EDITOR

A COMPUTER PROGRAM
FOR
DOCUMENTATION PURPOSES

WILLIAM H. BOLTE

DECEMBER 1968

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

PICATINNY ARSENAL
DOVER, NEW JERSEY
TECHNICAL MEMORANDUM 1971

EDITOR - A COMPUTER PROGRAM
FOR DOCUMENTATION PURPOSES

BY

WILLIAM H. EILTE

DECEMBER 1968

DATA PROCESSING SYSTEMS OFFICE
PICATINNY ARSENAL
OCEAN, N.J.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>SECTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. ABSTRACT</td>
<td>1</td>
</tr>
<tr>
<td>2. INTRODUCTION</td>
<td>2</td>
</tr>
<tr>
<td>3. GENERAL DESCRIPTION</td>
<td>3</td>
</tr>
<tr>
<td>4. CODE DESCRIPTION &amp; APPLICATION</td>
<td>4</td>
</tr>
<tr>
<td>A. EDITOR PROGRAM CODES</td>
<td>4</td>
</tr>
<tr>
<td>4. INDIVIDUAL SAMPLE CODE ILLUSTRATIONS</td>
<td>7</td>
</tr>
<tr>
<td>C. ILLUSTRATIVE EXAMPLES</td>
<td>11</td>
</tr>
<tr>
<td>5. APPENDICES</td>
<td>15</td>
</tr>
<tr>
<td>A. FLOW CHART OF BASIC PROGRAM LOGIC</td>
<td>16</td>
</tr>
<tr>
<td>B. LISTING OF FORTRAN IV SOURCE PROGRAM</td>
<td>21</td>
</tr>
<tr>
<td>6. TABLE OF DISTRIBUTION</td>
<td>26</td>
</tr>
</tbody>
</table>
I. ABSTRACT

This report describes the application of an editing program called 'Editor' for documentation purposes. This program was written in FORTRAN IV language for the IBM-360/65 Digital Computer by Mr. Sidney Kravitz (Mechanical Engineer) of the Data Processing Systems Office, Picatinny Arsenal. It requires a total memory length of 20,592 bytes. This report enjoys the unique distinction of having been physically prepared by the very process it describes.

This program is presently being used on an experimental basis to evaluate its feasibility for Arsenal wide distribution.
2. INTRODUCTION

PROGRAM DOCUMENTATION CAN BE AND OFTEN IS A TEDIOUS, ALTHOUGH
NECESSARY, TASK UNDERTAKEN AT THE COMPLETION OF DEVELOPING A
COMPUTER PROGRAM. "EDITOR" WAS WRITTEN TO EXPEDITE THIS TASK.

"EDITOR" OFFERS A SOLUTION TO THE SHORTAGE OF AVAILABLE
TYPISTS AND SECRETARIES. IT TAKES ADVANTAGE OF THE IBM-1403 HIGH
SPEED PRINTER LOCATED AT THE PILATINNY COMPUTER CENTER, THUS
ENABLING THE USER TO HAVE HIS REPORTS "TYPED" BY THE COMPUTER.
ALL TEXT EDITING INSTRUCTIONS (LINE SPACING, UNDERLINING, CENTER-
ING, PAGE NUMBERING: ETC...) ARE PASSED TO THE "EDITOR" PROGRAM
BY MEANS OF ALPHA NUMERICAL CODES PUNCHED IN COLUMN 1 OF THE INPUT
DATA CARDS. THE DATA CARDS CONTAIN THE TEXT OF THE REPORT, HAVING
BEEN PUNCHED FROM FORTRAN CODING SHEETS UPON WHICH THE AUTHOR HAS
"WRITTEN" HIS REPORT.

THIS PROGRAM IS RATHER SIMPLE AND EASY TO USE. HOWEVER, IT IS
FELT THAT BETTER UNDERSTANDING OF THE INDIVIDUAL PROGRAM CODES
WILL BE ACHIEVED BY STUDYING ACTUAL SAMPLES AND VARIATIONS OF
EACH. THEREFORE, THE LATTER PART OF THIS REPORT WILL CONTAIN
SAMPLES OF MOST OF THE PROGRAM CODES AND THEIR EFFECTS AS WELL AS
AN ILLUSTRATIVE EXAMPLE OF PART OF THE ACTUAL CODED DECK USED FOR
THE DOCUMENTATION OF THIS REPORT.

PAGE 2
3. GENERAL DESCRIPTION

This program was designed to allow the printed output pages to fall within the common government standard (8 x 10-1/2 inches) page size. Page numbering is accomplished automatically starting after the first page, since this is often the title page. Roman numeral numbering may be initiated at the option of the user up to a maximum of XVII. Arabic numeral numbering then begins and continues until the end of this report. If desired, additional reports can immediately follow, separated by the appropriate logic control cards.

