=0
4
TEST RANDOM BIT
=1
5
INTERCHANGE
SP1 WITH SP2
AND M1 WITH M2
6
CHANGE THE FIRST M1 SPACES TO SIZE SP1
CHANGE THE NEXT M2 SPACES TO SIZE SP2
7
EXIT

Figure 10 Y.DOCO1100 6 August 1982



```
1
PROC SPACE.FILL(N);
INTEGER COUNT,Q,R,M1,M2,SP1,SP2,SP.SIZE,PTR,I,K,NEW.PTR,NO.SPS,X;

2
LSIZE - LPOS => X;
X/N => Q;
Q*N -: X => R;

3
Q + 1 => SP1 + 1 => SP2;
R => M2 -: N => M1;

4
IF LCOUNT & 1 = 0 OR M2 = 0

5
SP1 => SP.SIZE;
SP2 => SP1;
SP.SIZE => SP2;
M1 => NO.SPS;
M2 => M1;
NO.SPS => M2;

6
1 +> BUFFPTR + X => PTR - 1 => NEWPTR;
$WH 1 -> BUFFPTR > 0 DO
$IF BUFF[BUFFPTR] /= VSP $TH
   BUFF[BUFFPTR] => BUFF[1 -> PTR]
$EL
$FO SP1 DO SP => BUFF[1 -> PTR] OD
$IF 1 -> M1 = 0 $TH SP2 => SP1 $FI
$FI   OD
NEWPTR => BUFFPTR;

7
END
```

Figure 10 Y.DOC01111 6 August 1982

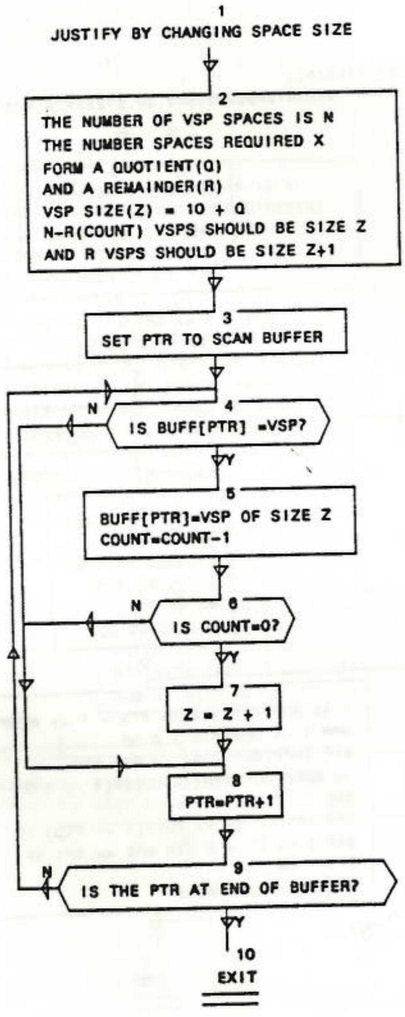
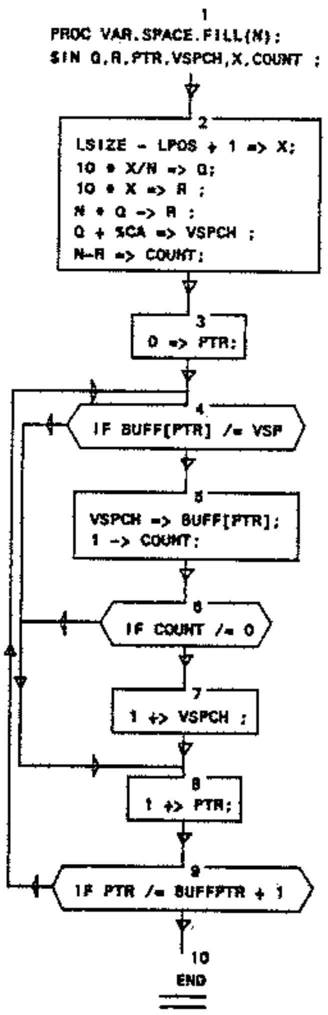


Figure 11 Y.DOC01100 6 August 1982



```
1
PROC VAR.SPACE.FILL(N);
$IN Q,R,PTR,VSPCH,X,COUNT ;

2
LSIZE - LPOS + 1 => X;
10 * X/N => Q;
10 * X => R ;
N * Q -> R ;
Q + %CA => VSPCH ;
N-R => COUNT;

3
0 => PTR;

4
IF BUFF[PTR] /= VSP

5
VSPCH => BUFF[PTR];
1 -> COUNT;

6
IF COUNT /= 0

7
1 +> VSPCH ;

8
1 +> PTR;

9
IF PTR /= BUFFPTR + 1

10
END
```

Figure 11 Y.DOCO1111 6 August 1982

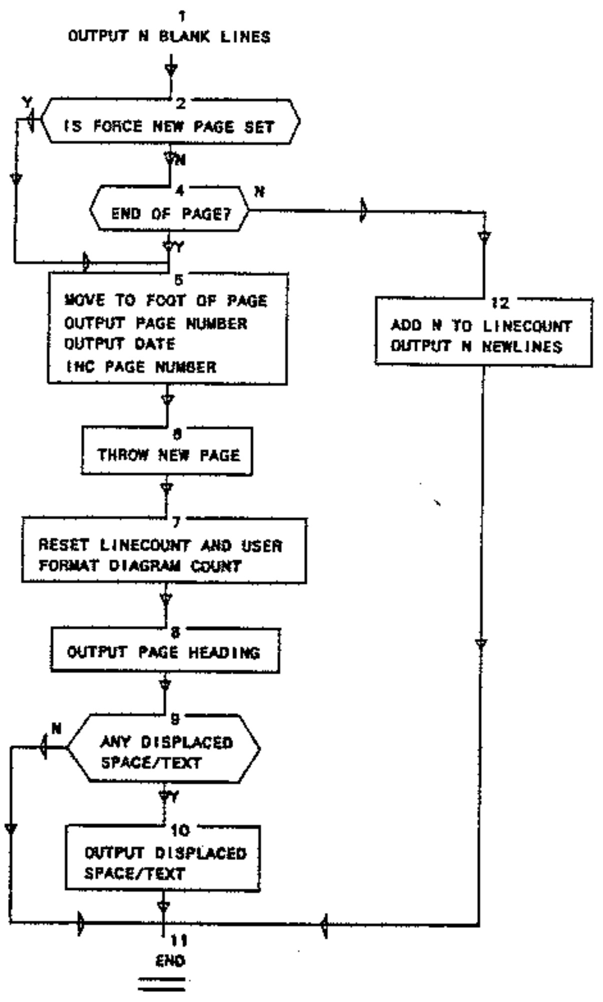
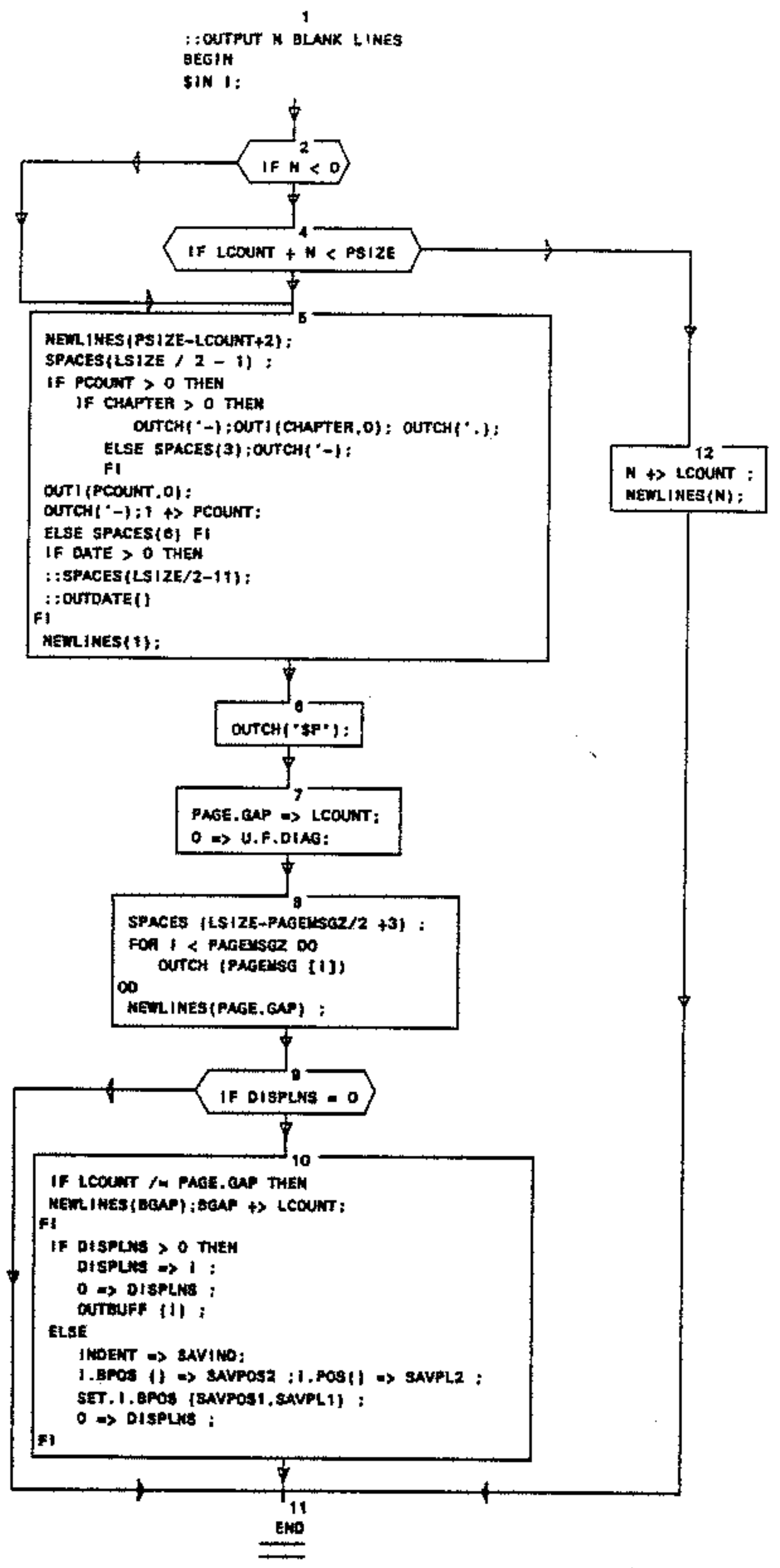


