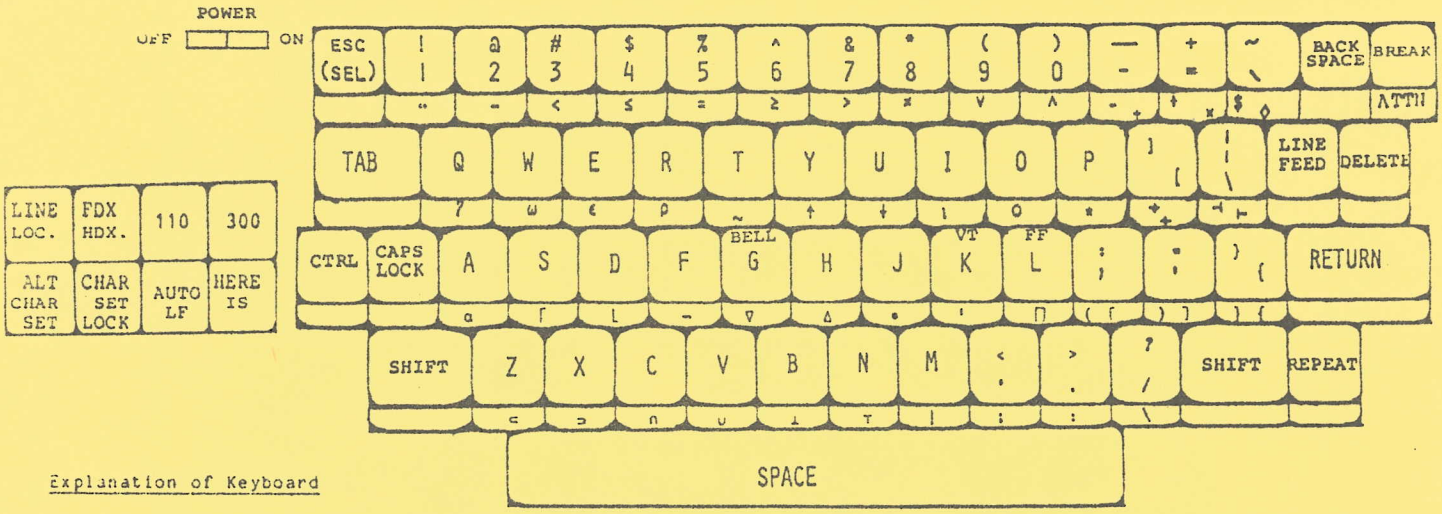
DECsystem-10

Introduction to Timesharing

SYRACUSE UNIVERSITY
COMPUTING CENTER
machinery hall

DECsystem-10 SIGN-ON/OFF



Explanation of Keyboard

The tops of the keys show a full ASCII character set. The yellow characters on the key fronts are the APL special characters. The DECsystem-10 uses only the ASCII character set, so the yellow key fronts should be ignored. Of special interest to the DECsystem-10 user are the ESC key (ESCape, also called altmode), the BACKSPACE key (equivalent to RUBOUT for everything but DDT, where DELETE should be used), the TAB key, the LINE FEED key, and the CTRL (ConTRoL) key. CAPS LOCK is not the same as shift. It only affects the sending of alphabetic characters. Do not use the BREAK and REPEAT keys.

To Sign on and to Sign off the DECsystem-10

1. Check all eight buttons at the left of the keyboard. Only the one marked 300 should be down. The other seven should be up. (To change the state of a button, press it once.)

2. Press ON (upper left of keyboard). The red light marked STD CHAR SET will light. (This is the ASCII character set.)

3. Press RETURN button. The computer will print a one-line response followed by a slash (prompt) at the left margin.

4. After the slash prints, type LOGIN followed by a space and your account number. For example:

   .LOGIN 7000,12

   Press RETURN after typing your account number. The DECsystem-10 monitor will respond by printing any special system messages. Then, on a new line, the system will print the word PASSWORD: .

5. You may now enter your password. Note that your password does not print at the terminal. Printing at the terminal is turned off in order to help you to keep your password secret. If the LOGIN is successful, a message of the day will print, followed by a period (prompt) at the left margin. You may now run any user programs on the DECsystem-10.

6. To sign off from the DECsystem-10, enter the command KJOB. The monitor will respond by printing the word CONFIRM: . If you wish to save all files, you should respond by typing the letter F. When H (for Help) is entered instead of F, the system will print out descriptions of other acceptable responses to the word CONFIRM. After the charge information and exit messages have printed, you may press the OFF switch and leave the terminal; or you may sign on again to any timesharing system such as APL on the 370/155, or the DECsystem-10.

Sample Sign-on/Sign-off

For the purpose of illustration, assume your LOGIN number is 7000,12 and your password is YO. The printed lines shown below are similar to the lines of print that you would see at the terminal if you signed on with this account number and password. Note that the password is not printed.

```
072) SU TIMESHARING 10.33.06 FRI 08/04/78 V1M28
/LOGIN 7000,12
LOGIN 7000,12
JOB 15 SYRACUSE UNIVERSITY 6.03 TTY6
Password:

1037    04-Aug-78       Fri
```

Sample Sign-off

```
.KJOB
Confirm: F
Job 15, User [7000,12] Logged off TTY6      1036  4-Aug-78
Saved all files (65 blocks)
Runtime 0.66 Sec

CHARGE:$0.10+$0.03(C)+$0.05(F)+$0.02(K)
Y-T-D ACCOUNT USE:13%
```

Warning

DO NOT PULL THE PAPER FROM THE TOP. Turn the carriage up by hand. If the terminal does not have paper in the feed, call the Computing Center for help - ext. 3-3245.

SYRACUSE UNIVERSITY COMPUTING CENTER

INFORMATION SERIES

DOCUMENT	TITLE
GI25-1107	DECsystem-10 Glossary
DATE	
12/07/76	
AUTHOR	John Thornton

CLASSIFICATION: General Information

AUDIENCE: DECsystem-10 Users

LEVEL: Introductory

DISTRIBUTION: AID

REFERENCES: Supercedes CCIS Memo P27-1079 titled as above.

ABSTRACT:

This memo is intended to explain several of the computing words and concepts that can confuse novice users. It is not a glossary in the usual sense, but 4+ pages of prose with several words underlined. The underlined words are explained in context. They are also indexed at the end of the memo.

4 Pages

$0.00

This memo is intended to explain several words and concepts relating to the DECsystem-10 computer that often confuse or mystify the novice computer user. The words that are explained are underlined in the text of this memo. The words are also listed alphabetically at the end of the memo with the line number where each is explained.

PARTS OF THE COMPUTER THAT WE NEED

The Syracuse University DECsystem-10 computer has nicknames of course, so don't be confused if someone refers to the DEC-10 or to the 10. The DEC-10 is made up of several parts. The most important part is called the Central Processing Unit or CPU. This is the unit which controls the interpretation and execution of instructions. That is, the CPU does the actual calculations, manipulations, etc. The CPU needs a "scratch pad" to make notes on while it does its work. This storage is called core storage or memory.

The other parts of the computer are called peripheral devices. These devices are used for input and output (I/O) and for storage of information. The input devices are the card reader, a device that reads the information stored on punched cards, and a terminal, a personal typewriter-like device that can be used for both input and output. Output can also be sent to the line printer, a high speed printing device which all users can send information to for printing.

Since core storage is limited, we need a place to store large amounts of information for future use. The storage devices are all made of magnetic materials that can be magnetized in such a way that they represent information. A disk is named for its disc shape; it is always connected to the computer. Mastape is magnetic recording tape that can store very large amounts of information, but a tape must be mounted onto a tape drive whenever it is to be used. DECtape is a special form of magnetic tape designed especially for DECsystem machines. All of the physical parts of the computer are collectively referred to as the computer's hardware.

STORING INFORMATION

Each piece of information fed into the computer consists of a number or a character. The characters used are the letters (A-Z), the digits (0-9) and the special symbols -, /, *, $, ., (,) and +. Any information combining these characters is called alphanumeric data. Alphanumeric data is represented in the computer or in storage in a special coded form. The code used by the DEC-10 is called ASCII, "American Standard Code for Information Interchange."

There are several terms used for measuring amounts of core storage. A bit is the smallest unit of memory. It can store only a zero or a one, i.e. on or off. The number of bits needed to represent one alphanumeric character is called a byte.

Another measure commonly used is the amount of memory needed to represent a number, called a word of memory. On the DEC-10, a byte consists of 7 bits and a word contains 36 bits, i.e. 5 bytes + 1 extra bit. The only other measure we need is a large one. K is an abbreviation for 1024 words of memory, i.e. 4K means 4096 words of core.

TALKING ABOUT INFORMATION

We now need some terms for discussing the information stored in memory (or on a peripheral device). A record is a collection of words, normally the amount of information to be input or output at one time. When we store information on tape or disk, we store it in groups of related information known as files. A file can be a data file (containing alphanumeric or numeric information to be processed later), a program file (containing a set of instructions telling the computer exactly what you want it to do) or it may be a text file (containing alphanumeric information to be written out later). The size of the file is measured in blocks, a unit consisting of 128 computer words.

GETTING ON

There are two techniques for using the computer. Batch processing allows you to turn in a program, have it run and later pick up the results. The alternative is to use a timesharing system, where you use a terminal to input programs and, without much waiting, get the results typed out at the terminal. A timesharing system is designed so that users alternate use of the CPU in rapid succession in such a way that each user seems to have complete control of the computer.

The computer is used by writing a program according to one of several systems of syntax or "grammar". These systems are called programming languages. There are several types of programming languages; the most basic is the machine's own set of instructions, generally called machine language. Since this language consists entirely of numbers, it is very confusing and some programmers prefer an assembly language (a symbolic form of the machine language). On the DEC-10, the assembly language is called MACRO. An assembly language uses abbreviations (called mnemonics) to represent instructions instead of numbers, as machine language uses. For instance, ADD in MACRO and 270 in machine language both instruct the computer to do an addition.

PROGRAMS, PROGRAMS

Other programmers prefer a higher level language, a language that is more like algebra or English. Programs written in one of these languages are often called source programs. These programs must be compiled, translated into machine language, before they can be run. A program that actually does this translation is called a compiler. Each higher level language has its own compiler. The compiled version of a source

"waiting line", a temporary storage location for files to await processing. This waiting line is called the queue or the spooling mechanism.

Hopefully, the terms in this memo will have a little more meaning now. More information is available for beginning users of the Syracuse University Computing Center (SUCC). Many SUCC memos are for sale in the Business Office, Room 122 Machinery Hall. Others are available free of charge in the AID Office, room 116 Machinery Hall (AID = Assistance, Information, Documentation). Help with computing problems is also available in the AID Office.

INDEX

program is called the object program or the relocatable binary program.

Each programming language uses different words or symbols to represent the same instruction. A method for solving a particular problem that is general enough to be translated into any of the programming languages is called an algorithm. A program that is stored on the computer and made available to all the users is often called a system program. Some system programs can aid you in manipulating, correcting or analyzing your files and programs. These are called utilities. The entire collection of system programs, utilities and compilers is generally referred to as the computer's software.

MURPHY'S LAW

When programming a computer, you must remember Murphy's Law which states that if anything can go wrong with your program, it probably will go wrong. When this occurs, you must sit down and test the program, study the program and agonize over the program until the reasons for error are discovered. This process is called debugging.

FILES, ETC.

In order for you to refer to a particular file, you need a method of naming it. On the DEC-10 a file specification is made up of two parts: a user-created filename and a "last name" called its extension. The extension indicates the type of file, whether it is a data file or a text file, which programming language it was written in, etc. Since there will be files that you don't mind others using and other files that you don't want others to use, the DEC-10 allows you to protect your files. The protection of a particular file is a three digit number which indicates to the computer which users you will allow to use the file.

A FEW FINAL NOTES

The DEC-10 has a system program called the monitor which controls the communication between you and the computer. The monitor fetches system programs and starts them for you. When you wish to begin working at a terminal, you must identify yourself to the monitor using your account number, also called your project, programmer numbers or PPN. This process is referred to as logging in.

Once you have logged in, the monitor prints a prompt (a period) which indicates that it is ready to accept a command. You may now use one of about 110 monitor commands that allow you to create files, run programs or ask for print-out. In many monitor commands, you are allowed to omit certain parts of the command line. In this case, the DEC-10 has a series of defaults or assumed values for the missing information. Since users must share most of the peripheral devices, some DEC-10 devices have a

FIG S-4. CHARACTERS AVAILABLE IN S/R SYMBOL.