Normally 24 lines of printed information are produced per page of output (double spacing). If page overflow occurs, the information is continued on the next page with the proper page number sequence.

This program produces output on a one to one basis, that is, one punched card of information produces one line of output information. All punched cards do not necessarily require program codes (in column 1) all the time. Only when a certain operation categorized by one of the program codes is desired, should it appear, being supplied by the user.
## 4. CODE DESCRIPTION & APPLICATION

### 4.A. EDITOR PROGRAM CODES

<table>
<thead>
<tr>
<th>CODES</th>
<th>MEANING</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>CENTER &amp; UNDERLINE THE LINE</td>
</tr>
<tr>
<td>C</td>
<td>CENTER THE LINE</td>
</tr>
<tr>
<td>F</td>
<td>LAST LINE ON THE LAST PAGE OF THE REPORT (A NEW REPORT MAY BE PLACED IMMEDIATELY AFTER THIS CARD)</td>
</tr>
<tr>
<td>L</td>
<td>LAST LINE ON A PAGE</td>
</tr>
<tr>
<td>S</td>
<td>SELECTIVE UNDERLINING - UNDERSCORES ONLY (BUT ALL OF) THE CHARACTERS FOUND ON THE CARD.</td>
</tr>
<tr>
<td>U</td>
<td>UNDERLINES EVERYTHING BETWEEN THE LEFTMOST CHARACTER AND THE RIGHTMOST CHARACTER OF THE LINE</td>
</tr>
<tr>
<td>V</td>
<td>THE FIRST 'V' PLACES THE PROGRAM IN THE 'V-MODE'. THE SECOND 'V' REMOVES IT FROM THE 'V-MODE'. THE THIRD 'V' PLACES THE PROGRAM BACK INTO THE 'V-MODE' AGAIN, ETC.. WHEN IN THE V-MODE, THE PROGRAM WILL LIST ALL 30 COLUMNS OF CHARACTERS AND SINGLE SPACE ALL LINES. ALL CODES FOUND IN COLUMN 1 ARE DISREGARDED AS COMMANDS BUT ARE NEVERTHELESS LISTED. THE ONLY EXCEPTION TO THE RULE IS THE 'V'-CODE ITSELF. A 'V' WILL NEVER BE LISTED IN COLUMN 1 OF THE LISTING WHETHER IN THE 'V-MODE' OR NOT. HOWEVER, PAGE NUMBERING STILL CONTINUES.</td>
</tr>
</tbody>
</table>
**CODES**

**MEANING**

**DIGIT**

Any odd digit between 1 and 9 will cause the program to skip \((\text{DIGIT}+1)\) lines before printing the information found on that card.

**DIGIT**

Any even digit between 2 and 10 will cause the program to skip \((\text{DIGIT}+1)\) lines after printing the information found on that card.

**V**

Allows single spacing (even though double spacing normally occurs - it is overridden in this case).

**R**

Roman numeral numbering is normally (automatically) initiated by the program starting with the second page of the report. 'R' denotes the last line of the last page for Roman numerals. If no 'R' appears, the program will begin page 1 after page XVII. A second 'R' in the same report will be disregarded by the program. When the input data is preceded by an 'R' code, program control skips Roman numeral numbering and employs numerical numbering only, again starting with the second page of the report.
THE FOLLOWING ARE SOME POINTERS TO REMEMBER CONCERNING THE
USE OF 'EDITOR'.

1) NOT ALL CARDS REQUIRE A PROGRAM CODE.
2) REQUIRED CODES ARE TO BE PUNCHED IN COLUMN 1 ONLY.
3) A 'V' IN COLUMN 1 IS NEVER PRINTED ON THE OUTPUT LISTING
   REGARDLESS OF WHETHER THE PROGRAM IS IN THE 'V-MODE' OR NOT.
4) ONE PUNCHED CARD PRODUCES ONE LINE OF OUTPUT.
5) ROMAN NUMERAL NUMBERING CANNOT EXCEED XVII.
6) WITH PROPER USE OF THE 'DIGIT' & 'IN' CODES ANY NUMBER OF
   LINES CAN FOLLOW OR PRECEDE A LINE OF OUTPUT.
7) IF THE USER PUNCHES LINE INFORMATION BETWEEN COLUMNS 7 TO
   72 INCLUSIVELY, PROPER PAGE ALIGNMENT WILL OCCUR. WRITING
   BEYOND COLUMN 72 COULD NECESSITATE THE NEED FOR PHOTOGRAPHIC
   REDUCTION IN ORDER FOR IT TO FALL WITHIN THE 4 X 10-1/2
   INCH LIMITS.
8) UNLESS SPECIFIED BY A PROGRAM CODE, DOUBLE SPACING NORMALLY
   OCCURS.
4.A. INDIVIDUAL SAMPLE CODE ILLUSTRATIONS