Figure 12 Y.DOC01100 6 August 1982



```
1
::OUTPUT N BLANK LINES
BEGIN
$IN I;
```

```
2
IF N < 0
```

```
4
IF LCOUNT + N < PSIZE
```

```
5
NEWLINES(PSIZE-LCOUNT+2);
SPACES(LSIZE / 2 - 1) ;
IF PCOUNT > 0 THEN
   IF CHAPTER > 0 THEN
        OUTCH('-);OUTI(CHAPTER,0); OUTCH('.);
      ELSE SPACES(3);OUTCH('-);
      FI
OUTI(PCOUNT,0);
OUTCH('-);1 +> PCOUNT;
ELSE SPACES(6) FI
IF DATE > 0 THEN
::SPACES(LSIZE/2-11);
::OUTDATE()
FI
NEWLINES(1);
```

```
6
OUTCH("$P");
```

```
7
PAGE.GAP => LCOUNT;
0 => U.F.DIAG;
```

```
8
SPACES (LSIZE-PAGEMSGZ/2 +3) ;
FOR I < PAGEMSGZ DO
   OUTCH (PAGEMSG [I])
OD
NEWLINES(PAGE.GAP) ;
```

```
9
IF DISPLNS = 0
```

```
10
IF LCOUNT /= PAGE.GAP THEN
NEWLINES(BGAP);BGAP +> LCOUNT;
FI
IF DISPLNS > 0 THEN
   DISPLNS => I ;
   0 => DISPLNS ;
   OUTBUFF (I) ;
ELSE
   INDENT => SAVIND;
   I.BPOS () => SAVPOS2 ;I.POS() => SAVPL2 ;
   SET.I.BPOS (SAVPOS1,SAVPL1) ;
   0 => DISPLNS ;
FI
```

```
11
END
```

```
12
N +> LCOUNT ;
NEWLINES(N);
```

Figure 12 Y.DOC01111 6 August 1982

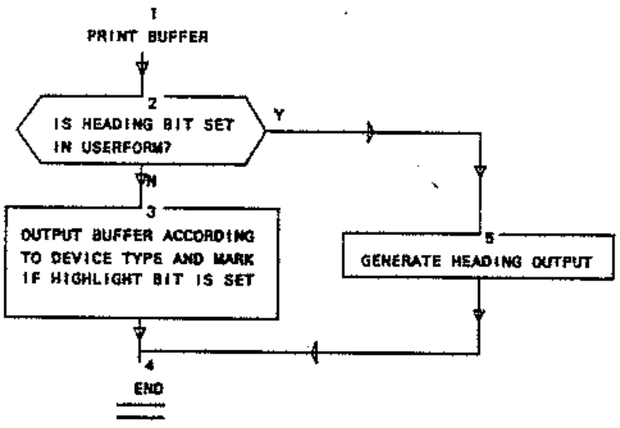
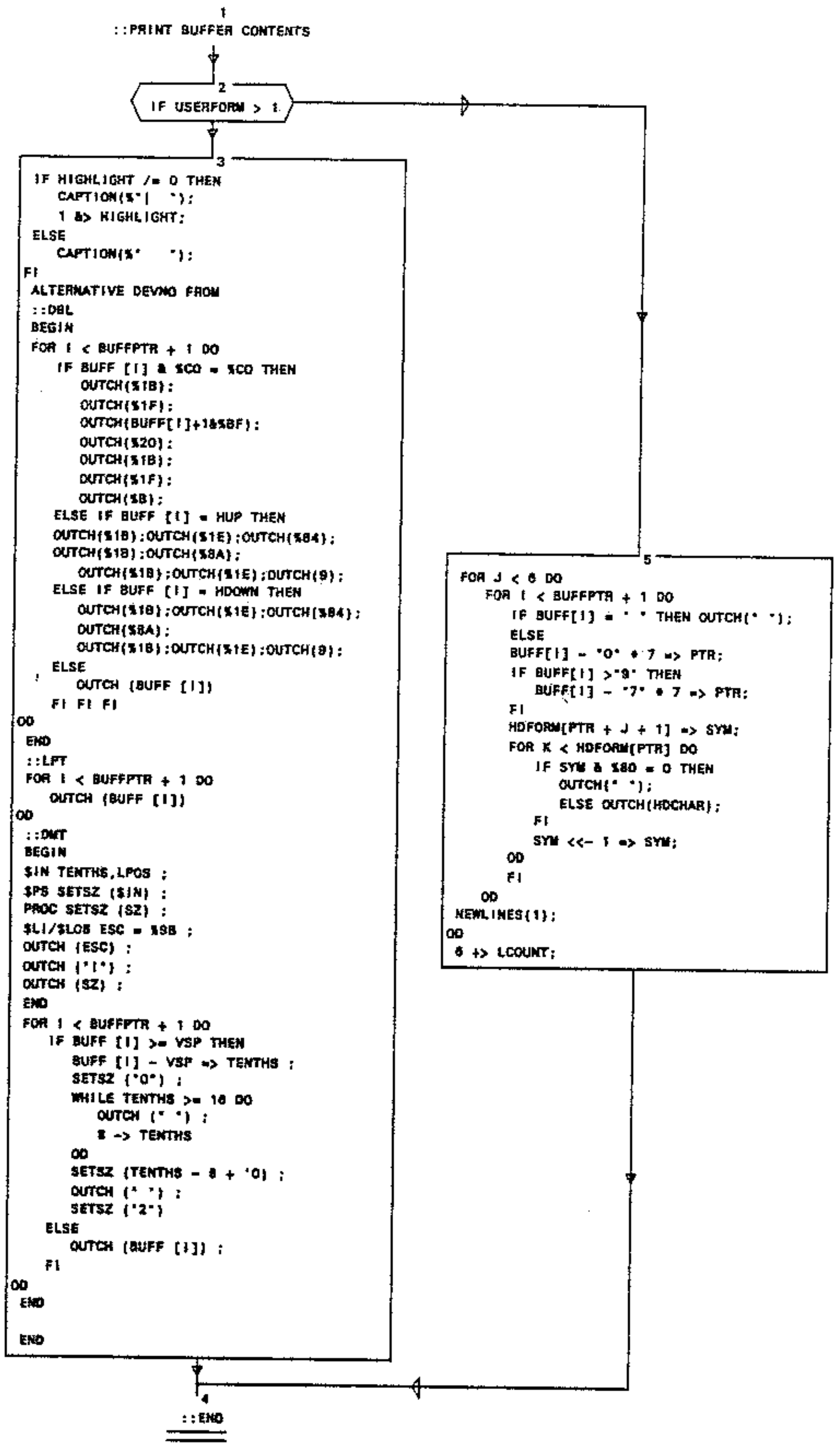