(SUCC VERSION FEB 76

SYMBOL NUMBER FOR INTEGER MODE CALL (DECIMAL)
EBCDIC CHARACTERS ON SYSTEM/370 (HEXADECIMAL)
ASCII CHARACTERS ON DECSYSTEM-10 (OCTAL)

0		16 10	|	32 20 173	}	48 30 26	∑	64 40 40		80 50 46	&	96 60 55	—	112 F0 60	0
1		17 11 10	BS	33 21 175	{	49 31	▫/▫	65 C1 101	A	81 D1 112	J	97 61 57	/	113 F1 61	1
2		18 12 136	∧	34 22 6	μ	50 32 30	≤	66 C2 102	B	82 D2 113	K	98 E2 123	S	114 F2 62	2
3		19 13 1	=	35 23 7	π	51 33 31	≥	67 C3 103	C	83 D3 114	L	99 E3 124	T	115 F3 63	3
4		20 14 2	→	36 24 11	ϕ	52 34	△	68 C4 104	D	84 D4 115	M	100 E4 125	U	116 F4 64	4
5		21 15 15	CR	37 25 12	⊖	53 35 133	⌈	69 C5 105	E	85 D5 116	N	101 E5 126	V	117 F5 65	5
6		22 16 3	≠	38 26 13	ψ	54 36 135	⌉	70 C6 106	F	86 D6 117	O	102 E6 127	W	118 F6 66	6
7		23 17 4	±	39 27 14	χ	55 37 134	\	71 C7 107	G	87 D7 120	P	103 E7 130	X	119 F7 67	7
8		24 18	—	40 28 20	ω	56 38	↑	72 C8 110	H	88 D8 121	Q	104 E8 131	Y	120 F8 70	8
9		25 19		41 29 21	λ	57 39	↓	73 C9 111	I	89 D9 122	R	105 E9 132	Z	121 F9 71	9
10		26 1A	=	42 2A 22	∝	58 3A	‡	74 4A 37	¢	90 5A 41	| △	106 6A 140	∞	122 7A 72	▫ ▫
11		27 1B 5	∫	43 2B 23	δ	59 3B	‡	75 4B 56	▫	91 5B 44	$	107 6B 54	▫'	123 7B 43	#
12		28 1C 32	⊃	44 2C 24	∈	60 3C 33	←	76 4C 74	<	92 5C 52	*	108 60 45	▫/ /▫	124 7C 100	@
13		29 1D 27	∨	45 2D 25	η	61 3D	×	77 4D 50	(	93 5D 51	)	109 6D 137		125 7D 47	¶
14		30 1E 176	~	46 2E 16	UC	62 3E 34	↑	78 4E 53	+	94 5E 73	▫ '	110 6E 76	>	126 7E 75	=
15 1F		31 1F 36	≈	47 2F 17	LC	63 3F 35	↓	79 4F 174	|	95 5F	¬	111 6F 77	? △	127 7F 42	¶¶

SYRACUSE UNIVERSITY COMPUTING CENTER

INFORMATION SERIES

DOCUMENT	TITLE
LF19-1193	Comparison of FORTRAN Compilers at S.U.
DATE	
08/28/78	
AUTHOR	John Thornton

AUDIENCE: FORTRAN Users

LEVEL: Intermediate

DISTRIBUTION: AID

REFERENCES: (Supersedes CCIS Memo LF19-1050, titled as above.)

ABSTRACT:

This memo discusses the differences between FORTRAN on the IBM System/370 and on the DECsystem-10. The differences involve features available in each version of FORTRAN that are not available in other versions. The intent is to illustrate problems and required changes in converting a FORTRAN program from one machine to another, and to help users decide which version of FORTRAN would be best for their purposes.

7 Pages

$0.00

Introduction

The Syracuse University Computing Center has available for users five versions of FORTRAN, two on the DECsystem-10 and three on the IBM/370. On the DEC-10 are two very similar versions, FORTRAN-10 and FORTRAN-40. FORTRAN-10 includes all of FORTRAN-40; plus it does code optimization (and therefore uses more memory). In this memo, these two FORTRANs will be referred to as DEC-10 FORTRAN unless a difference is being discussed. The IBM/370 has IBM's two FORTRANs, level-G and level-H. Level H of IBM FORTRAN includes all of level G and has code optimization features. In this memo, the name FORTRAN-G refers to both levels G and H. The fifth FORTRAN compiler is called WATFIV. WATFIV is a high speed compiler with descriptive error messages making it good for debugging.

This memo describes major differences in FORTRAN on the two machines. Also included are brief descriptions of features that vary from version to version. Since this memo is a summary of differences, users are advised to consult the appropriate manual (all are in the AID Office) before using any features they are unfamiliar with.

Comparison of IBM/370 and DECSYSTEM-10

	IBM/370	DEC-10
Number of bits per word	32	36
Maximum integer values	2,147,483,647 (2**31-1)	34,359,738,367 (2**35-1)
Real number magnitudes	10**-78 to +75	10**-38 to +38
Real number significant digits	7.2 (decimal digits)	8 (decimal digits)
Double precision significant digits	16.8 (decimal digits)	16 (decimal digits)
Number of characters that can be stored in one word	4	5

Conversion Between IBM/370 and DEC-10

The major conversion problems occur when dealing with input and output. Each version of FORTRAN uses logical unit numbers to determine where data will come from or go to. Each logical unit number used must be associated with a particular data set or data file (some numbers have defaults). The way in which the association is made is different on the two machines. On the DEC-10, the programmer includes an OPEN statement in his FORTRAN program (OPEN or CALL DEFINE FILE if direct access I/O is desired). On the IBM/370, however, the association is made on a DD (data definition) JCL card. The logical unit numbers available for FORTRAN-G are 1

through 99, for DEC-10 FORTRAN, 1 through 63, and for WATFIV, 1 through 16.

Logical Unit Numbers and Default Devices:

IBM/370	WATFIV	DEC-10
1 card reader	1 card reader	1,3,5 teletype
2 card punch	2 card punch	2 card reader
3 line printer	3 line printer	4 console teletype
4-99 assignable	4-16 assignable	6 paper tape reader
		7 paper tape punch
		8 display
		9-15 DECtape
		20-24 disk
		19 assignable
		20-24 disk
		25-63 assignable

Another difference between the machines is the character representation. IBM uses a representation referred to as EBCDIC which uses 8 bits per character, allowing 4 characters per word (variable or array element). The DEC-10 on the other hand, uses ASCII representation, 7 bits per character, and allows 5 characters per word. The difference in the number of characters per word is an important conversion consideration.

Differences in Library Functions

Since FORTRAN-G has COMPLEX*16 and INTEGER*2 variable types (not available in DEC-10 FORTRAN), there are library functions for these types. There are also several double precision functions in FORTRAN-G that are not available in DEC-10 FORTRAN.

The arcsine and arccosine functions are named ARSIN and ARCOS in FORTRAN-G, but they are named ASIN and ACOS in DEC-10 FORTRAN.

FORTRAN-G has two groups of functions not available in DEC-10 FORTRAN: an error function and gamma and log-gamma functions (all with various types).

All of FORTRAN-G's trigonometric functions take arguments measured in radians; DEC-10 FORTRAN has, in addition, two functions (SIND and COSD) whose arguments are measured in degrees.

For the fourteen most common FORTRAN-10 (not FORTRAN-40) functions, the programmer is allowed to use a "generic" name whatever the numerical type of the argument. For instance, if the ABS (absolute value) function is used with integer or real or double precision variables, the FORTRAN-10 compiler will generate a call to the appropriate function.

The WATFIV library functions are the same as those of FORTRAN-G.

Special Features of FORTRAN-10 and FORTRAN-40

1. More than one statement may be written on the same line, separated by a semicolon, e.g. X=0;Y=3. (WATFIV also has this feature, but FORTRAN-40 does not.)

2. Two additional logical operators are provided, .XOR. and .EQV.. The EXCLUSIVE OR (.XOR.) will produce a true result if one but not both of the logical expressions it combines is true. EQUIVALENCE (.EQV.) is the logical analogy to .EQ..

3. The logical relations .EQ. and .NE. may be used with complex variables and constants.

4. Symbols may be used in place of the logical relations, i.e. .EQ., .NE., .LT., .LE., .GT., and .GE. become ==, #, <, <=, >, and >= respectively.

5. Literal, octal, and logical constants may be used in expressions, e.g.

```
IF(X.EQ.'STOP') GO TO 999
Z = X + "1776

I = 10*(J.EQ.1)
```

When a logical value is used in an arithmetic statement, .TRUE. has the value -1 and .FALSE. has the value zero.

6. Logical operators may be used on numerical variables or constants, in which case the operation is performed between corresponding bits.

IF L=6, then the expression L.OR.10 has the value 14

```
   L     0110 (base 2)
.OR.   1010  (base 2)
   =     1110 (base 2)=14 (base 10)
```

7. Arrays may be dimensioned to begin with a subscript other than 1. For instance, DIMENSION A(-5/10) sets up a sixteen element array whose first element is A(-5).

8. Arrays can have any number of subscripts (if space allows).

9. Subscripts used to address an array element may be real constants or expressions (FORTRAN-10 only).

10. Object-time dimensions. When dimensioning a dummy array in a subprogram, only a 1 is required for each subscript. When the subprogram is called, the dimensions will become the dimensions of the array argument supplied, e.g.

```
SUBROUTINE SUB(B)

DIMENSION B(1)
```

In this case, the argument to subroutine SUB can be a one-dimensional array of any length.

11. Variables in blank common as well as those in named common may be initialized in a BLOCK DATA subprogram (WATFIV also allows this).

12. A character array may be initialized in a DATA statement using one long quoted string, i.e. the string does not need to be separated into word-length strings.

13. Computed GO TO statements may use either an integer variable or an integer expression, e.g. GO TO (10,20,30), NTB-1.

14. When inputting a FORTRAN program from the terminal, a tab may be used to skip to the statement field. For a continuation line, the tab is followed by a single digit, then the remainder of the statement.

15. DO loop parameters may be positive or negative, constants or expressions. FORTRAN-10 also allows real constants or expressions, but these values are truncated before use.

```
DO 1 I=-10,J+2,3

DO 1 I=10,2,-2
```

16. FORTRAN-10 allows IBM type specifications: REAL*4, REAL*8, INTEGER*4, etc. However, since some of these types are not available, the type may not be as prescribed in IBM FORTRAN. In particular, INTEGER*2 and LOGICAL*1 both define full word variables in DEC-10 FORTRAN (a warning message is given). Also, COMPLEX*16 defines a single precision complex variable (with a warning). FORTRAN-40 only allows those types that are valid DEC-10 FORTRAN types.

17. TYPE<ACCEPT> Two special input/output statements for terminal I/O are provided. TYPE and ACCEPT statements automatically refer to the teletype.

18. DEC-10 FORTRAN allows free-field formatting for input and output. The user does not need to specify the size, but merely the type of field for all types except alphabetic (A format), e.g.

```
TYPE 10,X,Y<I

10 FORMAT (2F,I)
```

19. An O format is available for output of octal data (FORTRAN-G and WATFIV have a Z format for hexadecimal output instead).

20. A special character ($) may be used in format statements to suppress the carriage return at the end of the line. This is helpful in writing interactive programs, e.g.

```
    TYPE 39

39  FORMAT (' A=',$)

    ACCEPT 40,A

40  FORMAT(F)
```

This series of statements could cause the input of a value for A to appear as: A=39.2, rather than:

```
A=
39.2
```

21. When using sequential I/O, in addition to REWIND, BACKSPACE and END FILE statements, DEC-10 FORTRAN provides a SKIP RECORD and an UNLOAD statement.

22. REREAD. DEC-10 FORTRAN provides this special statement, which allows the program to read again the most recently read record according to a new format. FORTRAN-G (at SU) allows this feature, using a subroutine named REREAD.

23. ENCODE and DECODE statements cause conversion from binary to ASCII or from ASCII to binary. These features are similar to use of the CORE subroutine in FORTRAN-G and to the "core-to-core I/O" feature of WATFIV.

24. PAUSE 'string'. This statement causes the program to stop execution, print out the quoted string at the terminal and await user response (FORTRAN-G and WATFIV have a similar statement, but the computer operator must respond).

25. Direct Access I/O is provided, but rather than using a DEFINE FILE statement (as in FORTRAN-G and WATFIV), the programmer either calls the DEFINE FILE subroutine or specifies random access in an open statement.

26. Optimization of code is performed by FORTRAN-10 (but not by FORTRAN-40).

27. A powerful "Dynamic Debugging Technique" (known as DDT) is provided. DDT is designed to get rid of "bugs" in programs (see DDT manual).

Special Features of WATFIV

1. Multiple statements on a card are allowed, separated by a semicolon (FORTRAN-10 also allows this).

2. Multiple assignment statements. The same value may be assigned to more than one variable by using something of the form: X=Y=0.

3. Arrays can have a maximum of seven subscripts (FORTRAN-G also).

4. The features of a DATA statement and any type specification statement may be combined, e.g. INTEGER I,J/1,2/ (FORTRAN-G also allows this).

5. PRINT, PUNCH, READ. These special I/O statements are provided, for which appropriate logical unit numbers are assumed (FORTRAN-G also).

6. Z format. Hexadecimal output is accomplished with this format. FORTRAN-G also has Z format, while DEC-10 FORTRAN has O format for octal output.

7. Hexadecimal constants are allowed only in a DATA initialization statement.

8. Format-free output. WATFIV eliminates the need for format statements by providing default fields (except for A format), e.g. PRINT,X,Y,I

9. Expressions are allowed in an output list as long as the expression does not begin with a left parenthesis.

10. CHARACTER*n type specification. This special type specification defines a variable which can hold n characters and allows the variable to be initialized by a single long quoted string, e.g. CHARACTER*10 ABC/'ALPHABETIC'/

11. Core-to-core I/O. WATFIV has special forms of the READ and WRITE statements allowing formatted conversion of data without physical input or output. These statements are READ (var, fmt) list and WRITE (var, fmt) list, where var is a character variable or array and otherwise the statements are normal.

This feature is similar to ENCODE and DECODE statements in DEC-10 FORTRAN and the use of the CORE subroutine in FORTRAN-G.

12. Three special variable types are provided: LOGICAL*1, INTEGER*2 and COMPLEX*16. Also, the DOUBLE PRECISION specification (same as REAL*8) is allowed (FORTRAN-G also has these).

Special Features of FORTRAN-G and FORTRAN-H

1. The combination of the features of a DATA statement with a type specification statement is allowed, e.g. REAL X,Y/1.2,3.4/ (WATFIV also allows this).

2. Arrays can have a maximum of seven subscripts (WATFIV also).

3. PRINT, PUNCH and READ. These special I/O statements are provided, for which the appropriate logical unit numbers are assumed (WATFIV also has these statements).

4. Z format allows hexadecimal output. WATFIV also has this, while DEC-10 FORTRAN has O format for octal output instead.

5. Hexadecimal constants are allowed only in a DATA initialization statement.

6. Three special variable types are provided: LOGICAL*1, INTEGER*2 and COMPLEX*16. Also the DOUBLE PRECISION specification (same as REAL*8) is allowed (WATFIV also has these).