1. CODE - "A" - CENTER & UNDERLINE

CARD1

RESULT1

2. CODE - "C" - CENTERS THE LINE

CARD1

RESULT1

3. CODE - "S" - SELECTIVE UNDERLINING

CARD1

RESULT1

4. CODE - "U" - UNDERLINE

CARD1

RESULT1

Page 7
5. COUF - **DIGIT** - MULTIPLE SPACING

A) ODD NUMBER

CARDS:

- FINAL CARE
- 3 SECOND CARE
- FIRST CARE

RESULT:

- FIRST CARE
- \((3+1)\) OR 4 SPACES
- SECOND CARE
- FINAL CARE

B) EVEN NUMBER

CARDS:

- FINAL CARE
- 4 SECOND CARE
- FIRST CARE

RESULT:

- FIRST CARE
- SECOND CARE
- \((4+1)\) OR 5 SPACES
- FINAL CARE

PAGE 3
A. B. INDEX SAMPLE CODE ILLUS. [CBM1]

6. CODE - "V" - VERBATIM RCCE

CARDS:

4  EAC 'V' RCCE
5  B17CELC //*
6  2LZ 3DBFFL TXY
7  REGIA 'V' RCCE

RESULT:

REGIA 'V' RCCE
2LZ 3DBFFL TXY
B17CELC //*
END 'V' RCCE

7. CODE - "N" - NC SKIP - SINGLE SPACE

CARDS:

NC SKIP EXAMPLE

RESULT:

SINGLE SPACE
NC SKIP EXAMPLE

6. CODE - "F" - LAST LINE OF LAST PAGE OF REPORT

CARD:

F (LINE DATA PRINTED IF PRESENT)

RESULT:

REINITIALIZES PAGE NUMBERING SEQUENCE

PAGE 9
9. CODE - "R" - LAST PAGE OF ROMAN NUMERALS

CARD:

```
R  (LINE DATA PRINTED IF PRESENT)
```

RESULT:

THE PAGE FOLLOWING BEGINS PAGE NUMBERING WITH 1

10. CODE - "L" - AN EXAMPLE OF THE "L" CODE (LAST LINE ON PAGE) CAN BE FOUND IN SECTION 4.C.
I. THE FOLLOWING IS THE LISTING OF THE TABLE OF CONTENTS OF THE
ACTUAL PROGRAM CODED DECK USED FOR THE DOCUMENTATION OF THIS REPORT.
IT WAS LISTED UNDER THE "V-MODE".

<table>
<thead>
<tr>
<th>SECTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Abstract</td>
<td>1</td>
</tr>
<tr>
<td>2. Introduction</td>
<td>2</td>
</tr>
<tr>
<td>3. General Description</td>
<td>3</td>
</tr>
<tr>
<td>4. Code Description &amp; Application</td>
<td>5</td>
</tr>
<tr>
<td>5. Editor Program Codes</td>
<td>5</td>
</tr>
<tr>
<td>6. Individual Sample Code Illustrations</td>
<td>7</td>
</tr>
<tr>
<td>7. Illustrative Examples</td>
<td>11</td>
</tr>
<tr>
<td>8. Appendices</td>
<td>15</td>
</tr>
<tr>
<td>9. Flow Chart of Basic Program Logic</td>
<td>16</td>
</tr>
<tr>
<td>10: Listing of Fortran IV Source Program</td>
<td>21</td>
</tr>
<tr>
<td>11. Table of Distribution</td>
<td>26</td>
</tr>
</tbody>
</table>
2. The following is the listing of Section 4.8 of the actual program used for the documentation of this report. It was listed under the 'V-mode'.