1
PRINT BUFFER
2
IS HEADING BIT SET IN USERFORM?
Y
N
3
OUTPUT BUFFER ACCORDING TO DEVICE TYPE AND MARK IF HIGHLIGHT BIT IS SET
5
GENERATE HEADING OUTPUT
4
END

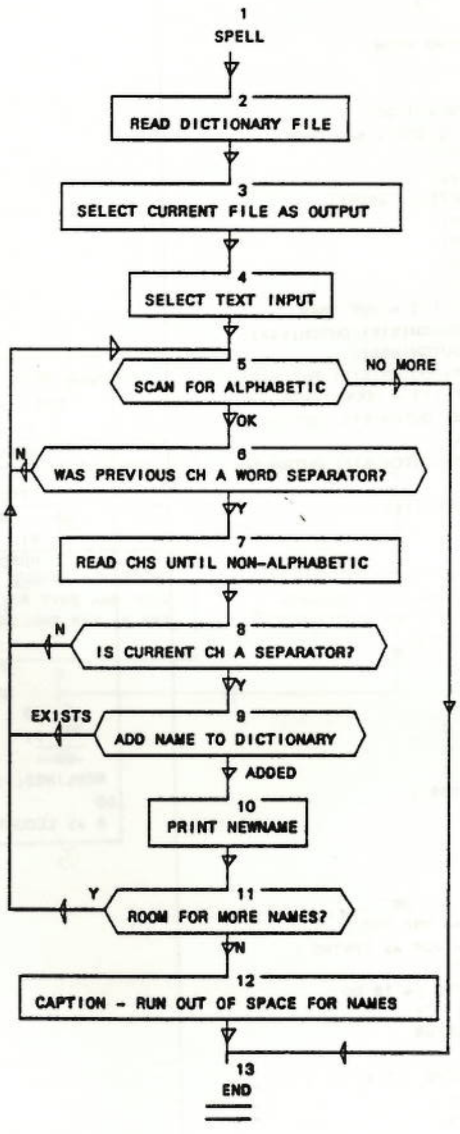
Figure 13 Y.DOC01101 6 August 1982



```
 IF HIGHLIGHT /= 0 THEN
    CAPTION(%"|   ");
    1 &> HIGHLIGHT;
 ELSE
    CAPTION(%"    ");
FI
 ALTERNATIVE DEVNO FROM
 ::DBL
 BEGIN
 FOR I < BUFFPTR + 1 DO
    IF BUFF [I] & %C0 = %C0 THEN
       OUTCH(%1B);
       OUTCH(%1F);
       OUTCH(BUFF[I]+1&%BF);
       OUTCH(%20);
       OUTCH(%1B);
       OUTCH(%1F);
       OUTCH(%B);
   ELSE IF BUFF [I] = HUP THEN
   OUTCH(%1B);OUTCH(%1E);OUTCH(%84);
   OUTCH(%1B);OUTCH(%8A);
      OUTCH(%1B);OUTCH(%1E);OUTCH(9);
   ELSE IF BUFF [I] = HDOWN THEN
      OUTCH(%1B);OUTCH(%1E);OUTCH(%84);
      OUTCH(%8A);
      OUTCH(%1B);OUTCH(%1E);OUTCH(9);
   ELSE
      OUTCH (BUFF [I])
   FI FI FI
OD
 END
 ::LPT
 FOR I < BUFFPTR + 1 DO
    OUTCH (BUFF [I])
OD
 ::DMT
 BEGIN
 $IN TENTHS,LPOS ;
 $PS SETSZ ($IN) ;
 PROC SETSZ (SZ) ;
 $LI/$LO8 ESC = %9B ;
 OUTCH (ESC) ;
 OUTCH ("[") ;
 OUTCH (SZ) ;
 END
 FOR I < BUFFPTR + 1 DO
    IF BUFF [I] >= VSP THEN
       BUFF [I] - VSP => TENTHS ;
       SETSZ ("0") ;
       WHILE TENTHS >= 16 DO
          OUTCH (" ") ;
          8 -> TENTHS
       OD
       SETSZ (TENTHS - 8 + '0) ;
       OUTCH (" ") ;
       SETSZ ("2")
    ELSE
       OUTCH (BUFF [I]) ;
    FI
OD
 END

 END
```

```
 FOR J < 6 DO
    FOR I < BUFFPTR + 1 DO
       IF BUFF[I] = " " THEN OUTCH(" ");
       ELSE
       BUFF[I] - "0" + 7 => PTR;
       IF BUFF[I] >"9" THEN
          BUFF[I] - "7" + 7 => PTR;
       FI
       HDFORM[PTR + J + 1] => SYM;
       FOR K < HDFORM[PTR] DO
          IF SYM & %80 = 0 THEN
             OUTCH(" ");
             ELSE OUTCH(HDCHAR);
          FI
          SYM <<- 1 => SYM;
       OD
       FI
   OD
NEWLINES(1);
OD
 6 +> LCOUNT;
```

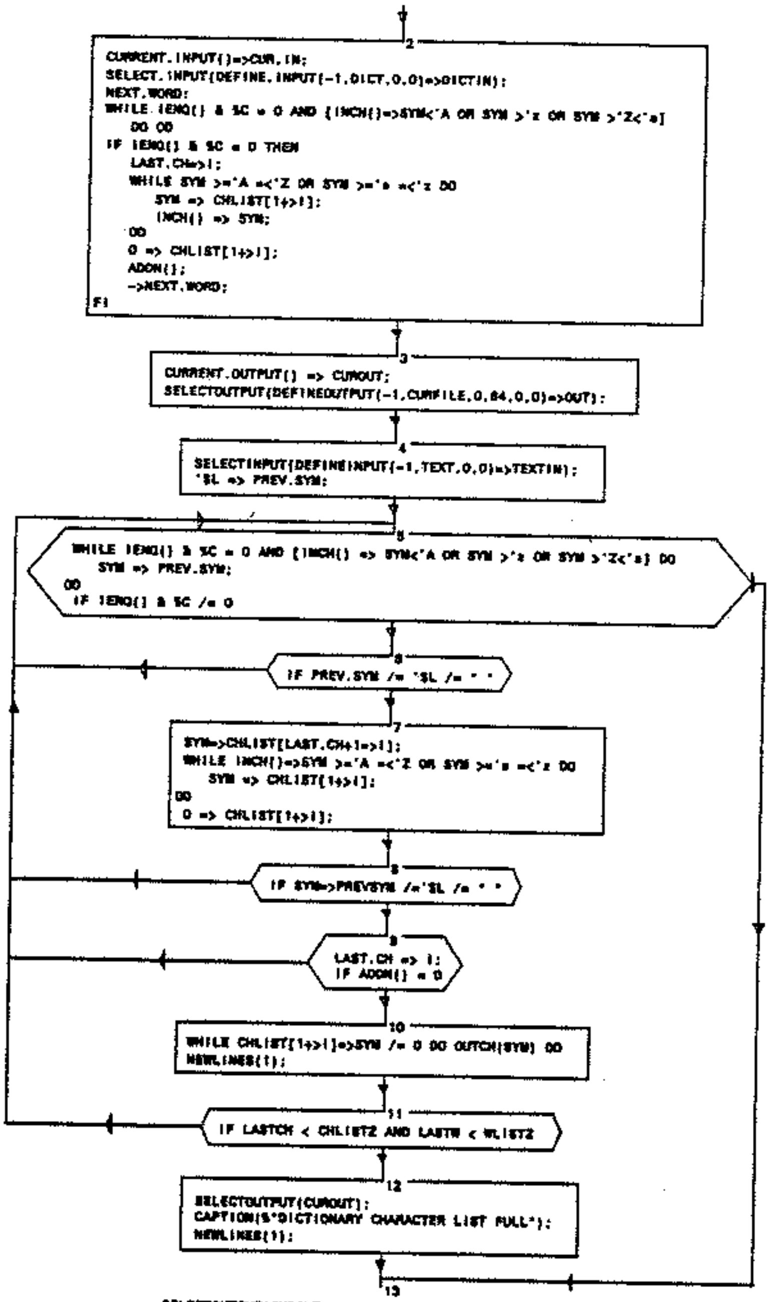
Figure 13 Y.DOC01111 6 August 1982



1 SPELL
2 READ DICTIONARY FILE
3 SELECT CURRENT FILE AS OUTPUT
4 SELECT TEXT INPUT
5 SCAN FOR ALPHABETIC
NO MORE
OK
6 WAS PREVIOUS CH A WORD SEPARATOR?
N
Y
7 READ CHS UNTIL NON-ALPHABETIC
8 IS CURRENT CH A SEPARATOR?
N
Y
9 ADD NAME TO DICTIONARY
EXISTS
ADDED
10 PRINT NEWNAME
11 ROOM FOR MORE NAMES?
Y
N
12 CAPTION - RUN OUT OF SPACE FOR NAMES
13 END