7. The dollar sign ($) is considered alphabetic, i.e. variable names may begin with it.

8. Function type may be specified inside the function, e.g.

```
FUNCTION INT(X)

REAL*8 INT
```

9. FORTRAN-G has a debugging feature which aids in analyzing problem programs.

10. A subroutine named REREAD allows a program to read the same record twice. (See Memo LF13 -- Aid Office.)

11. Another subroutine, CORE, allows formatted I/O operations to or from a variable (see Memo LF13 -- Aid Office). This is similar to ENCODE and DECODE statements of DEC-10 FORTRAN and to "core-to-core I/O" in WATFIV.

.COPY *.* FOR [454,12]

DECsystem-10 PACKET

INTRODUCTION TO TIMESHARING

TABLE OF CONTENTS

PRICE $1.25

SYRACUSE UNIVERSITY COMPUTING CENTER

INFORMATION SERIES

DOCUMENT	TITLE
GI23-1105	Introduction to Timesharing on the DECsystem-10
DATE	
09/10/77	
AUTHOR:	Dorothy A. Kerr

CLASSIFICATION: General Information

AUDIENCE: DECsystem-10 Users

LEVEL: Introductory

DISTRIBUTION: For sale in Room 122 Machinery Hall.

REFERENCES: Supersedes CCIS Memo P1-1074 entitled "Getting Started with Timesharing"

ABSTRACT:

This document provides the basic information needed to use the DECsystem-10. It describes the login and logout procedures, the basic use of the text editor SOS, and the most commonly used system commands.

19 Pages

$0.50

I. Introduction

The DECsystem-10 at Syracuse University may be used for both timesharing applications and for batch programming. To use the DEC-10 for timesharing applications you must be able to:

1) sign on, i.e. establish communication with the computer through a terminal,
2) run certain system programs to create, edit and manipulate files, and
3) sign off, i.e. terminate contact with the computer when you are through with your work.

This memo provides the basic information you will need to do each of these things. More advanced information for timesharing users is provided in CCIS Memos SI27, "DECsystem-10 Reference Sheets" and UA29, "DECsystem-10 Utilities"; batch users should see CCIS Memo SI19, "DECsystem-10 Beginners Guide to Multi-Programming Batch."

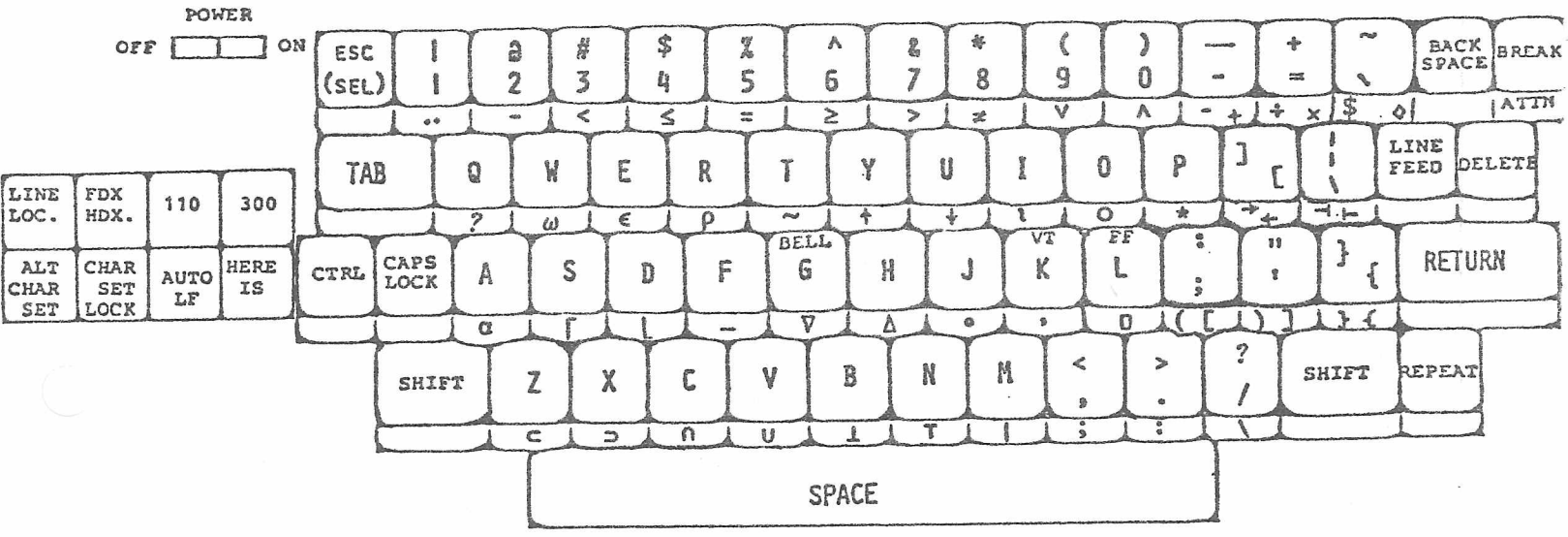
Before you can begin to use the DECsystem-10, you must have a valid account number. See the assistant business manager, Room 122 Machinery Hall, to apply for your incentive account, or get an account from your instructor for any course requiring the use of the DEC-10.

II. How to Sign on

Login and sign-on are two terms which are used interchangeably as names for the process of establishing communication with the computer from a terminal. This section of the memo describes in detail the steps you need to perform to complete the login process. It also tells you what you should do if you have problems during login.

The first thing you must do to sign on to the DEC-10 is locate a terminal. There are approximately 70 Decwriter terminals located at various places around campus. A list of terminal locations is available outside Room 122 Machinery Hall.

The keyboard of a Decwriter terminal looks like this:



The tops of the keys show the full ASCII (American Standard Code for Information Interchange) character set. This is the character set used to transmit information to and from the DEC-10. The yellow characters on the keyfronts are the APL special characters and should be ignored by DEC-10 users.

Once you have found a terminal and familiarized yourself with the keyboard, you should proceed to sign on as follows:

1. Check all 8 buttons at the left of the keyboard. Only the one marked 300 should be down. The other 7 should be up. (To change the state of a button, press it once.)

2. Press ON (upper left of keyboard). The red light marked STD CHAR SET will light indicating that the ASCII character set will be used.

3. Press the RETURN button. The computer will print a one line response followed by a period (prompt) at the left margin. The response should look something like this:

```
09E)SYRACUSE UNIV. TIMESHARING
.
```

4. After the period (prompt) prints, type the word login, a space, your account number and then press RETURN. For example, if you have account number 1234,56 this step would be as follows:

```
.login  1234,56
```

Note that your account number consists of two parts. The first part is your project number, and the second part is your programmer number. The two parts of the account number must be separated by a comma as shown above.

After you press RETURN, the word LOGIN and your account number will be printed again and then another message will print at your terminal. It will look something like this:

```
JOB 25 Syracuse Univ 6.03 TTY15
```

JOB 25 indicates the job number assigned to you by the system. Syracuse Univ 6.03 is the name of the current version of the monitor, (the system's control program) and TTY15 is an identification for the terminal you are using.

5. Next the computer types Password: to indicate that it is ready for you to type in the password which was assigned to you when you were given your account. When you type in your password, it will not print at the terminal. Printing is suppressed for purposes of security so that anyone picking up your discarded output or looking over your shoulder will not be able to learn your password. Remember to press RETURN after your password, and at the end of any line you type to the computer.

INTRODUCTION TO TIMESHARING ON THE DECSYSTEM-10

CONTENTS

If you are successful in all of the above steps, the time of day and any important messages from the operator will print, followed by a period (prompt) at the left margin. At this point you have completed the login procedure and are ready to begin working with the computer.

Below is a sample of the login process for account number 1234,56. The underlined information was typed by the computer.

```
09E) SYRACUSE UNIV. TIMESHARING
.login 1234,56
LOGIN 1234,56
JOB 25 Syracuse Univ 6.03 TTY15
Password:
1441    16-Jun-77        Thur

.
```

Problems at Sign-on

While the login procedure is fairly straightforward if all goes well, there are a number of things that can go wrong. If you press RETURN and get no response at all when you first begin to login, it may mean that the PDP-11, the computer which communicates between terminals and the DEC-10, is not functioning. If you get the message SYRACUSE UNIV. TIMESHARING, and then type LOGIN and get the message ?DECSYSTEM-10 NOT AVAILABLE, it means that the DEC-10 computer has some kind of problem and is not available for public use. If either of these problems occur, call the dispatcher, ext. 3-3245, for information about how long the computer will be unavailable.

When both computers are available, you may still have problems if you enter the wrong account number or password. If you get the message:

```
?LGNIET INVALID ENTRY-try again
#
```

try reentering your account number and password by typing your account number after the # and your password after the request Password:. Do not type LOGIN again. If trouble persists, check with the AID staff (Room 116 Machinery Hall) to see if you are entering your number improperly. If you are still unable to sign on, see the assistant business manager (Room 122 Machinery Hall) to verify that your account number and password have been entered into the accounting system correctly.

Changing Your Password:

If at any time you believe that someone else may have knowledge of your account number and password, you should change the password.

in fact, it is a good practice to change your password at regular intervals to insure the security of your account.

Your password may be changed only when you login. To indicate that you want to change your password, append /PASSWORD to your account number when you login. If you have account number 1234,56 you would enter the following:

.LOGIN 1234,56/PASSWORD

When the system responds

PASSWORD:

enter your old password. The computer will then print

New password:

You should then enter the characters you want for your new password. For example, to change your password to LIN, this line would be

New password: LIN

Note that the printing of the new password is not suppressed. Therefore, you must be very careful to blot it out or otherwise destroy the output so others cannot learn your new password.

This new password which you just entered must be used the next time you try to login.

III. The Monitor

When you have completed logging in to the DEC-10, you are in contact with the system program known as the monitor. The monitor is the controlling program on the DEC-10. Its function is to interpret commands, to select user programs to be run, to gather accounting information and in general to perform all those tasks which are necessary to keep the system running smoothly for all users.

One of the ways you can tell that you are communicating with the monitor is by the period prompt which it prints. The period which is printed at the end of login, for example, was printed by the monitor. Other programs have different prompts, such as *, line numbers, etc.

When you are in contact with the monitor, i.e. at monitor level, you may enter any of a number of monitor commands, you may begin running programs or you may sign off from the system. Each of these activities will be described later in this memo.

IV. How to Sign-off

Before going into any detail on how to use the DEC-10, let's discuss how to terminate the connection between your terminal, your account number and the DEC-10. This process of terminating communication between you and the computer is known either as logout or sign-off.

The simplest way to sign off is to type K/F in response to the monitor prompt. K means run the KJOB (Kill JOB) program, and F means try to do a fast logout and save all files which have been created. This fast logout will work unless you have created too many files and are over your allocated storage quota. Section IX of this memo describes what to do when this occurs.

After you type K/F and press return, information will print out telling you the date and time, how much file storage space you used, how long your job ran and what your charges are. The sign-off and associated information look like this: (Underlined information is typed by the computer).

```
.K/F
Job 25, User [1234,56] Logged off TTY15     1442  16-Jun-77
Saved all files (50 blocks)
Runtime  2.52 Sec

CHARGE: $0.14=$0.06(C)+$0.06(P)+$0.02(K)
Y-T-D ACCOUNT USE: 81%
```

The amount you are charged for using the computer is based on three things. The first is connect time (C) which is the elapsed time from the beginning of a terminal session to the end. The second is CPU or central processing unit time (P). This is a measure of the actual amount of computing done by your job. Finally, you are charged for the amount of core your job used when running (K). The total charge for your job and the charge broken down into each of these components are reported on the line labelled CHARGE. The last line tells you the per cent of your total account used so far this year.

V. Naming Files

The DEC-10 is set up so that all programs and data are stored as files. A DEC-10 file is simply a collection of information that is given a name and stored as a unit. Because all programs and data are stored as files, it is essential that you learn to name files, create files, edit files, and perform other functions with files.

File names on the DEC-10 have two parts. The first part of the name is the actual name of the file; the second part of the name is called the file extension and indicates what kind of file it is. The name part of a file name may be from 1-6 letters or digits, while a file extension has from 0-3 letters or digits. The file name and extension, if any, must be separated by a period.

The choice of a file name is entirely up to you as a programmer. Generally it is a good idea to choose a name which indicates the purpose or contents of the file and an extension which indicates the kind of file it is. Some standard file extensions and their meanings are:

.FOR	Source file in FORTRAN language
.BAS	Source file in BASIC language
.CBL	Source file in COBOL language
.ALG	Source file in ALGOL language
.DAT	Data file
.REL	Relocatable binary file
.TMP	Temporary file

If you write a FORTRAN program to calculate an average, for example, you might want to call it AVG.FOR or AVRAGE.FOR, or some similar name indicating its purpose (to calculate an average) and its type (FORTRAN). Similarly, a COBOL program for processing payroll data might be called PAYROL.CBL while the data it uses might be in a file called PAYROL.DAT.

A file on the DEC-10 also has as part of its full name the project-programmer number (account number) of the person who owns it. This project-programmer number part of the file name doesn't need to be specified when you reference your own files. However, it is a part of each file name so that there is no possibility of confusing your file with a file of the same name and extension but belonging to another user. That is, if your project programmer number is 1234,56 and you have a file named MYPROG.FOR, the full specification for your file is MYPROG.FOR[1234,56]. Since your file is the only one with your account number as part of its name, there is no chance for confusion with any other file.

VI. Creating and Editing Files

The easiest and most commonly used way to create files on the DEC-10 is to use a text editor. This section of the memo describes how to use SOS, a line oriented text editor which may be used to create and edit files. More details on the use of SOS may be found in CCIS Memo TX-2, The SOS Manual.

Since SOS is simply a program which manipulates text, the first thing you have to do to use SOS is give the monitor command which runs the SOS program. This is done by entering the word SOS after the monitor prompt.

You must also give the name of the file you want to create or edit. If you give the name of a file which has not been created previously, SOS will assume that you want to create a new file with that name. If you give the name of a file which already exists on disk, SOS assumes that you want to edit that file.