4.8. Individual Sample Code Illustrations

1. CODE: "**" - Center & Underline
   
   CARD: A CENTER & UNDERLINE EXAMPLE
   
   RESULT: CENTER & UNDERLINE EXAMPLE

2. CODE: "**" - Centers the line
   
   CARD: C CENTER EXAMPLE
   
   RESULT: CENTER EXAMPLE

3. CODE: "**" - Selective Underlining
   
   CARD: S SELECTIVE UNDERLINING EXAMPLE
   
   RESULT: SELECTIVE UNDERLINING EXAMPLE

4. CODE: **" " "Underline
   
   CARD: U UNDERLINE EXAMPLE

Page 17
RESULT: UNDEFINED EXAMPLE

4. INDIV. SAMPLE CODE ILLUS. (CONT.)

5. CODE - "DIGIT" - MULTIPLE SPACING AT ONE NUMBER
CARS:

FIRST CARD
SECOND CARD
FIRST CARD

RESULT:

FIRST CARD
SECOND CARD
FINAL CARD

6. EVEN NUMBER
CARS:

FIRST CARD
SECOND CARD
FIRST CARD

RESULT:

FIRST CARD
SECOND CARD
FINAL CARD

7. INDIV. SAMPLE CODE ILLUS. (CONT.)

8. CODE - "DIGIT" - VERTICAL MODEL
CARS:

5 END "DIGIT"
"DIGIT"
A "DIGIT"
V BEGIN "DIGIT"

RESULT:
4.6. ILLUSTRATIVE EXAMPLES (CONT.)

BEGIN 'Y' MODE
2LZ 30FAPF TXY
BITCOLD /#
END 'Y' MODE

7. CODE = **N** - NO SKIP - SINGLE SPACE
   CARD:
   NO SKIP EXAMPLE
   SINGLE SPACE
   RESULT:
   SINGLE SPACE
   NO SKIP EXAMPLE

8. CODE = **F** - LAST LINE OF LAST PAGE OF REPORT
   CARD:
   F  (LINE DATA PRINTED IF PRESENT)
   RESULT:
   REINITIALIZES PAGE NUMBERING SEQUENCE

9. CODE = *** -- LAST PAGE OF HUMAN MINERALS
   CARD:
   ...  (LINE DATA PRINTED IF PRESENT)
   RESULT:
   THE PAGE FOLLOWING BEGINS PAGE NUMBERING WITH 1

10. CODE = **U** - AN EXAMPLE OF THE *U* CODE (LAST LINE ON
    PAGE 1 CAN BE FOUND IN SECTION 4.6.)
5. APPENDICES
5.4. FLOW CHART OF BASIC PROGRAM LOGIC
READ DATA CARD

CHECK NUMERICAL DIGIT CODES FOR LINE SKIPPING

SKIP REQUIRED NUMBER OF LINES

CHECK FOR REGULAR PROGRAM CODES

USE THE APPROPRIATE FOLLOWING SUBROUTINES REQUIRED BY PROGRAM CODES

SELECT LIMIT ULINE CENTER SPLINE PAGEEND

END OF DATA?

STOP

FLOW CHART OF BASIC LOGIC

PAGE 17
SUBROUTINE SELECT

NOCHARACTER

TEST CHARACTERS

BETA(I) = ' ' (BLANKS)

CHARACTER

BETA(I) = ' - ' (UNDERLINE)

RETURN

SUBROUTINE LIMIT

DETERMINE FIRST CHARACTER POSITION

DETERMINE LAST CHARACTER POSITION

RETURN

PAGE 18
SUBROUTINE PAGE#0

ROMAN NUMERALS

TEST
PAGE NUMBER

ARABIC NUMERALS

WHITE
PAGE NUMBER

WHITE
PAGE NUMBER

PAGE INCREMENT

CAUSE PRINTER
TO MOVE TO TOP
OF NEW PAGE

RETURN

PAGE 20
2. B. LISTING OF FORTRAN IV SOURCE PROGRAM

C EDITOR PROGRAM, SECOND VERSION; SEPTEMBER 1969

DIMENSION ALPHA(80), BETA(80), ROMAN(20)

DATA C/"RY", U/"RY", A/"AY", X/"X"/ NEWLINE, P/"P"/

1 PAGE/"PAGE"/, X1/"1"/, X2/"2"/, X3/"3"/, X4/"4"/, X5/"5"/

2 X6/"6"/, X7/"7"/, X8/"8"/, X9/"9"/, NEWLINE, SPACE/" "/,

3,5/"S"/

20 LMAN=0

IA=0
LINE=1

WRITEx(6,3)

WRITE(6,3)

FORMAT(11x/)

FORMAT(11x/)

1 READ(5,2,END=50) X, ALPHA(I), I=1,79

2 FORMAT(90A1)

3 FORMAT(17X,9A1,/) /

w FORMAT(17X,9A1,/) /

4...format...