Figure 14 Y.DOCO1101 6 August 1982

```
PROC SPELL(DICT,TEXT);
$IN I,CH,SYM,CURIN,CUROUT,OUT,DICTIN,TEXTIN,PREVSYM,LASTCH,LASTW;
$LI/$AD[$LO8] CURFILE= ;
::POP $IN S1,S2,S3;
::POP CREATE.SEGMENT(-1,%2000);MAP(PW1=>S1,4,1);
::POP CREATE.SEGMENT(-1,%2000);MAP(PW1=>S2,5,1);
::POP CREATE.SEGMENT(-1,%2000);MAP(PW1=>S3,6,1);
::MU6 RELEASE.SEGMENT(59);
::MU6 CREATE.SEGMENT(59,%10000);
::MC68000 RELEASE.SEGMENT(48);RELEASE.SEGMENT(49);
::MC68000 RELEASE.SEGMENT(50);RELEASE.SEGMENT(51);
::MC68000 CREATE.SEGMENT(48,%4000);CREATE.SEGMENT(49,%4000);
::MC68000 CREATE.SEGMENT(50,%4000);CREATE.SEGMENT(51,%4000);
::MC68000 MAP(48,-1,0);MAP(49,-1,0);MAP(50,-1,0);MAP(51,-1,0);
::POP $LI CHLISTZ=15000,WLISTZ=2200;
::MU6 $LI CHLISTZ=40000,WLISTZ=6000;
::MC68000 $LI CHLISTZ=40000,WLISTZ=6000;
-1 => LASTCH => LASTW;
$PS ADON()/$IN;
$DOC01.2.1
```



```
2
CURRENT.INPUT()=>CUR.IN;
SELECT.INPUT(DEFINE.INPUT(-1,DICT,0,0)=>DICTIN);
NEXT.WORD:
WHILE 1ENQ() & SC = 0 AND (INCH()=>SYM<'A OR SYM >'z OR SYM >'Z<'a]
   DO OD
IF 1ENQ() & SC = 0 THEN
   LAST.CH=>I;
   WHILE SYM >='A =<'Z OR SYM >='a =<'z DO
      SYM => CHLIST[1+>I];
      INCH() => SYM;
   OD
   0 => CHLIST[1+>I];
   ADON();
   ->NEXT.WORD;
FI

3
CURRENT.OUTPUT() => CUROUT;
SELECTOUTPUT(DEFINEOUTPUT(-1,CURFILE,0,64,0,0)=>OUT);

4
SELECTINPUT(DEFINEINPUT(-1,TEXT,0,0)=>TEXTIN);
'$L => PREV.SYM;

5
WHILE 1ENQ() & SC = 0 AND (INCH() => SYM<'A OR SYM >'z OR SYM >'Z<'a) DO
    SYM => PREV.SYM;
OD
 IF 1ENQ() & SC /= 0

6
IF PREV.SYM /= '$L /= ' '

7
SYM=>CHLIST[LAST.CH+1=>I];
WHILE INCH()=>SYM >='A =<'Z OR SYM >='a =<'z DO
   SYM => CHLIST[1+>I];
OD
 0 => CHLIST[1+>I];

8
IF SYM=>PREVSYM /='$L /= ' '

9
LAST.CH => I;
IF ADON() = 0

10
WHILE CHLIST[1+>I]=>SYM /= 0 DO OUTCH(SYM) OD
NEWLINES(1);

11
IF LASTCH < CHLISTZ AND LASTW < WLISTZ

12
SELECTOUTPUT(CUROUT);
CAPTION(%"DICTIONARY CHARACTER LIST FULL");
NEWLINES(1);

13
```

```
SELECTOUTPUT(CUROUT);
SELECTINPUT(CURIN);
END.OUTPUT(OUT,0);
ENDINPUT(DICTIN,0);
ENDINPUT(TEXTIN,0);
::POP RELEASE.SEGMENT(S1);RELEASE.SEGMENT(S2);
::POP RELEASE.SEGMENT(S3);
::MU6 RELEASE.SEGMENT(59);
::MC68000 RELEASE.SEGMENT(48);RELEASE.SEGMENT(49);
::MC68000 RELEASE.SEGMENT(50);RELEASE.SEGMENT(51);
END
```

Figure 14 Y.DOC01111 6 August 1982

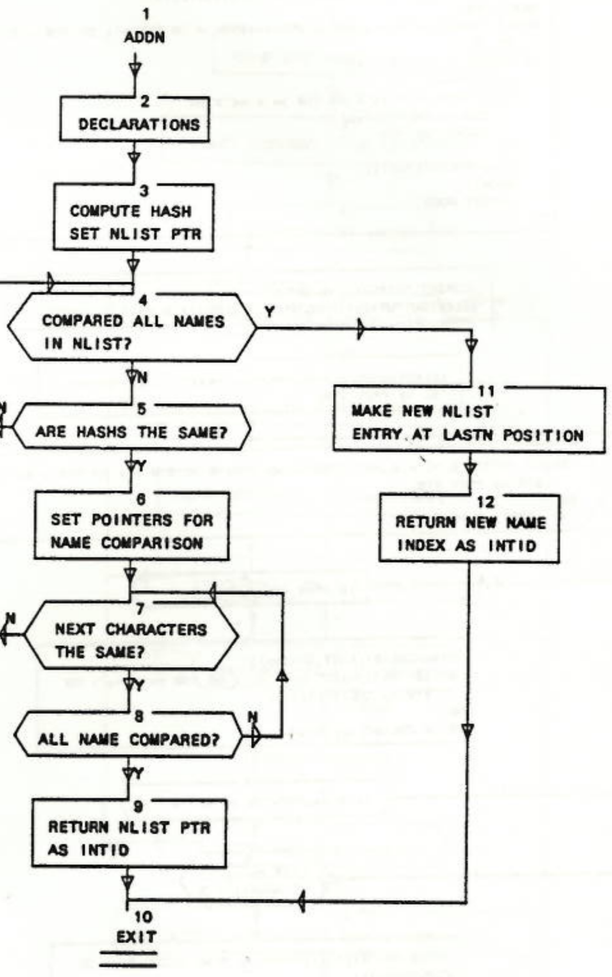
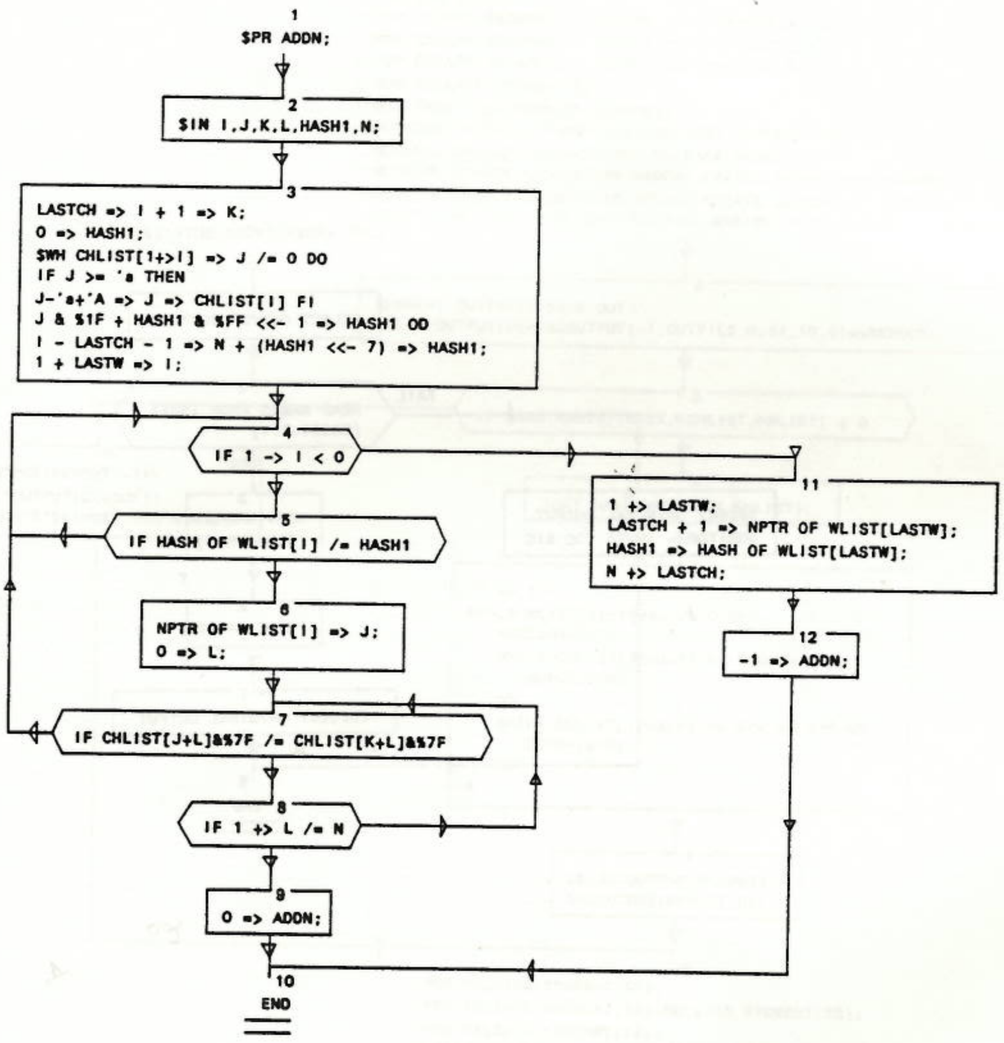