Input Mode

First let's consider using SOS in input mode to create a new file. Suppose you want to create a file called TESTA.FOR. To create this file, type

```
.SOS TESTA.FOR
```

and press RETURN. SOS will respond with a line number indicating that it is ready for you to begin. Type in the first line of your file and press RETURN. SOS will print the next line number and you can then enter the next line of your file. Continue entering lines one by one after the line number prompt until your entire file has been entered.

When you have entered the last line of your file, press the escape key (labelled ESC). A $ will print, and then the * prompt on the next line. This indicates that SOS has left input mode, and is now in edit mode. That is, the entries you make now will not be considered lines of your file, but will be interpreted by SOS as instructions on how to edit your file. The edit mode prompt is the * while the input mode prompt was a line number.

The following example shows SOS being used in input mode to create a file. After the file information was entered, ESC (which prints as a $) was pressed to end input mode and enter edit mode. The underlined information was typed by the computer.

```
.SOS MYFILE.DAT
Input: MYFILE.DAT
00100  THIS IS A TEXT FILE
00200  WHICH ILLUSTRATES
00300  THE USE OF SOS
00400  TO INPUT A FILE.
00500  $
*
```

Note that the $ prints when the escape character is pressed. In this memo, $ represents the escape character, not a dollar sign.

Edit Mode

Edit mode is entered by pressing ESC in input mode, or by giving the SOS command and an old file name. The edit mode prompt is the *. While in edit mode you may do such things as print out lines, delete lines, replace lines, or leave SOS entirely. The following is a list of some edit mode commands along with an explanation of how they are used.

P Print at the terminal the line or lines indicated. To print line 100, use
P100
To print lines 200 to 400, use
P200:400

D Delete from the file the indicated line or lines. To delete line 100, use
D100
To delete lines 300 to 600, use
D300:600

I Insert a line. For example, to insert line 350 you would type I350 and press RETURN. The number 00350 will print at the margin and then you may enter the line you wish to insert. If you already have a line numbered 350, SOS will interpret the command I350 to mean that you want to insert a line after line 350 but before the next line in your file, and will choose an appropriate line number for you. Normally the insert command will allow you to insert only one line at a time. However if there is room to insert more than one line (incrementing by 100), SOS will remain in input mode until escape is entered or until the next line of the file is reached. SOS also enters input mode after the insert command if the line you are inserting is the last line in the file. This allows you to add lines to the end of a file.

R Replace the indicated line or lines. To replace line 100, enter
R100

SOS will respond 00100 and wait for you to enter the replacement line. To replace lines 200 to 400, enter
R200:400

Note that the replace command functions like the insert command in that SOS will enter input mode if there is room for multiple insertions.

S Substitute one character string for another. To use this command, type an S followed by the old character string, the escape character, the new character string, the escape character and then the line number in which the substitution is to take place. For example, to change the word FOR in line 600 to the word TO, the following command would be used ($ represents the escape character)

SFORTO600

Be sure to use enough characters to uniquely identify the item you want changed, because the substitute command changes all occurrences of the old string to the new string.

E Exit from SOS and save the file (with line numbers). This is the normal way to exit from SOS.

ES Strip line numbers and then exit and save the file. This command should be used for creating the final copy of data files and text files. It is also an easy way to renumber a file since if you reenter SOS to edit a file which has had the line numbers stripped, SOS will automatically generate a new

set of line numbers for the file. These new line numbers will start at 100, with increments of 100.

A Enter alter mode to edit character by character within a line of the file. To enter alter mode for line 300, type A300. Alter mode editing is described in the next section of this memo.

Alter mode

Sometimes when you have a few small changes to make to a line, it is easier to go directly into the line and change a few characters rather than replace the entire line or substitute for parts of the line. To do this, you use SOS in alter mode.

Alter mode is entered from edit mode by giving the alter command followed by a line number. Once in alter mode you can move an imaginary pointer back and forth within the specified line to indicate where you want to change, delete or insert characters. There are four commands for moving the pointer, and several other commands to do the actual changes to the line. One major difference between alter mode and the other modes is that alter mode does not give prompts or print the users command.

The alter mode commands to move the pointer are:

space Pressing the space bar will move the pointer forward one character.

backspace Pressing the backspace key will move the pointer back one character.

L Typing the letter L will print out the entire line and return the pointer to the beginning of the line.

S Typing the letter S followed by any character will cause the pointer to skip to the first occurrence of that character. If you want to skip to other than the first occurrence of a character, precede the S command with the number of the occurrence to be skipped to. For example, to skip to the third q in a line you would enter 3SQ.

The next group of commands is used to make changes to a line. These commands act on or before the character indicated by the pointer.

C Change the next characters. For example, CA will change the next character to the letter A. 3CXYZ will change the next 3 characters to XYZ.

D — Delete the next characters. For example, D will delete the next character, while 3D will delete the next 3 characters.

I — Insert, before the next character, whatever is typed next. Stop inserting when the escape key or return is pressed.

RETURN — Leave alter mode and return to edit mode.

Exercises

The only way to really gain familiarity with a text editor is to use it, so a sample exercise for you to try at your terminal has been included. The left column of the following example is a series of commands to the various modes of SOS. (The SOS command is, of course, given to the monitor.) The right column describes the results of typing the corresponding lines of the left column. Try this exercise at a terminal. If you have any problems or don't understand any of the results, reread the appropriate sections of this memo, or see CCIS Memo TX2, The SOS Manual, for more details on SOS. Remember that in this example, only what you enter is listed. The prompts from SOS and the monitor are not shown.

Type this:	To do this:
	INPUT MODE
SOS FILEA.TST	
AAAAA	
BBBBB	
CCCCC	
DDDDD	
EEEEE	
$	(Press the escape key, not the dollar sign.)
	EDIT MODE
P300	print line 300.
P300:500	print lines 300 through 500.
I100	insert a line after line 100.
ABABAB	
D400	delete line 400.
E	exit from SOS.
SOS FILEA.TST	reenter SOS (edit mode).
P100:500	print the whole file.
R200	replace line 200.
ZZZZZ	
I500	insert lines 600,700,...
FFFFF	Note that if there is room for the insertion of
GGGGG	another line (incrementing by 100), then SOS
HHHHH	will insert another line.
$	(Press the escape key, not dollar sign.)
D200:500	delete 3 lines.
R100	replace line 100 with some numbers.
123451234512345	
A100	enter alter mode for line 100.

ALTER MODE - Note that commands do not print.

<space><space>	press the space bar twice to move to the 3rd character of the line.
<bksp>	press the backspace key to move back to the 2nd character.
L	enter the letter L to print out the line and return the pointer to the start of the line.
2S3	skip to the 2nd 3 after the pointer
C0	change it to a zero.
D	delete the next character.
S1	skip to the next 1.
i000$	insert 3 zeroes before the 1. (end the insertion by pressing the escape key.)
<cr>	press RETURN to end alter mode and return to edit mode.
P100:800	print the entire file.
E	exit from SOS.

VII. Commands

When you type SOS followed by a file name, you are indicating to the monitor, or controlling program, that you want to run the program SOS. In other words, you are giving the monitor the SOS command. Since the command was given to the monitor, SOS is said to be run at monitor level. There are many other programs besides SOS which you may want to run at monitor level. These programs may be used to do such things as list the names of your files, type at the terminal the contents of a file, list a file on the line printer or delete a file which is no longer needed. Some of the more commonly used of these programs are set up like SOS so that they may be run using a command. Others do not have a specific command associated with them, but instead are run by using the R (run) command followed by the program name. This section of the memo describes some of the basic commands which may be given to the monitor. For more detail on these and other commands see CCIS Memo SI27, "DECsystem-10 Reference Sheets" or the DEC-10 Operating System Command Manual.

The DIRECTORY Command

The directory command requests from the monitor a listing of the "directory" of your file storage area. That is, it asks for a list of the names of the files you have created and for some of the information associated with those files. To get a directory listing, type DIRECT in response to the monitor prompt as follows.

```
.DIRECT
```

(If you prefer, you may use the three letter abbreviation DIR instead.) The monitor response to the DIRECT command is to type at your terminal the names of your files, the written size of each in blocks, the protection codes associated with each file and the date

each file was created. These are all explained below. For example, if you have created two files, F1.BAS and FORM.FOR, and have edited FORM.FOR using SOS, the output from the directory command might look like this:

```
.DIRECT

F1          BAS         6         <057>         27-APR-77  DSKA:[1234,56]
FORM        FOR         7         <057>         21-MAY-77
FORM        QOR         7         <057>         22-MAY-77

Total of 20 blocks in 3 files on DSKA:[1234,56]
```

The first column contains the file names of all files in your storage area. The second column contains the extensions for these files. Although you only created two files, three are listed. The third file, FORM.QOR, is a backup file created by SOS. It contains a copy of your file FORM.FOR as it was before you edited it. Many other programs also create auxiliary or backup files which will show up when you give the directory command. These files usually have the same name as another file but a different extension. Information on how to delete these extra files is presented later in this section.

The third column of the output tells the number of blocks required to store the information in each file. A block is simply a unit of measure which is equivalent to the space required to store 640 characters. When files are written to disk, the amount of space they actually use is always greater than the size reported here. First, every file has an additional two blocks of system information associated with it. Second, space for files is always allocated in units of 5 blocks at a time. Therefore, a file like F1.BAS, whose size is reported as 6 blocks, will actually use 10 blocks -- 6 for data, 2 for system information and 2 empty. It is the space actually allocated to a file, or its allocated size, that is used in determining whether or not you have exceeded the file storage quota established when your account was set up. See Section IX of this memo for more information on storage quotas.

The number in angle brackets in the fourth column of output gives the protection associated with each file. The protection code is simply a numeric indication of who is allowed to access your files. For details on protection, see CCIS Memo SI27, "DECsystem-10 Reference Sheets (File Specifications)."

The fifth column of the directory output lists the dates on which the files were created. DSKA:[1234,56], which appears at the top right of the output gives the system identification for the device on which your files are stored and gives your project programmer number. The summary line at the bottom of the output tells you the total amount of storage you are using. Remember that this is in written blocks, not allocated blocks, so the actual amount of storage charged to your account number will be greater than this.

The TYPE Command and the PRINT Command

The directory command gives you a list of the names of the files in your storage area. If you would like to list at the terminal the contents of one or more of those files, use the TYPE command. For example, the command

```
.TYPE FORM.FOR
```

will cause the contents of file FORM.FOR to be typed at the terminal.

If the file you want to list is a long one and you don't want to wait while it prints at the terminal, you may have it printed on the high speed line printer instead. To list the contents of FORM.FOR on the high speed printer, enter

```
.PRINT FORM.FOR
```

If you want more than one copy of a file printed, you may use a special "switch" for multiple copies.

```
.PRINT FORM.FOR/COPIES:3
```

will cause 3 copies of file FORM.FOR to be printed. Line printer output is dispatched every half hour or so, and may be picked up in the bins in the user area on the first floor of Machinery Hall.

The DELETE Command

Once you are through using a file it is a good idea to delete it. You may also want to delete the backup files created by SOS, or any other backup or auxiliary files which may have been created. To delete a file type DELETE (or DEL) followed by the name of the file to be deleted. To delete the file FORM.QOR for example, type

```
.DELETE FORM.QOR
```

If you want to delete more than one file, list the names of the files to be deleted, separated by commas, after the DELETE command.

```
.DELETE FORM.FOR,FORM.QOR
```

for example, will delete the two files FORM.FOR and FORM.QOR. Be sure not to delete any files which you may wish to use in the future.

The EXECUTE Command

The programs which you write in FORTRAN, COBOL or any other higher level language are known as source programs. They are written in a language which is understandable to you, but which must be translated for the computer into another form called machine

language. Machine language consists of a series of 0's and 1's and is the only language which the computer can understand and execute. The process of translating from a higher level language to machine language is known as compilation.

The compiled version of a program is called a relocatable binary program. It is written out to disk as a file with the same name as the source file and with the extension REL. Before the instructions in the program can be executed, the relocatable binary version must be loaded into the computer's core memory. The process of bringing a REL file from disk into core is called loading the program. Only after loading is complete can execution begin.

The compile, load and execute sequence described above can be initiated in many ways. The easiest way is with the EXECUTE command. If you type

```
.EXECUTE MYPROG.FOR
```

for example, first the FORTRAN compiler will be called to compile the source version of MYPROG.FOR and to create a relocatable binary file named MYPROG.REL. After compilation is completed, the file MYPROG.REL will be loaded into core and execution will begin. Note that in this case the FORTRAN compiler is called because of the extension FOR; the extension CBL would cause the COBOL compiler to be used, etc.

If all is successful, you will get the results you expected from your program. If an error occurs, either at compile time or during execution, you will have to analyse your program to see what is wrong, and use SOS to edit it to correct any errors you found. Then after correcting any errors, issue the EXECUTE command again to compile, load and execute the corrected version of your program. This process of run, analyse, edit, run, analyse, edit, run... is called debugging.

Once your program is correct, you may want to run it several times on different sets of data. If so, there is no need to recompile your program each time it is to be run. All you need to do is load the program into core and begin execution. You may still use the EXECUTE command to do this, because the EXECUTE command always checks to see if a REL file of the same name as the file being processed already exists. If it does not find an appropriate REL file, it compiles the source file it is given. If it does find a REL file, and the REL file is newer than the source file, it assumes that the REL file was produced from the given source file by a previous compilation step. Therefore it does not try to recompile but simply uses the REL file which it found and continues with the load and execute steps.

There are other commands similar to EXECUTE which may be used when you only want to compile a program, or only load a program. These commands are discussed in detail in CCIS Memo SI28, "Compilers and Compile Class Commands."