IF(XC.EQ.0) IA=IA+1

W=IA/2

IX=IA-2*W

IF(IX.NE.1) GO TO 15

IF(XC.EQ.X1) SPACE

BETA(I)=ALPHA(I)

2 CONTINUE

WRITE(6,23) (BETA(I), I=1,80)

3 FORMAT(17X,80A1)

LINES=LINES+1

GO TO 17

15 LSKIP=0

IF(XC.EQ.X1) LSKIP=1

IF(XC.EQ.X2) LSKIP=2

IF(XC.EQ.X3) LSKIP=3

IF(XC.EQ.X4) LSKIP=4

IF(XC.EQ.X5) LSKIP=5

IF(XC.EQ.X6) LSKIP=6

IF(XC.EQ.X7) LSKIP=7

IF(XC.EQ.X8) LSKIP=8

IF(XC.EQ.X9) LSKIP=9

LSKIP=LSKIP/Z

LSKIP=LSKIP-2=LSKIP1

IF(LSKIR) 9,9,10

DO 11 I=1,LSKIP

CALL SPLINE(LINE)

IF(LINE.EQ.50) CALL PAGENO(LROMAN,LINE)

PAGE 21
11 CONTINUE
9 IF(XC.EQ.C) CALL LIMIT(ALPHA,NR,NL)
   IF(XC.EQ.U) CALL LIMIT(ALPHA,NR,NL)
   IF(XC.EQ.A) CALL LIMIT(ALPHA,NR,NL)
   IF(XC.EQ.C) CALL CENTER(ALPHA,NR,NL)
   IF(XC.EQ.A) CALL CENTER(ALPHA,NR,NL)
   IF(XC.EQ.A) CALL LIMIT(ALPHA,NR,NL)
   IF(XC.EQ.U) CALL ULINE(BETA,NR,NL)
   IF(XC.EQ.A) GO TO 6
   IF(XC.EQ.A) GO TO 6
   IF(XC.EQ.XN) GO TO 6
   IF(XC.EQ.XS) CALL SELECT(ALPHA,BETA)
   IF(XC.EQ.XS) GO TO 0
5 WRITE(6,3)(ALPHA(I),I=1,79)
   LINE=LINE+2
   GO TO 12
6 WRITE(6,4)(ALPHA(I),I=1,79)
   LINE=LINE+1
   IF(XC.EQ.U) WRITE(6,7)(BETA(I),I=1,79)
   IF(XC.EQ.V) WRITE(6,7)(BETA(I),I=1,79)
   IF(XC.EQ.S) LINE=LINE+1
   IF(XC.EQ.S) LINE=LINE+1
   IF(XC.EQ.S) LINE=LINE+1
   IF(XC.EQ.S) GO TO 0
7 FORMAT(1H+,16X,79A1,7I)
12 IF(LSKIP.EQ.0) GO TO 13
   IF(LSKIP)13,21,13
21 DO 14 I=1,LSKIP
   CALL SPLINE(LINE)
   IF(LINE.EQ.50) CALL PAGENDIROMAN,LINE)
14 CONTINUE
13 IF(XC.EQ.R) GO TO 16
   IF(XC.EQ.R) GO TO 16
   IF(XC.EQ.Y) GO TO 16
17 IF(LINE.EQ.49) CALL SPLINE(LINE)
   IF(LINE.EQ.50) CALL PAGENDIROMAN,LINE)
   GO TO 19
16 IF(LINE.EQ.50) GO TO 17
   LINE=LINE+1
   DO 18 I=LINE+49
   CALL SPLINE(LINE)
18 CONTINUE
   GO TO 17
19 IF(XC.EQ.F) GO TO 20
   IF(LMAN.EQ.1) GO TO 1
   IF(XC.EQ.R) LMAN=1
   IF(XC.EQ.R) LROMAN=18
GO TO 1
50 STOP
END

SUBROUTINE SELECT(ALPHA, BETA)
DIMENSION ALPHA(80), BETA(80)
DATA RANK/' ', HANK/' '/
DO 1 I = 1, 79
1 IF(ALPHA(I).EQ.RANK) BETA(I)=RANK
IF(ALPHA(I).NE.RANK) BETA(I)=HANK
CONTINUE
RETURN
END

SUBROUTINE LIMIT(ALPHA, NR, NL)
DIMENSION ALPHA(80)
DATA BLANK/' '/
NL=0
NR=0
DO 1 I = 1, 79
1 IF(BLANK.NE.ALPHA(I)) GO TO 2
CONTINUE
RETURN
DO 4 I = 1, 79
J=80-I
4 IF(BLANK.NE.ALPHA(J)) GO TO 5
CONTINUE
GO TO 3
NR=J
GO TO 3
END