1
ADDN
2
DECLARATIONS
3
COMPUTE HASH
SET NLIST PTR
4
COMPARED ALL NAMES IN NLIST?
Y
N
5
ARE HASHS THE SAME?
N
Y
6
SET POINTERS FOR NAME COMPARISON
7
NEXT CHARACTERS THE SAME?
N
Y
8
ALL NAME COMPARED?
N
Y
9
RETURN NLIST PTR AS INTID
10
EXIT
11
MAKE NEW NLIST ENTRY AT LASTN POSITION
12
RETURN NEW NAME INDEX AS INTID

Figure 15 Y.DOCO1101 6 August 1982



```
1
$PR ADDN;

2
$IN I,J,K,L,HASH1,N;

3
LASTCH => I + 1 => K;
0 => HASH1;
$WH CHLIST[1+>I] => J /= 0 DO
IF J >= 'a THEN
J-'a+'A => J => CHLIST[I] FI
J & %1F + HASH1 & %FF <<- 1 => HASH1 OD
I - LASTCH - 1 => N + (HASH1 <<- 7) => HASH1;
1 + LASTW => I;

4
IF 1 -> I < 0

5
IF HASH OF WLIST[I] /= HASH1

6
NPTR OF WLIST[I] => J;
0 => L;

7
IF CHLIST[J+L]&%7F /= CHLIST[K+L]&%7F

8
IF 1 +> L /= N

9
0 => ADDN;

10
END

11
1 +> LASTW;
LASTCH + 1 => NPTR OF WLIST[LASTW];
HASH1 => HASH OF WLIST[LASTW];
N +> LASTCH;

12
-1 => ADDN;
```

Figure 15 Y.DOCO1111 6 August 1982

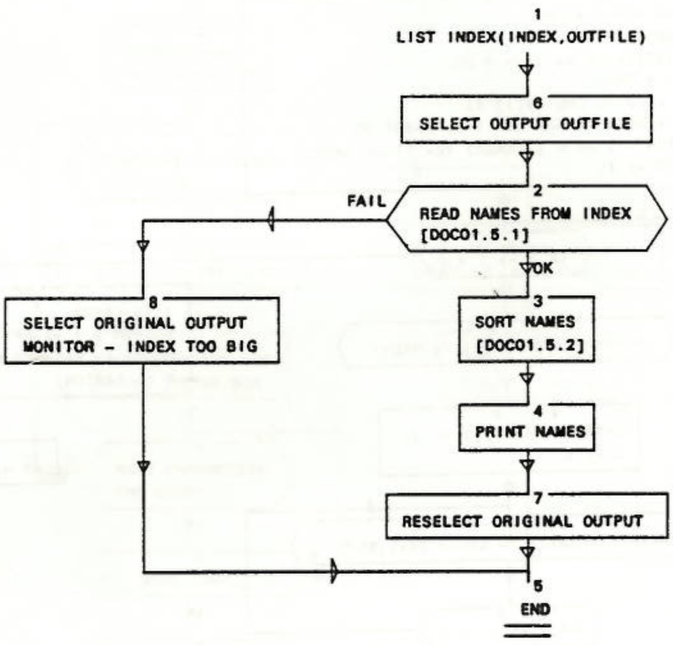
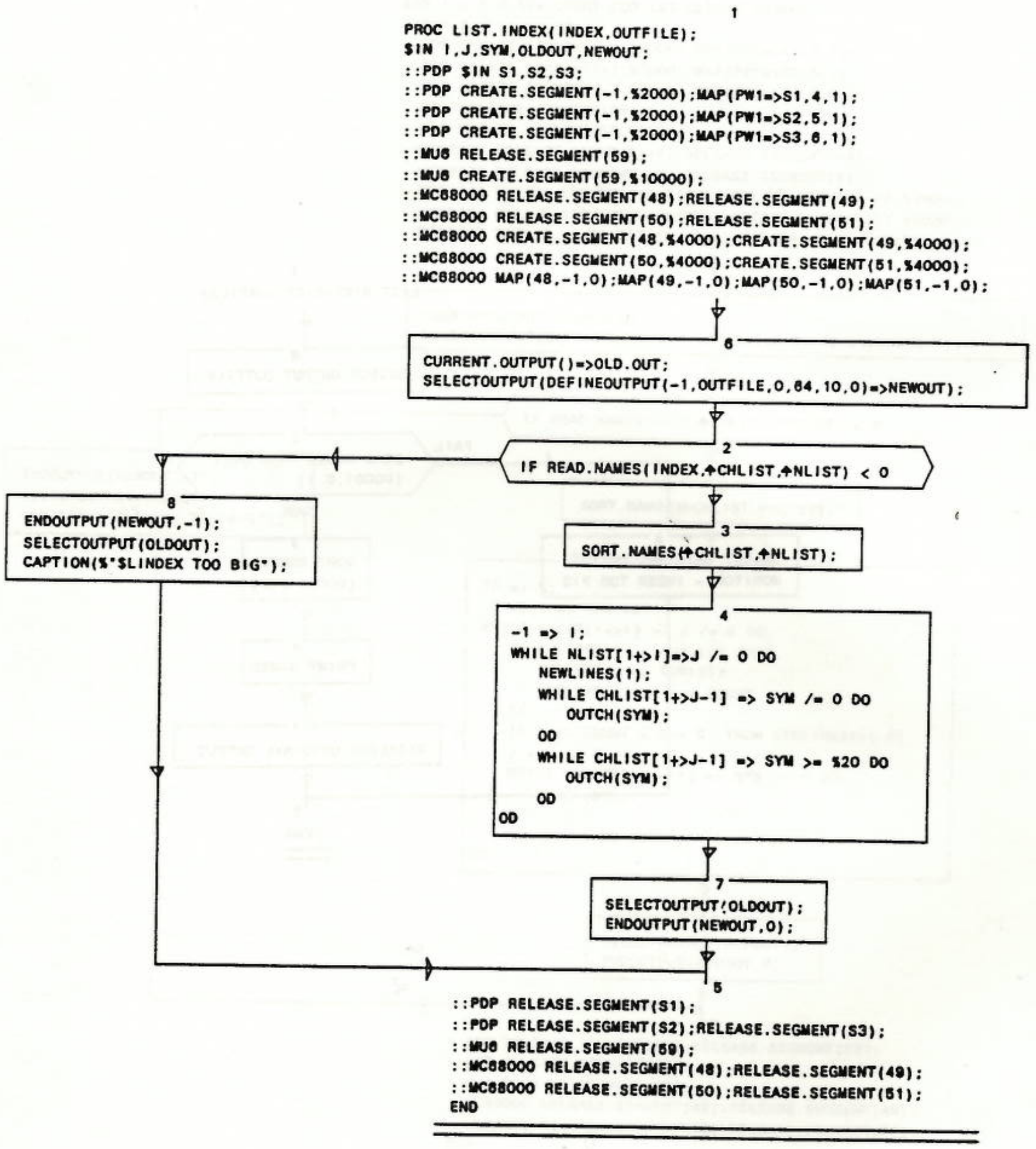