The SET TIME Command

When you are first learning to program, one of the easiest mistakes to make is to write a non-terminating or infinite loop. This is a serious problem since such a loop can quickly use up all of your account. To prevent a loop from executing indefinitely, you may set a time limit for execution with the SET TIME command. If you type

```
.SET TIME 5
```

immediately before the EXECUTE command to run a program, then program execution will be stopped as soon as 5 seconds of CPU time have been used. Since most small programs take a second or less to run, if your program runs for 5 seconds it may be an indication that the program contains an improperly specified loop. Of course for large programs, larger time limits should be used. Also, be sure to reset the time limit each time before you execute your program.

The R Command

Many system programs do not have a specific command associated with them. To run these programs, the R command is used followed by the name of the system program to be run.

```
.R QUOLST
```

for example will run the system program QUOLST. Many of the programs which may be run using the R command are documented in CCIS Memo UA29, "DECsystem-10 Utilities."

The INDEX Command

The INDEX command allows you to query the system to discover what programs and packages are available on the 370 and on the DEC-10. To use the INDEX command, enter

```
.INDEX
```

You will then be asked whether or not you want instructions. Respond either yes or no. If you request instructions, they will print followed by an * prompt. If no instructions are requested only the * will print. At that point you may begin your request for information. You should enter either the name of the particular computer program you are interested in, or the name of a subject you would like information about. The entry STOP in response to the * will return you to the monitor. See CCIS Memo GI26, "Index: A Guide to SU Computer Systems," for more information on the use of the INDEX program.

VIII. Miscellaneous Commands and Control Characters

Control C

The control C character is typed by holding down the key marked CTRL while at the same time typing the letter C. Control C (^C) is used to interrupt the program that is currently running and return to the monitor. If the program which is running is waiting for input, it can be stopped with one control C. Otherwise two control C's are needed to stop program execution. Once you have typed the one or two ^C's and are back at monitor level you may use any system commands you wish. You may want to use control C if you see that your program is producing meaningless results or if you suspect that your program may be in an infinite loop.

Control U

Control U (^U), which is formed by simultaneously pressing the control key and the letter U, is used when you have mistyped a line and would like to start that line all over again. It causes the deletion of the entire line currently being entered, issues a carrier return and line-feed and allows you to start typing the line over again. This is particularly useful when you discover a lot of errors in a line before you press return, or when you start to issue one command and decide you would rather issue a different command.

Backspace

The backspace key is used to correct errors in the line currently being entered. If you make a typing error while entering a line, simply backspace to the error and retype the line from that point. All characters which are backspaced over are erased and will be replaced by the new characters which you type. If you backspace over more than a few characters you may wish to make the correction easier to read by manually rotating the paper feed to move the paper up. The new characters will then by typed beneath the old ones rather than on top of them.

SET TTY LC

Sometimes, when you are preparing text files for example, you need to have the computer distinguish between upper and lower case alphabetic characters. If you want alphabetic characters to be transmitted exactly as entered, you must issue the command

```
.SET TTY LC
```

This should be done before you enter any text and not after, since it effects how the text is stored when it is entered. For example, if you want to use SOS to create a text file with both upper and lower case characters, you should first enter:

```
.SET TTY LC        and then
.sos filename
```

IX. How to Log Out if You Have Too Many Files

Each user of the DEC-10 is assigned two storage quotas. The first, known as your logged in quota, places a limit on the amount of file storage space which can be used while you are logged in. The second, known as your logged out quota, defines the amount of disk storage which you may keep when you are not logged in. Because many programs generate temporary files, your logged in quota is always larger than your logged out quota.

After you have been using the DEC-10 for a while, you may find that you have created more files than you are allowed to keep when you are logged out. Usually you discover that you are over your logged out quota when you try to log out with K/F and get a message like:

```
?DSKA  Logged out quota 100 exceeded by 15 blocks

Confirm:
```

This means that your files are taking up 15 blocks more than you are allowed to use when you are logged out. Since you must be below your logged out quota to sign off from the system, you must delete enough files to get at or below your quota before you can complete the logout process. The easiest way to do this is as follows:

1. Type ^C (control C) to end the logout program.

2. If necessary, give the DIRECT command to discover the names and sizes of your files.

3. Use the DELETE command to delete enough files to get below your logged out quota.

4. Type K/F again to try to logout. If you have deleted enough files, the logout will be successful. If not, go back to Step 1.

Often you will be over your logged out quota because you have generated a lot of temporary and/or backup files which you really don't need. For example, if you have been using SOS you will probably have generated some files with extensions starting with Q. Similarly, if you have been executing several programs, you may have generated some .REL files. These are generally the files you should attempt to delete first, since REL files can always be re-created from the source files and since the SOS backup files are usually not needed.

If you know that you want to get rid of all files with a certain extension, or all files with a certain name whatever their extension, you can use a shorthand technique for indicating which files to delete. This is done by using the * as a "wild card" as follows:

```
.DELETE *.REL
```

This will delete all files with the extension REL. Similarly,

```
.DELETE TEST.*
```

will delete all files with the name TEST, whatever their extension. The * wild card instruction is very powerful and of course should be used with care.

X. Where To Go From Here

The information covered in this memo should provide you with the basic tools you need to begin work with the DEC-10. In addition, it should provide the basic vocabulary you need to read and understand other DEC-10 documentation. The next memo you should read after you have mastered the information here is CCIS Memo SI27, "DECsystem-10 Reference Sheets." Also, you should check the INDEX program and CCIS Memo GI4, "Current Publications", for documentation which may help you with any special problems you are having. If you need more help than is available in the documentation, check with the AID Staff, Room 116 Machinery Hall.

Index

SYRACUSE UNIVERSITY COMPUTING CENTER

INFORMATION SERIES

DOCUMENT	TITLE
GI24-1213	DECsystem-10 Availability and Capabilities
DATE	
01/05/79	
AUTHOR	John Thornton

AUDIENCE: DECsystem-10 Users

COMPUTER(S): DECsystem-10

LEVEL: Introductory and Intermediate

REFERENCES: Supersedes GI24-1106, titled as above.
See Appendix A

ABSTRACT:

This memo contains general information about the DECsystem-10 computer at Syracuse University.

9 Pages

$0.00

1. Introduction

The Syracuse University Computing Center operates a Digital Equipment Corporation DECsystem-10 Computer System to complement and increase the computational resources of Syracuse University.

The DECsystem-10 Computer System is currently performing both general-purpose conversational time-sharing, and multi-programming batch processing. The time-sharing user can develop a program at a terminal, debug the program interactively, and execute the program with conventional or interactive input and output. In most cases, this mode of operation significantly reduces the time required to write a program and obtain useful results. The batch-processing user can submit a job to the system via cards or from a terminal, to be run as soon as possible or at a later time.

2. System Availability

a. Hours of Operation

See Posted Hours in Aid Office, Room 116, Machinery Hall, x3424.

b. Location and Terminal Accessibility

The DECsystem-10 is located in the first floor machine room of Machinery Hall. Both the "User Terminal Room", housing several terminals, and the "Dispatch Room", for receiving printed output, are also located on the first floor of Machinery Hall. Cards and tapes should be submitted to the Dispatcher in the Dispatch Room.

At the present time almost all terminals at Syracuse University (DECwriters, ADDS terminals but not IBM 3270 terminals) can communicate with the DECsystem-10.

c. Account Validation

In order to use the DECsystem-10, a validated project number, programmer number, and password must be stored on the systems disk. The procedure to be followed is:

1) Get an SUCC account number from the Assistant Business Manager, Room 122. She will give the user a copy of this number on a signed form.

2) The information on this form will be stored in the system within 24 hours, after which time the user can access the system.

3. System Capabilities

a. Hardware

The DECsystem-10 Model 1090 computer has the following hardware:

1 KL10B	Central Processing Unit
384K	Core Memory (K=1024 words)
7	Disk drives
4	Magnetic Tape Drives (Shared with IBM/370, 6250/1600 bytes/inch)
2	Magnetic Tape Drives (800bytes/inch)
3	DECtape Drives
1	High Speed Line Printer
1	Card Reader

Notes:
1) The KL10B processor is "microprogrammed". That is, many of the system's machine language instructions actually consist of several micro instructions, allowing more flexibility than a non-microprogrammed processor has. The processor has a 2048 word "cache" memory, more aptly described as fast memory. This memory can be used to read four instructions, at one time, from core memory so that for three out of four instructions, the processor does not have to wait for the next instruction to be retrieved from core.

2) Although we currently have 384K words of core memory, this is to be expanded to 512K sometime within the next year. At that time, we will have "Four-way interleaving", that is, the core will be divided into four 128K segments. The processor can then read one word of information from each segment (for a total of four at one time) into Cache memory, speeding up the processing.

3) One disk drive is reserved for system storage space - for languages, utility programs, libraries and information and documentation files. Two disk drives are available for system programmer's and user's private, mountable disk packs. The four remaining disk drives are grouped together as one unit named DSKA. This is where users have their storage space. Initially, users are given a 50 block storage quota although they can use up to 1000 blocks while signed on to the system.

A block contains 640 characters (letters, digits and special characters).

4) Magnetic tape provides a larger storage capacity, approaching 200,000 blocks, depending on the tape's recording density. This can be used to archive (store) old programs and data, to save infrequently used information between uses and to provide a backup for current disk storage. Magnetic tape also provides the only way of passing information from the DECsystem-10 to the IBM/370 (the opposite route can be accomplished by both magnetic tape and punched cards).

The primary densities available are 6250 bytes/inch and 1600 bpi. At the moment, there are also 2 800 bpi tape drives available but these are expected to be removed within the next year. Any new tape should be written at one of the 2 higher densities.

5) DECtape is a special format of magnetic tape. It is designed to provide the storage ability (archiving, backup, etc.) of regular magnetic tape, while allowing simple disk-like access to the information. The capacity, however, is much smaller than regular magnetic tape. Each DECtape can hold a maximum of 574 blocks of information.

6) There are nearly 75 terminals in various locations across campus. Each of these terminals is linked to a PDP-11 computer which controls the communication and links the terminals with either the DECsystem-10 or the IBM/370.

b. Software

The DECsystem-10 has one assembly language processor (MACRO-10) and eleven primary higher-level language processors (FORTRAN-10, BASIC, ALGOL, LISP, SAIL, FLECS, SNOBOL, PASCAL, FASBOL, BLISS, and COBOL). All of these processors except LISP are reentrant, thus conserving memory space and time.

A number of utility programs exist which considerably ease the construction, debugging, modification, and execution of programs. The following is a list of some commonly used system programs (CUSPS) and libraries.

BACKUP - This program can perform disk to tape and tape to disk file transfers. It simplifies the use of magnetic tape. BACKUP has four kinds of commands: tape positioning, status setting, action and runtime commands.

BASIC PROGRAM LIBRARY - This library includes a variety of useful and interesting programs and games for the user of the BASIC programming language.

CHANGE - CHANGE converts data files from one character set to another. This conversion is often required when moving data or programs from one computer to another, e. g. between S.U.'s DEC-10 and IBM/370. It is most often used to write an IBM-style data-set onto a magnetic tape from a DEC-10 type file, or the reverse.

CREF - This program produces a cross reference listing of a program. This listing consists of a list of variable names, labels, etc and indications of each location in the program where the names were used.

CSORT - The CSORT program sorts the records (lines) of a file so that a particular segment (called the key) of the lines is in alphanumeric order. This order is defined by the ASCII collating sequence. The sequence can be used in the standard order (ascending) or the reverse (descending).

DDT - The "Dynamic Debugging Technique" is a program for getting the bugs out of compiled programs. DDT allows "interactive execution", that is, the running of a program step by step, pausing to input or output values in order to isolate any problems the program may have. Also available are simplified versions of DDT designed for FORTRAN programs (FORDDT), for COBOL programs (COBDDT) and for ALGOL programs (ALGDDT).

DECUS LIBRARY - The DECUS Library is a set of ten tapes containing many general (and specific) interest programs. Included are tutorials, games, application programs, utilities, languages,... These programs are in many different languages. Except for a few commonly used programs, users are responsible for copying the desired programs into their own storage areas for use.

DIRECTORY - The DIRECTORY program allows the user to get a listing of the names of the files stored in the user's own storage area, or in other areas (if allowed access to those files). The listing includes other information in addition to filenames, e.g. creation dates, size of the files, etc.

DUMP - The DUMP program is used to get readable output of the (binary) contents of specified memory locations. The stored information can be converted into assembly language equivalent, octal digits or ASCII characters (among other forms), generally giving an idea of what is stored.

DYNAMO - "Dynamic Modelling"; a compiler developed at MIT for describing and running continuous Models. DYNAMO is a programming language which allows a shorthand notation to represent differential equations. These coded equations can then be solved by the computer.

FILCOM - The FILCOM program is designed to compare two versions of a file and to list any differences encountered (or just a note about whether or not there are differences, if you prefer). FILCOM can compare ASCII or binary files. The output file from FILCOM will contain a listing of lines that differ in the two files.

FILEX - FILEX is a file transfer program which is capable of doing certain conversions of format. It's most significant uses are copying an entire DECtape to another DECtape and converting .SAV type core image files to .EXE type files (the new format).

GLOB - This program reads one or more binary program files (compiled programs) and produces an alphabetical cross reference listing of all the global symbols used.

GRIPE - A "Suggestion and Complaint" box. GRIPE allows any user to type comments into a special storage area which staff members periodically check.

HELP - HELP is a program which types out information about many DEC-10 programs and commands. The command HELP gives a description of the HELP program. "HELP *" returns a list of the items for which there is information, i.e. things which have a help file. HELP followed by a name gives information about the named item, if available.

INDEX - The INDEX program is an on line documentation system. You can get a listing of the available Software at S.U.(on both computers) for a given topic, or discover if a particular program is available. INDEX also points towards further documentation and gives notes on where to get help and how much help to expect.

ISAM - This program is designed for the maintenance of indexed sequential files. With this program, you can build an ISAM file from a sequential file, do the reverse, or update an ISAM file's storage parameters.