PAGE 23
SUBROUTINE ULINE(BETA, NR, NL)
DIMENSION BETA(80)
DATA BANK/* , /
DATA CRANK/* , /
NL = NL-1
NR = NR+1
IF(NL .EQ. 0) GO TO 2
DO 1 I = 1, NL
BETA(I) = BANK
1 CONTINUE
2 DO 3 I = NL, NR
BETA(I) = CRANK
3 CONTINUE
IF(NR .EQ. 80) GO TO 4
DO 5 I = NR+1, 79
BETA(I) = BANK
5 CONTINUE
4 RETURN
END

SUBROUTINE CENTER(ALPHA, NR, NL)
DIMENSION ALPHA(80), GAMMA(80)
DATA TANK/* , /
DO 8 I = 1, 79
GAMMA(I) = ALPHA(I)
8 CONTINUE
NA = NR - NL + 1
NB = (79-NA)/2
NC = NB - NL
IF(NC) 1, 2 3
2 RETURN
3 J = 79 - NC
DO 6 I = 1, NC
ALPHA(I) = TANK
6 CONTINUE
DO 5 I = 1, J
NE = NC + I
ALPHA(NE) = GAMMA(I)
5 CONTINUE
GO TO 2
1 NC = NC
J = 79 - NC
DO 9 I = 1, J
9 CONTINUE
NE=NE+1
ALPHA(I)=GAMMA(NE)
9 CONTINUE
J=J+1
DO 10 I=J,79
ALPHA(I)=TANK
10 CONTINUE
GO TO 2
END

SUBROUTINE SPLINE(LINE)
WRITE(6,1)
1 FORMAT(1H0)
LINE=LINE+1
RETURN
END

SUBROUTINE PAGENO(LROMAN,LINE)
DIMENSION ROMAN(20)
DATA PAGE/"PAGE"/
DATA ROMAN/'I','II','III','IV','V','VI','VII','VIII','IX','X','XI','XII','XIII','XIV','XV','XVI','
"XVII"/
MAN=LROMAN-17
1 FORMAT(1H0)
IF(LROMAN.EQ.0) GO TO 5
WRITE(6,1)
IF(LROMAN.LT.18)WRITE(6,2)PAGE,ROMAN(LROMAN)
2 FORMAT(5X,A4,1X,A4)
IF(LROMAN.GE.18) WRITE(6,3) PAGE,MAN
3 FORMAT(5X,A4,1X)
5 LROMAN=LROMAN+1
LINE=1
WRITE(6,4)
4 FORMAT(1H/////)
RETURN
END
This report describes the application of an editing program called "EDITOR" for documentation purposes.

This program was written in FORTRAN IV Language for the IBM 360/65 Digital Computer and requires a total memory length of 2,582 bytes.

The "EDITOR" Program permits the user to have his reports "typed" on an IBM 1403 highspeed printer. All text editing instructions (line spacing, underlining, centering, page numbering, etc...) are passed to the "EDITOR" Program by means of Alpha Numerical codes punched in Column 1 of the input data cards. The data cards contain the text of the report, having been punched from FORTRAN coding sheets upon which the author has "written" his report.
<table>
<thead>
<tr>
<th>KEY WORDS</th>
<th>LINK A</th>
<th>LINK B</th>
<th>LINK C</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDITOR</td>
<td>ROLE</td>
<td>ROLE</td>
<td>ROLE</td>
</tr>
<tr>
<td>Documentation</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
</tr>
<tr>
<td>Computer Generated Text</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
</tr>
</tbody>
</table>
EDITOR MODIFICATION
Technical Memorandum #1871
Modified: 31 March 1969

CONTENTS: (1) "P" Code Description - Add to Section 4.
"Code Description and Application"

(2) Program Listing Replacement -
Replaces old TM 1871 Listing of Section 5. B.
"Listing of FORTRAN Source Program."

WILLIAM H. BOLTE
SEAD/DPSO Bldg. 351
Picatinny Arsenal
Ext. 3663
PROGRAM: "EDITOR" - A computer program for documentation purposes (T.M. 1871)

MODIFIED: March 1969

MODIFICATION: Added "P" Code

PURPOS' OF CODE: The "P" code allows one to skip a certain specified number of pages. This number is a user option.

INSTRUCTIONS FOR "P" CODE USAGE:

Punch the letter "P" in Column 1 of the card to be used, followed by an integer number. This number specifies the number of pages to be skipped and has an allowable field width of two (2); therefore, each data item must be right justified in its field, since leading and trailing blanks are treated as zeros (0's).

i.e.,

\[ \text{Col 1:2:3} \]

NOTE: Any information punched after column three (3) is ignored and hence not printed during output.