Figure 16 Y.DOCO1101 6 August 1982



```
1
PROC LIST.INDEX(INDEX,OUTFILE);
$IN I,J,SYM,OLDOUT,NEWOUT;
::PDP $IN S1,S2,S3;
::PDP CREATE.SEGMENT(-1,%2000);MAP(PW1=>S1,4,1);
::PDP CREATE.SEGMENT(-1,%2000);MAP(PW1=>S2,5,1);
::PDP CREATE.SEGMENT(-1,%2000);MAP(PW1=>S3,6,1);
::MU6 RELEASE.SEGMENT(59);
::MU6 CREATE.SEGMENT(59,%10000);
::MC68000 RELEASE.SEGMENT(48);RELEASE.SEGMENT(49);
::MC68000 RELEASE.SEGMENT(50);RELEASE.SEGMENT(51);
::MC68000 CREATE.SEGMENT(48,%4000);CREATE.SEGMENT(49,%4000);
::MC68000 CREATE.SEGMENT(50,%4000);CREATE.SEGMENT(51,%4000);
::MC68000 MAP(48,-1,0);MAP(49,-1,0);MAP(50,-1,0);MAP(51,-1,0);
```

```
6
CURRENT.OUTPUT()=>OLD.OUT;
SELECTOUTPUT(DEFINEOUTPUT(-1,OUTFILE,0,64,10,0)=>NEWOUT);
```

```
2
IF READ.NAMES(INDEX,↑CHLIST,↑NLIST) < 0
```

```
8
ENDOUTPUT(NEWOUT,-1);
SELECTOUTPUT(OLDOUT);
CAPTION(%"$LINDEX TOO BIG");
```

```
3
SORT.NAMES(↑CHLIST,↑NLIST);
```

```
4
-1 => I;
WHILE NLIST[1+>I]=>J /= 0 DO
   NEWLINES(1);
   WHILE CHLIST[1+>J-1] => SYM /= 0 DO
      OUTCH(SYM);
   OD
   WHILE CHLIST[1+>J-1] => SYM >= %20 DO
      OUTCH(SYM);
   OD
OD
```

```
7
SELECTOUTPUT(OLDOUT);
ENDOUTPUT(NEWOUT,0);
```

```
5
::PDP RELEASE.SEGMENT(S1);
::PDP RELEASE.SEGMENT(S2);RELEASE.SEGMENT(S3);
::MU6 RELEASE.SEGMENT(59);
::MC68000 RELEASE.SEGMENT(48);RELEASE.SEGMENT(49);
::MC68000 RELEASE.SEGMENT(50);RELEASE.SEGMENT(51);
END
```

Figure 16 Y.DOCO1111 6 August 1982

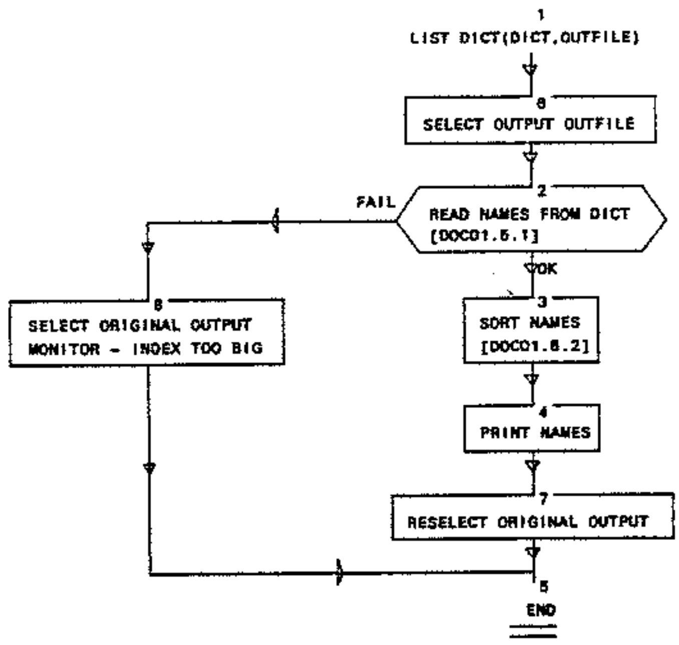
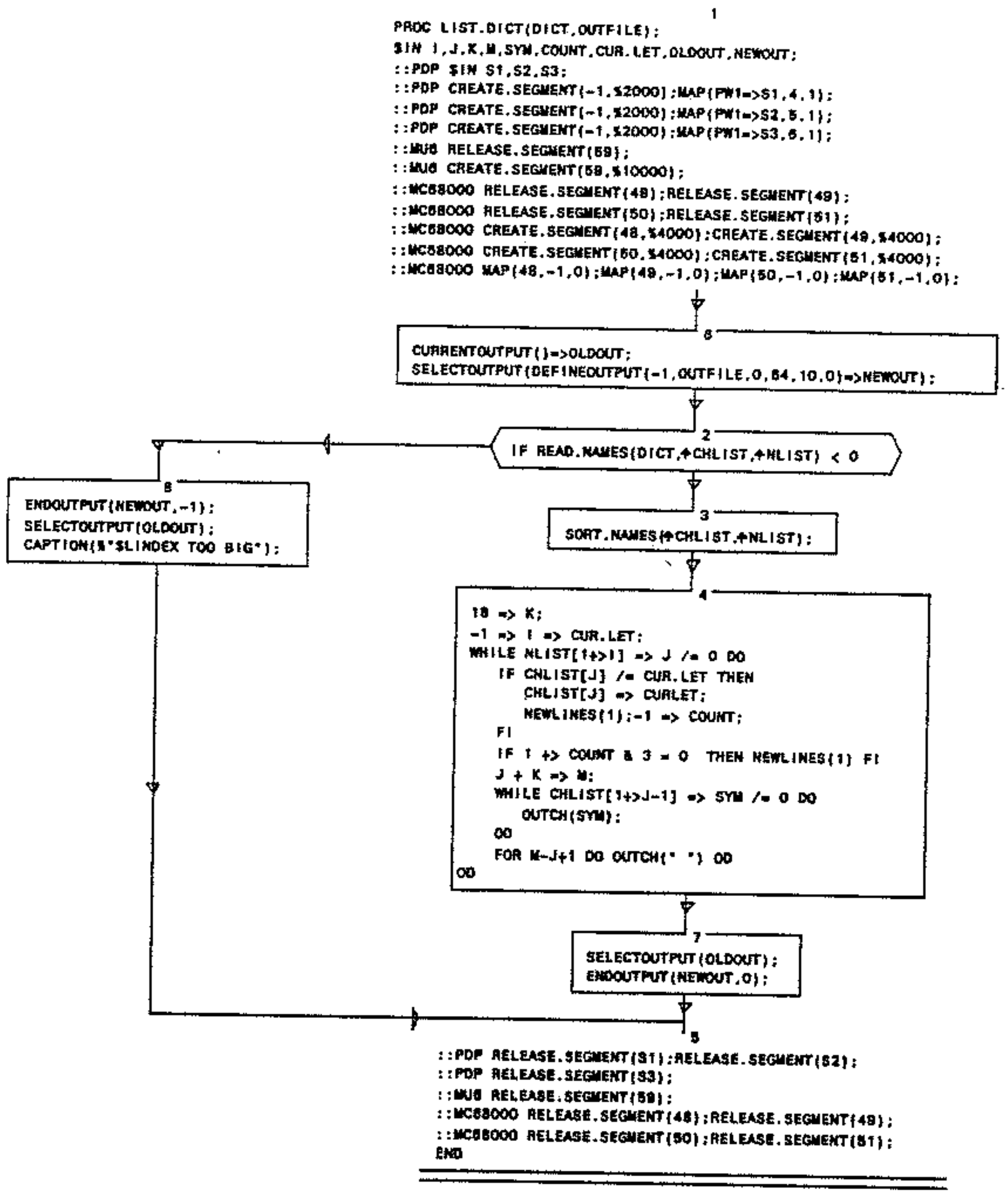


Figure 17 Y.DOC01101 6 August 1982



```
                                   1
PROC LIST.DICT(DICT,OUTFILE);
$IN I,J,K,M,SYM,COUNT,CUR.LET,OLDOUT,NEWOUT;
::PDP $IN S1,S2,S3;
::PDP CREATE.SEGMENT(-1,%2000);MAP(PW1=>S1,4,1);
::PDP CREATE.SEGMENT(-1,%2000);MAP(PW1=>S2,5,1);
::PDP CREATE.SEGMENT(-1,%2000);MAP(PW1=>S3,6,1);
::MU6 RELEASE.SEGMENT(59);
::MU6 CREATE.SEGMENT(59,%10000);
::MC68000 RELEASE.SEGMENT(48);RELEASE.SEGMENT(49);
::MC68000 RELEASE.SEGMENT(50);RELEASE.SEGMENT(51);
::MC68000 CREATE.SEGMENT(48,%4000);CREATE.SEGMENT(49,%4000);
::MC68000 CREATE.SEGMENT(50,%4000);CREATE.SEGMENT(51,%4000);
::MC68000 MAP(48,-1,0);MAP(49,-1,0);MAP(50,-1,0);MAP(51,-1,0);
```

```
6
CURRENTOUTPUT()=>OLDOUT;
SELECTOUTPUT(DEFINEOUTPUT(-1,OUTFILE,0,64,10,0)=>NEWOUT);
```

```
2
IF READ.NAMES(DICT,^CHLIST,^NLIST) < 0
```

```
8
ENDOUTPUT(NEWOUT,-1);
SELECTOUTPUT(OLDOUT);
CAPTION(%"$LINDEX TOO BIG");
```

```
3
SORT.NAMES(^CHLIST,^NLIST);
```

```
4
18 => K;
-1 => I => CUR.LET;
WHILE NLIST[1+>I] => J /= 0 DO
   IF CHLIST[J] /= CUR.LET THEN
      CHLIST[J] => CURLET;
      NEWLINES(1);-1 => COUNT;
   FI
   IF 1 +> COUNT & 3 = 0  THEN NEWLINES(1) FI
   J + K => M;
   WHILE CHLIST[1+>J-1] => SYM /= 0 DO
      OUTCH(SYM);
   OD
   FOR M-J+1 DO OUTCH(" ") OD
OD
```

```
7
SELECTOUTPUT(OLDOUT);
ENDOUTPUT(NEWOUT,0);
```

```
5
::PDP RELEASE.SEGMENT(S1);RELEASE.SEGMENT(S2);
::PDP RELEASE.SEGMENT(S3);
::MU6 RELEASE.SEGMENT(59);
::MC68000 RELEASE.SEGMENT(48);RELEASE.SEGMENT(49);
::MC68000 RELEASE.SEGMENT(50);RELEASE.SEGMENT(51);
END
```