LIBARY - This program is a text editor for creating and correcting COBOL library files. A COBOL library file is stored in a special format (thus the need for a special editor) and is used by the COPY verb in COBOL to simplify writing programs.

LINK-10 - This program is the default loading program. Consult the LINK-10 Manual in the AID Office.

MAKLIB - The MAKLIB program is used to update binary library files. It is possible to create files which contain binary "modules", i.e. program units stored so that they are accessible to programs you may write. These modules can be combined into library files for use by several programs.

PIP - The "Peripheral Interchange Program" transfers files between devices: disk to terminal, DECtape to line printer, etc. It is also capable of doing some processing of the information, renaming files and deleting files.

QUOLST - QUOLST is a simple program which returns the user's storage allocation and quota information.

RUNOFF - A text formatter, RUNOFF does all of the things involved in writing papers (justification, page numbering, footnotes, etc.). To use RUNOFF, create a text file containing the information to be used. Then insert the appropriate RUNOFF commands into the text. The RUNOFF program will justify the text, split into pages and take care of other details desired.

SETSRC - SETSRC is a program which allows you to find out and/or alter your "search list". The computer uses the search list whenever you specify a file on disk. Since there are actually several disks in the System, we need the search list to specify the order in which the disks should be searched for the file.

SOS - An extremely powerful and concise text-editing program with approximately 50 commands for inserting, deleting, appending, searching and displaying text. Editing is performed either line by line or by character within a line. In addition to conventional editing operations, it also provides interactive search, match and substitution operations.

STATPACK - STATPACK is an interactive system for performing a variety of statistics. The STATPACK commands deal with one form or another of statistical analysis. It is fairly simple to use, although its manual is fairly large.

SYSTAT - The System Status program returns a wide variety of information about the who, what, where, how much, etc. of system use. The command returns which users are currently logged in, what programs are being used, what devices are being used, and much more.

TECO - An extremely powerful and concise text-editing language with more than thirty commands for inserting, deleting, appending, searching and displaying text. Editing is performed on a character line, or a variable character string basis. In addition to conventional editing operations, it also provides interactive string search, match and substitution operations.

A complete list of the software available on both the DECsystem-10 and the IBM/370 is given in the INDEX system which is an on-line documentation program.

4. System Advantages

The DECsystem-10 hardware and software capabilities provide a computing system that has particularly significant strengths and advantages in the following areas:

Interactive usage - The monitor system has been designed to provide rapid response to a large number of simultaneous users, each of whom can be interactively creating, editing, debugging, or executing a program. This interactive mode of use results in a significant reduction in the time required to obtain useful results from the program. Because the usual waiting for batch processing output is eliminated, the user can concentrate his time on one project, rather than "multi-program" his time among a number of projects. Consequently human as well as computer output is increased.

These advantages of interactive usage are independent of the implementation language (e.g. FORTRAN, LISP, COBOL, or MACRO-10) or the ultimate mode of use (batch or time-sharing) and consequently benefit all users.

Instruction repertoire - The 398-instruction repertoire of the DECsystem-10 simplifies assembly coding and can reduce the size of higher-level language programs.

Word length - The word size of the DECsystem-10 is 36 bits. This permits single precision arithmetic to be accurate to 7 significant digits, while double precision is accurate to 18 significant digits. The DECsystem-10 represents alphanumeric characters in the 7-bit ASCII code (in contrast to the IBM/370 which uses the 8-bit EBCDIC code). Thus one word in the DECsystem-10 can store 5 characters.

5. Memos and Short Courses

a. Memos - There are several CCIS memos documenting various programs and systems of the DECsystem-10. The memos available are listed in CCIS memo GI4 (in the AID office); most are available free of charge in the AID office. The first thing needed by most users is the "Introduction to the DECsystem-10" packet available in Room 122 Machinery Hall for $1.25. See Appendix C for a list of CCIS Memos pertaining to the DECsystem-10.

b. Short Courses - Several short courses, in the form of an introduction to the facilities and software of the DECsystem-10 computer system, have been given in the past and will be given in the future with sufficient interest.

Other short courses, tailored to particular subjects, will be scheduled if sufficient requests are made.

6. Pricing Information

a. See CCIS Memo OP1-1131 titled "SUCC Charges".

7. Accounting

All charges for DECsystem-10 usage will be billed against the SUCC account number which covers all SUCC resources (IBM 370/155, APL, etc.). Those users who are using both computers should remember that the total of the charges on both computers will be billed against their account number, and compared to their budgetary limit.

8. Software and Hardware Problem Reporting

The reporting of system problems that are encountered should be done in the AID Office, Room 116 Machinery Hall (x3424), or if the AID Office is closed, report to the Dispatcher in the Dispatch Room on the first floor of Machinery Hall ext. 3-3245.

Be sure to bring terminal output which clearly and concisely demonstrates the problem and the environment in which it took place, as this greatly helps in analyzing the problem.

Another important method for reporting both software and hardware problems is to use an APL program named TROUBLE. You must, of course, know how to use APL to use TROUBLE. After entering the command)LOAD 1 TROUBLE (when signed on to APL), you will be asked to describe your problem. This information is regularly read by SUCC staff and greatly helps in the maintenance of the system.

APPENDIX A

DECsystem-10 CCIS Memos

GI23	Introduction to Timesharing on the DECsystem-10
GI24	DECsystem-10 Availabilities & Capabilities
GI25	DECsystem-10 Glossary
GI26	Index: A Guide to S.U. Computer Systems
GI27	Software Index
SI19	DECsystem-10 Beginner's Guide to Multiprogramming Batch
SI24	Creating Disk Files Using the DECsystem-10 Card Reader
SI25	DECtape on the SUCC DECsystem-10
SI26	DEC-10 Switch Lists
SI27	DEC-10 Reference Sheets
SI28	DEC-10 Compilers & Compile Class Commands
TX1	TECO-Text Editor and Corrector
TX2	SOS-A Line Oriented Text Editor Used on the DECsystem-10
TX3	RUNOFF-Text Formatting Programs
TX4	Summary of RUNOFF Commands
UA28	DDT - Debugging on the DEC-10
UA29	Introduction to the DECsystem-10 Utilities
UA32	PRINT10 Utility for Printing DEC-10 Documents on the IBM/370
LX3	Summary of COBOL on the DEC-10
LX4	LISP Cross Reference
LF18	FLECS: User's Manual
LF19	Comparison of FORTRAN Compilers at S.U.
LF21	DECsystem-10 FORTRAN Switches
SM25	STATPACK - Interactive Statistical Package for DECsystem-10 Version 4
OP1	SUCC Charges

PUBLIC TERMINALS AS OF 1/9/79

ID	TYPE	LOCATION		ID	TYPE	LOCATION	
053	LA37	706	BIOLOGICAL RSRCH. BLDG.	050	LA37	114	LINK HALL
054	LA37	209	BOWNE HALL	051	LA37	114	LINK HALL
05E	LA37	209	BOWNE HALL	052	LA37	114	LINK HALL
088	LA37	209	BOWNE HALL	053	LA37	114	LINK HALL
089	LA37	209	BOWNE HALL	049	LA37	304	LINK HALL
070	LA37	CTL1	BREWSTER-BOLAND DORMS	04A	LA37	304	LINK HALL
08C	LA37	CTL1	BREWSTER-BOLAND DORMS	04B	LA37	304	LINK HALL
05F	LA37	218	CARNEGIE (MATHEMATICS)	04C	LA37	304	LINK HALL
060	LA37	218	CARNEGIE (MATHEMATICS)	090	LA37	116	MACHINERY HALL-AID OFF.
076	LA37	218	CARNEGIE (MATHEMATICS)	367	3277	116	MACHINERY HALL-AID OFF.
07B	LA37	218	CARNEGIE (MATHEMATICS)	042	LA37	119	MACHINERY HALL
083	LA37	218	CARNEGIE (MATHEMATICS)	043	LA37	119	MACHINERY HALL
084	LA37	218	CARNEGIE (MATHEMATICS)	0A0	LA37	119	MACHINERY HALL
0A2	LA37	218	CARNEGIE (MATHEMATICS)	0AB	A980	119	MACHINERY HALL
0A3	LA37	218	CARNEGIE (MATHEMATICS)	0AC	A980	119	MACHINERY HALL
075	LA37	100	DAY HALL	0AD	A980	119	MACHINERY HALL
079	LA37	212	HEROY GEOLOGY BUILDING	363	3277	119	MACHINERY HALL
07A	LA37	212	HEROY GEOLOGY BUILDING	065	LA37	1XX	MAXWELL HALL
044	LA37	116	HINDS HALL	06D	LA37	1XX	MAXWELL HALL
045	LA37	116	HINDS HALL	062	LA37	4XX	MAXWELL HALL
046	LA37	076	HINDS HALL	055	LA37	B107	PHYSICS BUILDING
047	LA37	315	HINDS HALL	056	LA37	B107	PHYSICS BUILDING
048	LA37	315	HINDS HALL	057	LA37	B107	PHYSICS BUILDING
0BA	LA37	B001	HUNTINGTON HALL	058	LA37	B107	PHYSICS BUILDING
069	LA37	315	HUNTINGTON HALL	059	LA37	B107	PHYSICS BUILDING
061	LA37	239	H.B.C. HALL	064	LA37	FL 1	SHAW DORM (MAIN OFFICE)
081	LA37	239	H.B.C. HALL	067	LA37	304	SIMS HALL
08A	LA37	204	LAWRINSON DORM	068	LA37	304	SIMS HALL
08B	LA37	204	LAWRINSON DORM	05C	LA37	210A	SLOCUM HALL
054	LA37	B105	LINK HALL	05D	LA37	210A	SLOCUM HALL
04D	LA37	114	LINK HALL	05E	LA37	210A	SLOCUM HALL
04E	LA37	114	LINK HALL	06F	LA37	210A	SLOCUM HALL
04F	LA37	114	LINK HALL	066	LA37	508	UNIVERSITY PL.(POL SCI)

TYPE	QTY	TERMINAL DESCRIPTION
A980	3	ADDS 980 CRT WITH FULL ASCII SET
LA37	61	DECWRITER II WITH APL/ASCII SET
3277	2	IBM 3277 CRT WITH EBCDIC SET
TOTAL	66	FOR 3 TYPE(S) OF TERMINAL(S)

TIME SCHEDULE OF BUILDINGS HOUSING TERMINALS

BUILDING	MONDAY-FRIDAY	SATURDAY	SUNDAY
BOWNE	7:30AM-11:00PM	8:00AM-1:00PM	CLOSED
CARNEGIE	7:20AM-11:00PM	8:00AM-5:00PM	CLOSED
H.B.C.	6:10AM-11:10PM	6:30AM-5:00PM	CLOSED
HEROY	6:00AM-11:20PM	6:30AM-5:00PM	CLOSED
HINDS	6:20AM-11:00PM	6:30AM-5:00PM	CLOSED
HUNTINGTON	6:00AM-10:30PM	6:35AM-5:00PM	CLOSED
LINK	7:35AM-10:45PM	6:30AM-5:00PM	CLOSED
MACHINERY	8:00AM-MIDNIGHT	8:00AM-7:00PM	9:00AM-MIDNIGHT
MAXWELL	6:00AM- 5:00PM	6:35AM-5:00PM	CLOSED
PHYSICS*	7:30AM-10:30PM	CLOSED	CLOSED
SIMS	7:35AM-10:45PM	6:30AM-5:00PM	CLOSED
SLOCUM	6:30AM-10:45PM	6:30AM-5:00PM	CLOSED

*closed Friday at 5:00PM

All buildings are locked at noon on Saturday during home football games. Also, you will sometimes find some terminal rooms locked. Instructions for obtaining keys are usually posted.

ACADEMIC COMPUTING CENTER DIRECTORY
MACHINERY HALL

Area	Name	Loc. in MH	SU Ext.
Academic Computer Services			
Assistant Director	P. Kent	201	3160
Accounting			
Establishing an account, billing, changing passwords	L. Eberline	121	2775
Establishing an APL account	J. Quackenbush	122	2516
AID Office			
Supervisors	M. Boisvert	218	2816
	B. Riddle	206	3997
Consultant on Duty	AID Staff	116	3424
Course Assistants		118	3814
A programming error problem, or suspected system or operator error	AID Staff	116	3424
Arrangements for special short courses to be taught	J. Koegel	203	3992
Bulletin Board	J. Quackenbush	122	2516
Computer Dial Telephone Access			1313
Computing Center Director	W. J. Jones	250A	2677
Administrative Assistant	V. Sheehan	250C	2677
Computing problems, referral to CC Senior Staff members	AID Staff	116	3424
CCIS Memos			
No Charge	AID Staff	116	3424
Priced	K. Kelly	122	3991
General Information	J. Joy	250	3145
Disk Dataset Registration	AID Office	116	3424
Dispatcher	Staff	102	3245
Incentive accounts for students, faculty and staff	J. Quackenbush	122	2516
Keypunching service	K. Kelly	122	3991
Newsletter and SUCC Notes	J. Koegel	203	3992

Area	Name	Loc. in MH	SU Ext.
Operations			
Assistant Director	F. Martel	205	3993
Operations Manager	W. Eastwood	120	2133
IBM System 370 Operator		102	3245
DECsystem-10 Operator		102	3245
Physical facilities, Computing Hardware	P. Von Esch	125	3813
Public Keypunch Facilities	Dispatcher	103	3245
Public terminal lists, Short course schedules		122	Hallway racks
Purchasing Computing Center memos, packets, coding forms, templates	K. Kelly	122	3991
Registering for a short course	K. Kelly	122	3991
Reporting terminal failure and requesting supplies	Dispatcher	103	3245
Submitting batch jobs	Dispatcher	103	3245
Systems Engineering	W. Hooper	124	4325
Systems Programming			
Assistant Director	D. Hanley	221	3831
IBM System 370/155	E. Okun	204	3244
DECsystem-10	G. Malling	222	4111
Tape Librarian	J. Quackenbush	122	2516
Terminal Room		119	
Tours	Main Office	250	2677
User Program Libraries			
IBM System 370/155	M. Boisvert	218	2861
DECsystem-10	J. Thornton	206	4324
APL	M. Boisvert	218	2861
Using reference manuals	AID Staff	116	3424
Using video display terminals	AID Staff	116	3424

1/9/79

SYRACUSE UNIVERSITY COMPUTING CENTER

INFORMATION SERIES

DOCUMENT	TITLE
GI26-1214	INDEX: A Guide to S.U. Computer Systems
DATE	
01/15/79	
AUTHOR:	D. Kerr
UPDATE:	J. Thornton

AUDIENCE: All Users

COMPUTER(S): IBM 370/155 & DECsystem-10

LEVEL: Introductory

REFERENCES: Supercedes GI26-1142, titled as above.
1. CCIS Memo GI27, 'Software Index'

ABSTRACT:

This memo describes the use of the INDEX system. The purpose of the INDEX system is to provide an easy way to find out what computer programs and packages are available for public use at SUCC. It also points you to documentation for these programs. The INDEX system is designed so that you may get information either about a particular program or package or about a general subject topic.