ILLUSTRATIVE USE:

CODE - "P" - Multiple Page Skip

CARDS:

\[ \begin{array}{c}
\text{Printing on Card #3} \\
\text{P 2} \\
L \text{End of this page}
\end{array} \]

RESULT:

END OF THIS PAGE
(Skips two (2) pages)

PRINTING ON CARD #3
(In this case it is printed on top of third (3rd) page)

COMMENT:

This modification sheet is to be used in accordance with the Technical Memorandum #1871. Any problems, questions, inquiries or suggestions regarding the use of this code or any other subject matter from the T.M. 1871 should be directed to Mr. William H. Holte, SEAD/DFSO, Building 351, Ext. 2663.
C

EDIT1R PROGRAM LISTING REVISION 31 MAR 69
C
C REPLACES TM1871 PROGRAM LISTING

C EDITTR PROGRAM, FOURTH VERSION, MARCH 1969
C TM=1871 IS STILL TO BE USED AS REFERENCE
C
DIMENSION ALPH(A(80)), BETA(A(80)), ROMAN(A(20))
C
DATA C/*C*/ , U/*U*/ , A/*A*/ , XN/*XN*/ , R/*R*/ , XL/*XL*/ , P/*P*/ , PAGE/*PAGE*/
C X/*X*/ , X2/*X2*/ , X3/*X3*/ , X4/*X4*/ , X5/*X5*/ , X6/*X6*/ , X7/*X7*/ , X8/*X8*/
C X9/*X9*/ , V/*V*/ , SPACE/*SPACE*/ , F/*F*/ , S/*S*/
C
LINE=0
IA=0
LINE=1
LINE=IA

2 READ(5,23,END=20) XC, (ALPHA(I), I=1,79)
IF (XC.EQ.V) IA=IA+1
IA=IA/2
IF (IX*.NE.1) GO TO 4
IF (XC.EQ.V) XC=SPACE
BETA(IA)=XC
DD 3 IA=IA+1
BETA(IA+1)=ALPHA(I)
CONTINUE
WRITE (6,26) BETA(I), I=1,80
LINE=LINE+1
GO TO 13

3 CONTINUE
WRITE (6,26) (BETA(I), I=1,80)
LINE=LINE+1
GO TO 13

4 LSKIP=0
IF (XC.EQ.X1) LSKIP=1
IF (XC.EQ.X2) LSKIP=2
IF (XC.EQ.X3) LSKIP=3
IF (XC.EQ.X4) LSKIP=4
IF (XC.EQ.X5) LSKIP=5
IF (XC.EQ.X6) LSKIP=6
IF (XC.EQ.X7) LSKIP=7
IF (XC.EQ.X8) LSKIP=8
IF (XC.EQ.X9) LSKIP=9
LSKIP=LSKIP/2
LSKIP=LSKIP/2
IF (LSKIP) 7,7,5
DO 6 I=1,LSKIP

PAGE 21
CALL SPLINE (LINE)

IF (LINE.EQ.50) CALL PAGE_END (LROMAN, LINE)

6 CONTINUE

7 IF (XC.EQ.) CALL LIMIT (ALPHA, NR, NL)
IF (XC.EQ.) CALL LIMIT (ALPHA, NR, NL)
IF (XC.EQ.) CALL LIMIT (ALPHA, NR, NL)
IF (XC.EQ.) CALL CENTER (ALPHA, NR, NL)
IF (XC.EQ.) CALL CENTER (ALPHA, NR, NL)
IF (XC.EQ.) CALL LIMIT (ALPHA, NR, NL)
IF (XC.EQ.) CALL LIMIT (ALPHA, NR, NL)