Figure 17 Y.DOC01111 6 August 1982

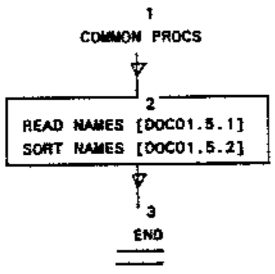
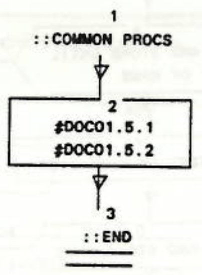


Figure 18 Y.DOC01101 6 August 1982


1
::COMMON PROCS
2
#DOCO1.5.1
#DOCO1.5.2
3
::END

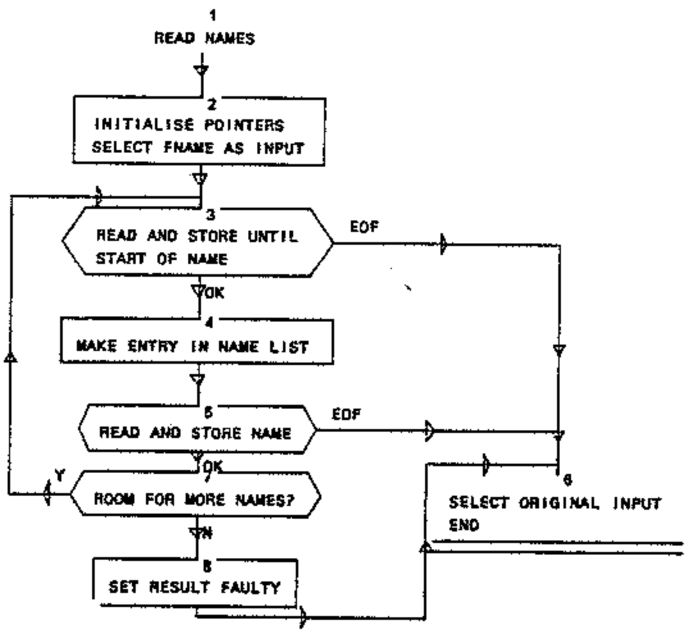
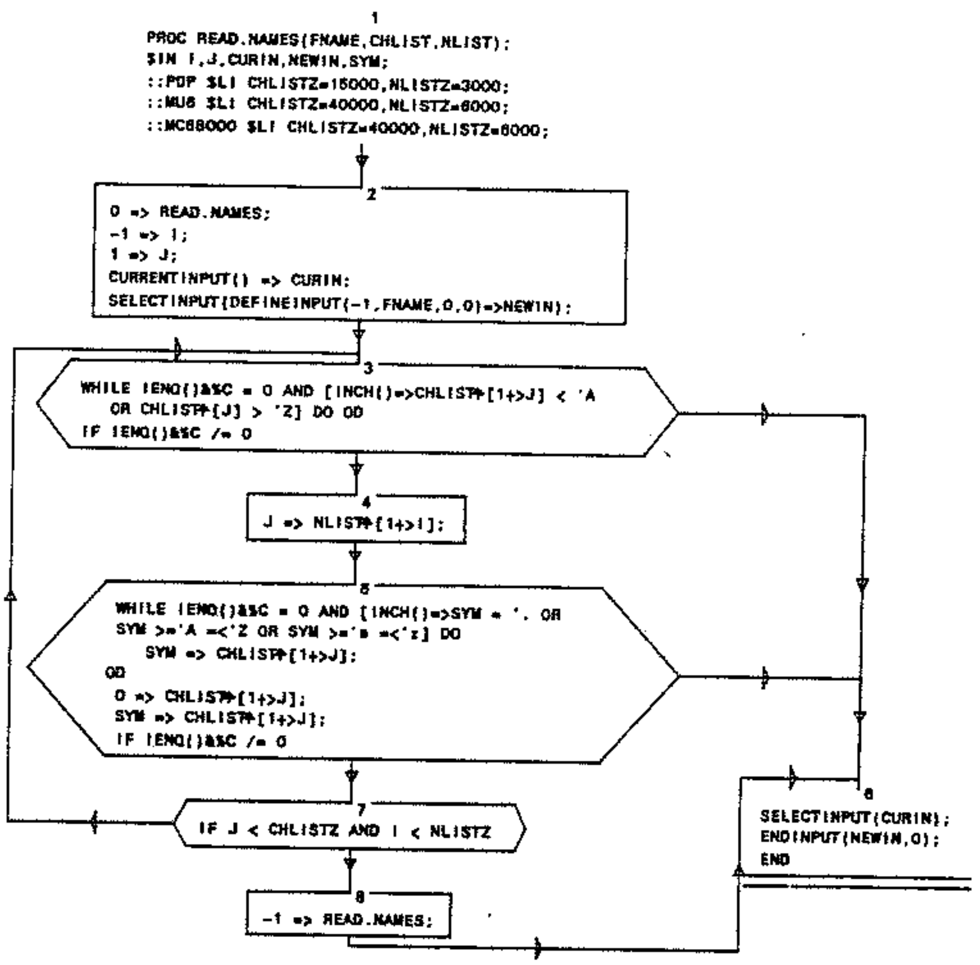


Figure 19 Y.DOC01101 6 August 1982



```
PROC READ.NAMES(FNAME,CHLIST,NLIST);
$IN I,J,CURIN,NEWIN,SYM;
::PDP $LI CHLISTZ=15000,NLISTZ=3000;
::MU6 $LI CHLISTZ=40000,NLISTZ=6000;
::MC68000 $LI CHLISTZ=40000,NLISTZ=6000;
```

```
0 => READ.NAMES;
-1 => I;
1 => J;
CURRENTINPUT() => CURIN;
SELECTINPUT(DEFINEINPUT(-1,FNAME,0,0)=>NEWIN);
```

```
WHILE IENQ()&SC = 0 AND [INCH()=>CHLIST^[1+>J] < 'A
   OR CHLIST^[J] > 'Z] DO OD
IF IENQ()&SC /= 0
```

```
J => NLIST^[1+>I];
```

```
WHILE IENQ()&SC = 0 AND [INCH()=>SYM = '. OR
SYM >='A =<'Z OR SYM >='a =<'z] DO
   SYM => CHLIST^[1+>J];
OD
 0 => CHLIST^[1+>J];
 SYM => CHLIST^[1+>J];
 IF IENQ()&SC /= 0
```

```
IF J < CHLISTZ AND I < NLISTZ
```

```
-1 => READ.NAMES;
```

```
SELECTINPUT(CURIN);
ENDINPUT(NEWIN,0);
END
```