5 Pages

$0.00

INDEX: A GUIDE to S.U. Computer Systems

INTRODUCTION

The purpose of the INDEX system is to provide an easy way to find out what computer programs and packages are available for public use at SUCC. It also points you to documentation for these programs. The INDEX system is designed so that you may get information either about a particular program or package or about a general subject topic.

USING THE INDEX SYSTEM

The INDEX system is available through any DEC-writer or ADDS terminal on campus. To begin using the INDEX system, turn on the terminal and press return. A line should print indicating that you are now in contact with the Syracuse University Timesharing System. Next, type the word INDEX, and press return. You will then be asked whether or not you want instructions. Type either YES or NO and press return. If you request instructions, they will print, followed by an * prompt. If no instructions are requested, only the * prompt will print. Once the * prompt has printed, you may begin your request for information. You should enter either the name of a specific computer program, or the name of a particular subject about which you would like information. The entry STOP, in response to the * prompt, will cause program execution to stop. Remember that you must press return after each line you type to the computer.

The following is an example showing how to use the INDEX system to request information about STATISTICS. Information typed by the computer is underlined.

042) SU TIMESHARING 10.42.53 MON 01/15/79 V1M34
/index
INDEX
DO YOU WANT INSTRUCTIONS? NO
*STATISTICS

At this point the information about the statistical packages available at S.U. will be printed.

INDEX SYSTEM OUTPUT

Four lines of output will be printed at your terminal for each INDEX entry which matches your request for information. At the beginning of the first line is the name of the program or package. If the reference is to an APL workspace or function, either WS or FN will appear in parentheses after the name. The rest of the first line of output is a short description of the program or package.

The next two lines contain information on language, machine, assistance and support. LANGUAGE defines the programming language you need to know in order to use the package or program being described. For example, to use the plot program FPLOT, you need to know APL, so APL is listed as its language. Most self-contained packages will have the package name as the language since no knowledge of anything beyond the package itself is required.

The entry titled MACHINE indicates on which computer a program or package is stored. Some programs are on the DEC-10, some on the IBM 370/155, and some are on both.

The purpose of the ASSISTANCE category is to identify the department responsible for the program or package, and to give a phone number to call for help. Assistance references to the AID Office indicate that an entry is the responsibility of the Computing Center.

The SUPPORT category defines the level of assistance you will receive from the department responsible for the program or package. The INDEX System Support Levels are:

1. FULL SUCC SUPPORT - maintenance, updating and documentation by SUCC. Staff will investigate problems, if necessary. Also, AID Office assistance and Senior Staff referral.

2. AID & STAFF CONSULTATION - but no maintenance by SUCC (for any software, updates that are provided to SUCC will be installed).

3. SOME CONSULTATION - help is available as time allows, if the problem seems fixable and not too difficult.

4. ENTRY LEVEL HELP ONLY - AIDs will consult only on the basics of getting started.

5. DOCUMENTATION ONLY - documentation is available for reference in the AID Office.

6. NON-SUCC SUPPORT - the department or person indicated by INDEX may be expected to attempt to answer simple questions not answered by on-line documentation files.

The final line of output provides information about documentation. That is, it tells you where you can find more documentation for this program or package. References may be made to manuals, texts, information files, CCIS memos, the APL Documentation System (ADS), etc. An indication is also given as to where to find the documentation (i.e., bookstore, 122 Machinery Hall, 116 Machinery Hall, etc.). All CCIS memos are available either in Room 116 Machinery Hall, (free memos) or in 122 Machinery Hall (those for sale).

SAMPLE OUTPUT

Suppose you used the INDEX system as shown in the previous example, to request information about the statistical packages at S.U. Your request and the program output would look like this:

```
*STATISTICS
APL STATPAK                          A COLLECTION OF APL STATISTICAL FUNCTIONS
   LANGUAGE:APL                          MACHINE:IBM/370
   ASSISTANCE:AID OFFICE X3424           SUPPORT:FULL
   DOCUMENTATION:ADS

BMD                                  STATISTICAL PROGRAMS, INCLUDING ANOVA
   LANGUAGE:BMD                          MACHINE:IBM/370
   ASSISTANCE:AID OFFICE X3424           SUPPORT:AID & STAFF CONSULTATION
   DOCUMENTATION:BMD USER PACKET (AVAILABLE IN ROOM 122 MH)

IMSL                                 MATHEMATICAL, STATISTICAL SUBROUTINES
   LANGUAGE:FORTRAN                      MACHINE:IBM/370
   ASSISTANCE:AID OFFICE X3424           SUPPORT:FULL SUCC SUPPORT
   DOCUMENTATION:CCIS MEMO SM20, IMSL MANUAL (AID OFFICE)

SPSS                                 STATISTICAL PACKAGE FOR THE SOCIAL SCIENCES
   LANGUAGE:SPSS                         MACHINE:IBM/370
   ASSISTANCE:AID OFFICE X3424           SUPPORT:FULL SUCC SUPPORT
   DOCUMENTATION:SPSS MANUAL (BOOKSTORE), SPSS USER'S PACKET

STATPAK                              PACKAGE FOR INTERACTIVE STATISTICAL ANALYSIS
   LANGUAGE:STATPAK                      MACHINE:DEC-10
   ASSISTANCE:AID OFFICE X3424           SUPPORT:FULL SUCC SUPPORT
   DOCUMENTATION:CCIS MEMO SM25
*
```

When the * prompt reappears, you may request information on another topic or enter STOP to exit from the program.

COMMANDS

There are several commands which may be used to alter the output from the INDEX system. These commands and their functions are as follows:

SHORT	Sets the output format to short form so that only program name, language, and machine are printed for each entry.
LONG	Sets the output format to long form. This is the default value.
LIMIT:n	Where n is a number from 1-30, causes INDEX to consult you before typing information for any topic which has more than n entries. The default value for LIMIT is 5. The limit setting is ignored when the short form of output is requested.

LPT	If you are using INDEX while logged into the DEC-10, this command will cause the output to be sent to the Line Printer rather than the terminal.
TTY	Causes output to be typed out at the terminal. This is the default.
HELP	Asks a question which allows you to choose between listing these commands or getting information on the program HELP.
STATUS	Types the current status of the command options.
SWITCHES	Describes how to use the commands as temporary switches.
COMMENT	Allows you to enter a one line comment about INDEX which will be seen by SUCC staff. This can be used to comment on an entry or to suggest an entry that is missing.
STOP	Stops program execution. Program execution is also stopped if you press return immediately after the prompt.

The commands SHORT, LONG, LPT, TTY and LIMIT may be entered one per line following the prompt, or they may be entered on the same line as a request for information. When entered following a request for information, the command must be preceded by a /; when they are entered following the prompt, no / is used. Commands entered with a request for information are temporary switches; that is, they affect only that one specific request. Commands entered on lines by themselves are permanent; they will remain in effect for all subsequent requests for information, or until canceled by another command. For example:

```
*STATISTICS/SHORT
```

will cause the output to be printed in short format for this request only, while

```
*SHORT
*STATISTICS
```

will cause the output for STATISTICS and for all subsequent requests to be in short format. To resume printing with the long format, enter LONG.

INDEX and the DEC-10

The INDEX system may also be used when you are logged into the DEC-10. To use the INDEX system when logged in, give the INDEX command to the monitor and proceed as described in this memo. When the DEC-10 is not available, you may not use the INDEX system at all.

MAINTAINING THE INDEX SYSTEM

SUCC staff attempts to keep Computing Center INDEX entries current and depends on information from the various departments to keep departmental entries current. If you find an entry which is incorrect, or know of a program or package which should be included in the INDEX system, please use the COMMENT facility of INDEX or notify the AID Office at 116 Machinery Hall, Extension 3-3424.

SWITCH: AID TO WS FULL PROBLEMS IN APL USING ⎕FX AND ⎕CR

LA29-1265

AUTHOR: DANA E. CARTWRIGHT

PUBLISHED: NOVEMBER 30, 1979

LEVEL: INTERMEDIATE

COMPUTER(S): IBM 370/155

SUPERSEDES: LA29-1051, LA29-1183, TITLED AS ABOVE

4 PAGES

PRICE: $0.00

SYRACUSE UNIVERSITY ACADEMIC COMPUTING CENTER

This memo is directed to APL users who have such large program packages that WS FULL errors are troublesome. We describe a scheme whereby an existing workspace may be easily modified so as to store some or all functions from it in the SHARP APL File Subsystem, thus making storage space previously occupied by those functions available in the workspace. In general, a package so modified will be considerably slower and will cost somewhat more to run than before. It is difficult to be specific as to the precise impact upon an arbitrary workspace, so users are advised to experiment with this approach before committing to it heavily.

The following characteristics of the approach are claimed:

1. No knowledge of ⎕*CR*, ⎕*FX* or the file system is needed.

2. Any function (with one exception, discussed below) may be moved into the files. In particular, one can take an existing workspace, written without planning for the possibility of file-resident functions, and apply this approach to it with a high level of confidence that the package will continue to operate correctly.

3. The normal APL function editor is used to edit file-resident functions.

4. At any time, all file-resident functions may be moved back into the workspace, subject of course to WSFULL restrictions.

5. Each file-resident function has an "anchor" function in the workspace which occupies less than 100 bytes of storage.

6.)FNS lists all functions of the workspace, whether or not they are file-resident.

Readers not familiar with ⎕*CR* and ⎕*FX* may wish to skip to the section entitled "Using the System."

Method

To make an APL function file resident, we divide it into two parts: an "anchor" function, which remains in the workspace, and a canonical representation matrix of a "body" function, which is stored as a component of a file. For example, the function

```
     ∇ R←MYFUN G;I;J
[1]    I←?G
[2]    J←I|G
[3]    R←I×J×G
     ∇
```

has an anchor function

```
     ∇ R←MYFUN G;Δ
[1]    INVOKE n
[2]    Δ
     ∇
```

where "n" is a file component number assigned to this function, and a body matrix,

```
Δ;I;J
I←?G
J←I|G
```

The "anchor" function is formed by taking the original function header minus local variables, adding a local variable "Δ", and attaching two (standard) new lines. The "body" matrix is formed from the canonicalrepresentation matrix of the original function by changing the first row to be "Δ", followed by the original local variable list.

To see how this new function/matrix pair is executed, we observe that the function INVOKE simply locates (in the appropriate file) the "body" matrix which was formed from the function named by its argument (in the case "MYFUN"), reads it into the workspace, and applies □*FX* thereto, always producing a function named "Δ". The second line of the "anchor" function then executes "Δ".

Claim: The "anchor/body" function pair executes equivalently to the original function.

We observe that syntactically, the "anchor" function is identical with the original function. Therefore, any proper call on the original function will perforce be a proper call on the "anchor". The INVOKE function simply brings into creation a "body" function identical to the "body" matrix already described, and the second line of the "anchor" function executes this new function. We note, however, that there are differences between the environment in which the "anchor/body" functions execute and that of the original function:

1. The name "Δ" has been added to the environment. Clearly this name should not be used for any other purpose in the workspace. Also note, however, that any valid APL identifier may be used instead, and it does not have to be the same for each function -- merely it must be consistent within the "anchor/body" pair.

2. The SI stack is twice as deep as it normally would be.

3. An additional file tie is in existence, this being the file containing the "body" matrices. INVOKE could UNTIE its file just before exiting, but this would considerably slow things down.

4. If the original function sets either its own trace or stop vector, it will now be setting it on the "anchor" function instead of the "body" function.

A note on efficiency: Recursive functions will work under this scheme, but they are very slow.

File Structure

Persons not familiar with the SHARP APL File Subsystem may wish to omit reading this section as it is not essential for using this system.

A single file is used for storage of filed functions. Component 1 is not used, following the convention of always reserving it to contain a natural language description of the file. Component 2 is a character vector which begins with a carriage return character (CR), followed by zero or more names of functions separated by the CR character. The first identifier in the vector names the function whose "body" matrix is contained in component 3 of the file.

Using the System

Please note that all functions and variables described herein may be copied from the public workspace CRFX in library 18. In particular, there is a group named SPACEGROUP which contains all functions, variables, and keywords which are necessary for the package.

It is, of course, necessary to have a file in which the "body" canonical representation matrices will be stored. Thus a file quota of at least 1 is necessary for the APL account which will be used. A file reservation limit of at least 50,000 bytes is also required. (This can be shaved to a certain extent.) Run the function INITIALIZE to create the file and initialize it. The name to be assigned to the file will be requested. I follow a practice of assigning the same name to the file as the workspace with which it will be used, since one can think of the file as just an extension of the workspace. The global variable SWAP is created by INITIALIZE, holding the name selected for the file. After the file is established, the functions INITIALIZE and REPLY may be erased.