A1900

A1950

A2000

A2050

A2100

A2150

A2200

A2250

A2300

A2350

A2400

A2450

A2500

A2550

A2600

A2650

A2700

A2750

A2800

A2850

A2900

A2950

A3000

A3050

A3100

A3150

A3200

A3250

A3300

A3350

A3400

A3450

A3500

A3550

A3600

A3650

A3700

A3750

A3800

A3850

A3900

A3950

A4000

A4050

A4100

A4150

A4200

A4250
IF (XC.EQ.R) LMAN=1
IF (XC.EQ.R) LROMAN=1
GO TO 2

17 AA=ALPHA(1)
AB=ALPHA(2)
MA=0
IF (AA.EQ.X1) MA=1
IF (AA.EQ.X2) MA=2
IF (AA.EQ.X3) MA=3
IF (AA.EQ.X4) MA=4
IF (AA.EQ.X5) MA=5
IF (AA.EQ.X6) MA=6
IF (AA.EQ.X7) MA=7
IF (AA.EQ.X8) MA=8
IF (AA.EQ.X9) MA=9
MR=0
IF (AB.EQ.X1) MB=1
IF (AB.EQ.X2) MB=2
IF (AB.EQ.X3) MB=3
IF (AB.EQ.X4) MB=4
IF (AB.EQ.X5) MB=5
IF (AB.EQ.X6) MB=6
IF (AB.EQ.X7) MB=7
IF (AB.EQ.X8) MB=8
IF (AB.EQ.X9) MB=9
MC=10*MA+MB
IF (LINE.EQ.1) GO TO 19
IF (LINE.LE.50) CALL SPLINE (LINE)
IF (LINE.LE.50) GO TO 13
IF (LINE.LE.50) CALL PAGENO (LROMAN,LINE)
13 D) 21 I=1,MC
D) 20 J=1,49
20 CALL SPLINE (LINE)
CALL PAGENO (LRoman,LINE)
21 CONTINUE
GO TO 2
28 STOP
C)
22 FORMAT (11//) 1)
23 FORMAT (80X)
24 FORMAT (17X,79A1,//)
25 FORMAT (17X,79A1)=
26 FORMAT (17X,80A1)
27 FORMAT (1H+,16X,79A1,//
END

PAGE 23
SUBROUTINE SELECT (ALPHA, BETA)
DIMENSION ALPHA(80), BETA(80)
DATA RANK/, HANK/  
DO 1 I = 1, 79
IF (ALPHA(I) .EQ. RANK) BETA(I) = RANK
IF (ALPHA(I) .NE. RANK) BETA(I) = HANK
1 CONTINUE
RETURN
END

SUBROUTINE LIMIT (ALPHA, NR, NL)
DIMENSION ALPHA(80)
DATA BLANK/' '/
NL = 0
NR = 0
DO 1 I = 1, 79
IF (BLANK .NE. ALPHA(I)) GO TO 3
1 CONTINUE
RETURN
2 NL = I
3 DO 4 J = 1, 79
4 IF (BLANK .NE. ALPHA(J)) GO TO 5
CONTINUE
GO TO 2
4 NL = J
5 GO TO 2
END

SUBROUTINE JLINE (BETA, WK, NL)
DIMENSION BETA(80)
DATA RANK/, CRANK/  
NLM = NL - 1
ILM = WK + 1
IF (NL .LE. 0) GO TO 1
1 IL = I + 1, NL
BETA(I) = RANK
CONTINUE
2 IL = I + 1, NL
BETA(I) = CRANK
CONTINUE
IF (NRP.EQ.80)
D) 4 I=NRP,79
BETA(I) = BANK
CONTINUE
RETURN
END

SUBROUTINE CENTER (ALPHA,NR,NL)
DIMENSION ALPHA(90), GAMMA(80)
DATA TANK/I0/
D1 5 I=1,79
GAMMA(I)=ALPHA(I)
CONTINUE
NA=NK-NL+1
NL=(79-NA)/2
NC=NH-NL
IF (NC) 6,2,3
RETURN
J=79-NC
D1 4 I=1,NC
ALPHA(I)=TANK
CONTINUE
D1 5 I=1,J
NC=NC+1
ALPHA(NC)=GAMMA(I)
CONTINUE
GO TO 2
NC=NC
J=79-NC
D1 7 I=1,J
NC=NC+1
ALPHA(I)=GAMMA(NC)
CONTINUE
J=J+1
D1 8 I=1,79
ALPHA(I)=TANK
CONTINUE
GO TO 2
END

SUBROUTINE SPLINE (LINE)
SUBROUTINE PAGENJ (ROMAN, LINE)
DIMENSION ROMAN(70)
DATA PAGE/'PAGE'/'
DATA ROMAN/'X', 'I', 'II', 'III', 'IV', 'V', 'VI', 'VII', 'VIII', 'IX', 'X', 'XI', 'XII', 'XIII', 'XIV', 'XV', 'XVI', 'XVII'/'
MAN=ROMAN(17)
IF (LROMAN.EQ.0) GO TO 1
WRITE (6,2)
IF (LROMAN.LT.18) WRITE (6,3) PAGE, ROMAN(LROMAN)
IF (LROMAN.GE.18) WRITE (6,4) PAGE, MAN
1 LROMAN=LROMAN+1
LINE=1
WRITE (6,5)
RETURN
C
2 FORMAT (1HD)
3 FORMAT (50X,34,1X,5X)
4 FORMAT (50X,4,14)
5 FORMAT (*1/////)
END