Figure 19 Y.DOC01111 6 August 1982

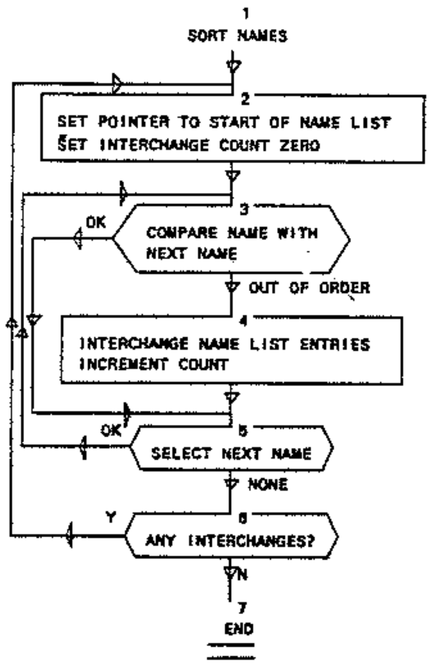
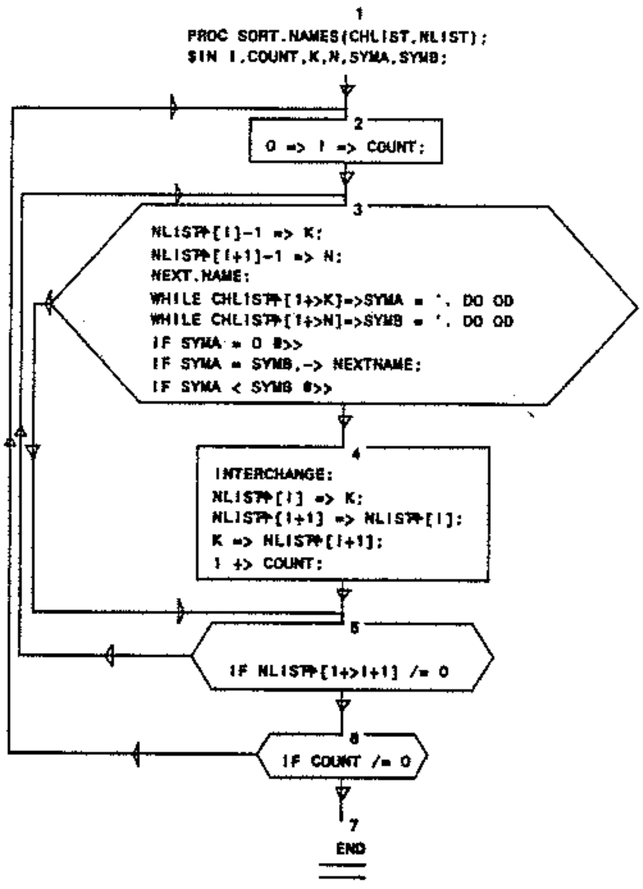

1
SORT NAMES
2
SET POINTER TO START OF NAME LIST
SET INTERCHANGE COUNT ZERO
3
OK
COMPARE NAME WITH NEXT NAME
OUT OF ORDER
4
INTERCHANGE NAME LIST ENTRIES
INCREMENT COUNT
OK
5
SELECT NEXT NAME
NONE
Y
6
ANY INTERCHANGES?
N
7
END

Figure 20 Y.DOC01101 6 August 1982



```
1
PROC SORT.NAMES(CHLIST,NLIST);
$IN I,COUNT,K,N,SYMA,SYMB;

2
0 => I => COUNT;

3
NLIST^[I]-1 => K;
NLIST^[I+1]-1 => N;
NEXT.NAME:
WHILE CHLIST^[1+>K]=>SYMA = '. DO OD
WHILE CHLIST^[1+>N]=>SYMB = '. DO OD
IF SYMA = 0 @>>
IF SYMA = SYMB,-> NEXTNAME;
IF SYMA < SYMB @>>

4
INTERCHANGE:
NLIST^[I] => K;
NLIST^[I+1] => NLIST^[I];
K => NLIST^[I+1];
1 +> COUNT;

5
IF NLIST^[1+>I+1] /= 0

6
IF COUNT /= 0

7
END
```

Figure 20 Y.DOC01111 6 August 1982

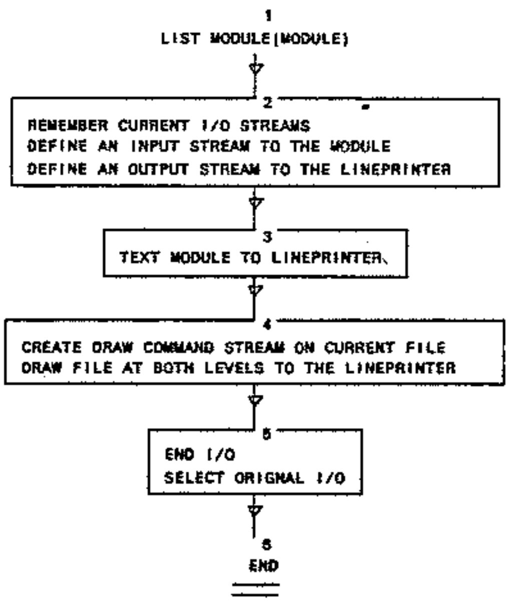
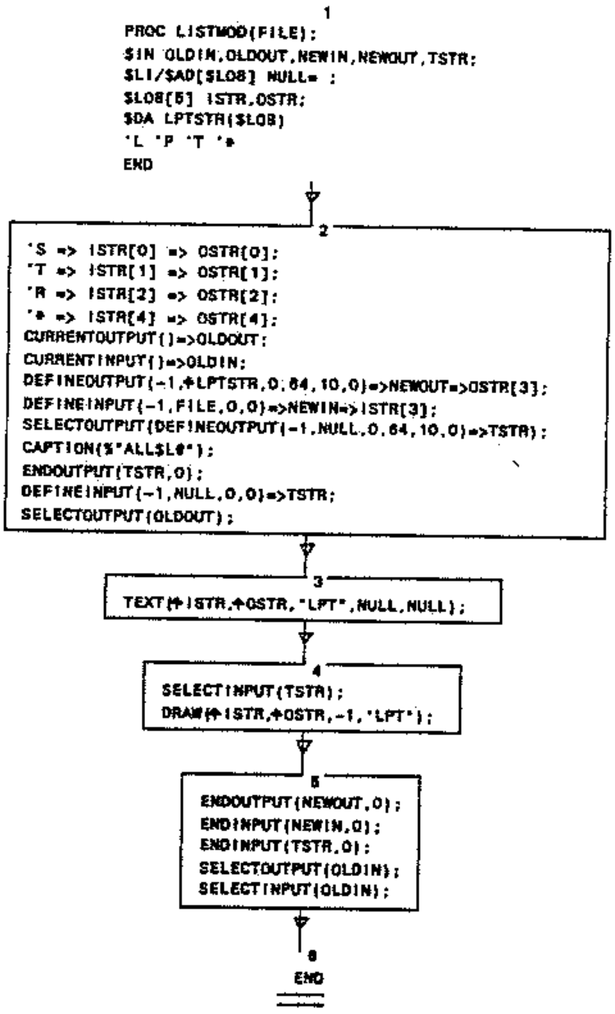

1
LIST MODULE(MODULE)
2
REMEMBER CURRENT I/O STREAMS
DEFINE AN INPUT STREAM TO THE MODULE
DEFINE AN OUTPUT STREAM TO THE LINEPRINTER
3
TEXT MODULE TO LINEPRINTER
4
CREATE DRAW COMMAND STREAM ON CURRENT FILE
DRAW FILE AT BOTH LEVELS TO THE LINEPRINTER
5
END I/O
SELECT ORIGNAL I/O
6
END

Figure 21 Y.DOC01101 6 August 1982



```
                    1
PROC LISTMOD(FILE);
$IN OLDIN,OLDOUT,NEWIN,NEWOUT,TSTR;
$LI/$AD[$LO8] NULL= ;
$LO8[5] ISTR,OSTR;
$DA LPTSTR($LO8)
'L 'P 'T '*
END
```

```
2
'S => ISTR[0] => OSTR[0];
'T => ISTR[1] => OSTR[1];
'R => ISTR[2] => OSTR[2];
'* => ISTR[4] => OSTR[4];
CURRENTOUTPUT()=>OLDOUT;
CURRENTINPUT()=>OLDIN;
DEFINEOUTPUT(-1,↑LPTSTR,0,64,10,0)=>NEWOUT=>OSTR[3];
DEFINEINPUT(-1,FILE,0,0)=>NEWIN=>ISTR[3];
SELECTOUTPUT(DEFINEOUTPUT(-1,NULL,0,64,10,0)=>TSTR);
CAPTION(%"ALL$L@");
ENDOUTPUT(TSTR,0);
DEFINEINPUT(-1,NULL,0,0)=>TSTR;
SELECTOUTPUT(OLDOUT);
```

```
3
TEXT(↑ISTR,↑OSTR,"LPT",NULL,NULL);
```

```
4
SELECTINPUT(TSTR);
DRAW(↑ISTR,↑OSTR,-1,'LPT');
```

```
5
ENDOUTPUT(NEWOUT,0);
ENDINPUT(NEWIN,0);
ENDINPUT(TSTR,0);
SELECTOUTPUT(OLDIN);
SELECTINPUT(OLDIN);
```

```
6
END
```

Figure 21 Y.DOC01111 6 August 1982