Functions are changed into "anchor/body" pairs (and vice-versa) by the SWITCH function. It takes as its argument a character scalar or vector naming the function which is to be operated upon. If the function exists solely in the workspace, then it is disembodied, the canonical representation matrix of its body going to the file, and the function itself being replaced by the appropriate "anchor" function. Conversely, if the designated function was previously SWITCHed into the file, then the "anchor" function and "body" matrix are merged to regenerate the original function.

SWITCHing a function is similar to editing it with respect to whether a)SAVE should be done afterwards. If a SWITCH is done and the workspace is not saved then the next time the workspace is loaded it will appear as if the SWITCH had never been done. The effects of SWITCH are made permanent by following it with a)SAVE. A double SWITCH is permanent, however, if the function was in the file to begin with. Specifically, if a SWITCHed function is brought into the workspace for editing and is then SWITCHed back into the file, the effects of the editing are permanent, regardless of whether or not the workspace is saved.

A function is edited by SWITCHing it into the workspace (if necessary) and then using the normal APL function editor. In general, it is convenient to leave the function in the workspace while debugging proceeds, only moving the function into the file when all editing is finished.

If it is desired to count the invocations of file-resident functions, then the lamp symbol should be deleted from line 5 of the function INVOKE, and a vector called COUNTS should be initialized with as many elements as there are file-resident functions. Displaying the second component of the file will reveal the names of all file-resident functions: the elements of COUNTS are considered to be in the same order as the names.

APLIN UTILITY

LA12-1279

AUTHOR: DANA E. CARTWRIGHT, 3rd

PUBLISHED: JANUARY 24, 1980

LEVEL: INTERMEDIATE

COMPUTER(S): IBM 370/155

SUPERSEDES: LA12-0774, TITLED AS ABOVE

6 PAGES

PRICE: $0.00

SYRACUSE UNIVERSITY ACADEMIC COMPUTING CENTER

PURPOSE

This utility may be used to move data existing in the 370 batch environment into SHARP APL files. Specifically, any OS data set conforming to certain restrictions detailed below may be translated and reformatted into a sequence of APL matrices which become the components of a SHARP APL file.

INPUT DATA

The data set to be moved into APL must be accessible sequentially and is preferably composed of fixed-size records. Variable-length records may be used but with due regard for the manner in which they will be read by the utility (discussed in detail in a later section). The data within a record must be so placed that it can be processed by a format control string similar to that employed in a Fortran FORMAT statement. This requires that the data be in fixed locations within a record, and that the records all be alike in this respect.

CONTROL DATA

The control data set consists of a set of APPEND statements which will control the operation of the utility. An APPEND card is written as follows:

'libnum fileid:passnum' APPEND shape RHO 'format'

The fields are defined as follows:

'libnum fileid:passnum'

This field identifies a SHARP APL file called fileid and located in library libnum. The file must give "⎕APPEND" and "⎕READ" access to user 1000 under the passnumber passnum. The file must exist at the time the utility is run. The file may be tied non-exclusively at a terminal while the utility is running. A full tie at a terminal will cause APLIN to abend.

shape

This field is a numeric vector specifying the shape of the components (as one would specify on the left-hand side of the APL operator ρ) to be appended to the file.

'format'

This field supplies a Fortran-like formatting string which is used by the utility to translate the OS data into APL data. The details of writing formats is given below.

UTILITY OPERATION

Using the information on the APPEND card(s) the utility constructs a pseudo-workspace containing a matrix corresponding to each APPEND statement. Call these matrices M1, M2, ... ,Mn. The shape of each matrix is just the shape specified on the corresponding control card.

The operation of the utility may then be described approximately as follows:

A: Read a record from the input data set.

B: Using the format string specified on the first APPEND statement convert the record into an APL vector (of a numeric or character type as appropriate). This vector then becomes a new row in M1.

C: Step B is repeated for M2,...,Mn. Note that this sequence is carried out with the same data read in step A.

D: Steps A, B, and C are carried out until one or more of the matrices becomes full.

E: When a matrix fills up it is appended to the appropriate file. The matrix is then cleared and the processing of the matrices is resumed.

F: The above steps are carried out until the input data is exhausted. At this time all partially filled matrices in the pseudo-workspace are forced out to their files.

FORMAT STRINGS

Valid format types are: E, F, I, U, A, and X. These function as follows.

Ew.d

w characters from the input are converted to a single APL number. There are d digits to the right of the implied decimal point unless the input field contains an explicit decimal point. A scaling factor of the form En, E-n, or E+n may appear in the input.

Fw.d

like Ew.d except no E may appear in the input.

Iw

w characters from the input are converted to a number (but no decimal point may appear in the input, only digits).

U1 and A1

both these formats take a single character from the input into an APL character. The U1 version does absolutely nothing to the character except move it, while the A1 form also performs a translation from EBCDIC to Z-code (EBCDIC=Extended Binary Coded Decimal Interchange Code, used extensively on the IBM 370, and Z-code is the character coding used internally in APL).

X

causes the next character in the input to be ignored.

Placing a repetition count in front of any format causes that format to be repeated that many times. Examples of formats:

E13.4 E9.0 2F15.1 3I6 17A1 80U1 1X 15X

Both the repetition count and the various field width specifications must be decimal numbers less than 256.

SPACE CONSIDERATIONS

The A and U formats create character-type APL structures exclusively. E and F create only real, double-word types. Mixing E and F with I type fields also causes generation of real results. I fields used by themselves and restricted to a field width of 9 or less generate integer, single-word results.

EXAMPLE

In this example it is assumed that a card deck exists which is to be placed into a SHARP APL file. Each card in the deck consists of a 10 character field occupying the first 10 columns of the card and 6 integers, each occupying successive 4-column fields. It is desired

to place these cards into two separate files, one containing the alphabetic labels, the other the numeric fields. Specifically, one file, called LABELS, is to be composed of matrices, 20 rows by 10 columns, where each row will be a label from a card, and a second file, called NUMBERS, will consist of 20 by 6 matrices, where each row of 6 elements will be 6 integers taken from a single card. The following outlined steps will carry out this objective.

CREATE FILES

To create the SHARP APL files the following lines of APL might be executed at a terminal. Assume the APL account number of the user is 12345678.

```
'LABELS' ⎕CREATE 1
(1 3⍴ 1000 ¯1 987) ⎕STAC 1
'NUMBERS' ⎕CREATE 2
(1 3⍴1000 ¯1 756) ⎕STAC 2
⎕UNTIE 1 2
```

Note that the second line above places the following matrix into the "access matrix" of the file called 'LABELS':

```
1000  ¯1  987
```

RUN UTILITY

The following job is run to transfer the card deck into the SHARP APL files created above.

APL MUST BE UP WHEN THE UTILITY IS RUN!

```
//JOBNAME  JOB  ..........REGION=200K
//STEP1 EXEC PGM=APLIN
//FT01F001 DD *
  '12345678  LABELS:987' APPEND 20 10 RHO '10A1'
  '12345678 NUMBERS:756' APPEND 20 6  RHO '10X,6I4'
/*
//FT01F002 DD *
(insert card deck of data here)
/*
//F
```

```
/*
//FT03F001 DD SYSOUT=A,DCB=(RECFM=UA,BLKSIZE=133)
//
```

The following output will be received from the utility.

```
APLIN UTILITY

CONTROL CARD ANALYSIS

  '12345678  LABELS:X' APPEND 20 10 RHO '10A'
                    FORMAT WILL USE    10 BYTES OF EACH OS RECORD
                      --AND WILL PRODUCE   101 ITEMS FOR APL

  '12345678 NUMBERS:X' APPEND 20 6  RHO '10X,6I4'
                    FORMAT WILL USE    34 BYTES OF EACH OS RECORD
                      --AND WILL PRODUCE    6 ITEMS FOR APL

EOF ON CONTROL DATA SET

MAX READ=    34

BSS CONNECTION ESTABLISHED

PRESENT FILE SIZE INFO    FIRST COMP  NEXT  BYTES USED  MAXIMUM

12345678 LABELS              1         1     0          50184

12345678 NUMBERS             1         1     0          50184

NEW FILE STATUS

12345678 LABELS              1         5     1836       50184

12345678 NUMBERS             1         5     3060       50184

FILES UNTIED - NORMAL UTILITY TERMINATION
```

Note that 4 components were written to each file (the next available component is 5, not the last one written). This also implies that exactly 80 data cards were read. If the number of cards read is not exactly enough to fill the matrices an integral number of times then the last component in each of the affected files will be reduced in size as necessary. In the above example, suppose that only 75 data cards were present in the input deck. Then only enough data to fill 15 rows of the "last" matrices would have been available. In that situation the following message would have been printed:

```
12345678 LABELS        SHAPE OF LAST COMPONENT=  15 10

12345678 NUMBERS       SHAPE OF LAST COMPENENT=  15  6
```

The message "FORMAT WILL USE ..." refers to the number of card columns which will be processed by the format statement appearing on the APPEND card, and how many APL-type values will be produced thereby. Note that after all of the control deck has been processed the utility picks the largest input value and prints it under th e message "MAX READ". This is the number of bytes which will be read from the data set each time a read is performed (see step "A" of the program outlined under "UTILITY OPERATION" above). In this example, the smallest read which can be performed on the data set is 80 bytes (the length of a card) so 80 bytes, not 34 , were read each time. If MAX READ is greater than the length of the shortest record then an I/O error with subsequent ABEND will result.

RUN TIME ESTIMATES

In the example as given above, a cost of about $1.35 per thousand data cards will be incurred. Note that this figure is subject to wide variations depending upon the exact formats chosen, size of matrices employed, etc. The "Equivalent Time Estimate" for a job similar to that illustrated needs to be about 12 seconds per thousand cards processed.

EBCDIC TO Z-CODE TRANSLATION

The "A" type of format implies a translation from EBCDIC characters to APL Z-code representation. The exact translation is discussed in CCIS Memorandum LA17. In general, however, the alphabet and numbers are translated properly, as are most of the special characters in common between EBCDIC and APL (,.;:'= _-+*$()/?).

ABEND CODES

An ABEND of 3056 indicates that the SHARP APL files system was not available when the utility was run. Any of the following codes indicate that the APLIN utility attempted to use the system incorrectly, and should be brought to the attention of the author.

3051 3052 3053 3054 3050.

SYRACUSE UNIVERSITY COMPUTING CENTER

INFORMATION SERIES

DOCUMENT	TITLE
GI11-1224	Job Cards and Variable Verb Job Cards
DATE	
03/12/79	
AUTHOR: Douglas S. Carlson	
UPDATE: Paul Dulfer	

AUDIENCE: S/370 Batch Users

LEVEL: Introductory

COMPUTER(S): IBM 370/155

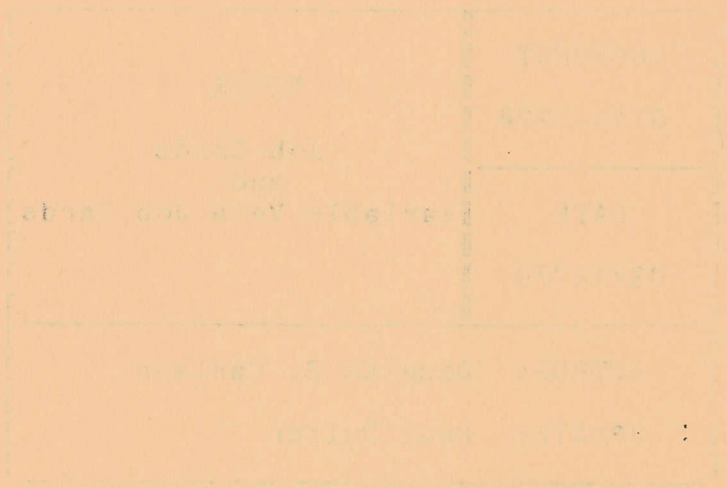
REFERENCES: Supersedes OS24-0656 OS/370 Job Card Rule.

ABSTRACT:

All batch programs that run on the IBM 370/155 require a JOB card or a Variable Verb JOB card as the first card of the card deck. This memo describes how to create these cards.

8 Pages

$0.00



1. Introduction

This memo deals with batch processing jobs, or simply, batch jobs. Batch jobs on the System/370 at this installation comprise all jobs which do not use one of the time-sharing systems (such as APL, ATS or the DECsystem-10).

Batch jobs may be submitted either in card deck form or from a terminal. In either case, once inside the machine, all batch jobs form a queue called the input queue; each job in the input queue is distinguished from the preceding job by the JOB card. Hence, the first card of each job that is submitted for processing must be the JOB card.

2. General form and definition of the JOB card

The general form of the JOB card is:

```
//j    JOB    (a,d,t,l,c,p),'m',REGION=nK
                accounting field
```

The symbols in lower case letters represent specifications to be made by the user and are defined as follows:

j --- the job name which can be any name or string of less than or equal to eight alphameric or national characters starting with an alphabetic or national ($,#,@) character.

a --- the account number (1 to 8 digits) assigned to the general user by the Center when he files a Computer Usage Application, or the mannumber (5 to 8 digits) assigned to a student user by his instructor, or the mannumber (5 to 8 digits) assigned to a programmer by his project director. This parameter is required.

d --- for the general user, the parameter d is the department code or account code (password) determined by arrangement with the Business Manager of the Computing Center. For the student user, the d parameter is the code prescribed by his instructor. This parameter is required.

t --- the estimated maximum equivalent time allowance for the job. The time parameter t may be given in the form aHbMcS where a, b, and c are integers specifying a number of hours, minutes and seconds, respectively. Parts of the specification may be dropped. For example, 7H and 3M45S and 95S are valid time specifications. If two or more parts of the specification appear, each number must be less

than the corresponding conversion factor. For example, 95S and 1M35S are valid, but 1M65S is invalid. The parameter t may also be stated simply as an integer, whereupon this is taken to be the time in minutes. This parameter may be omitted. If it is not specified, then a default value of two minutes is assumed.

l --- the estimated maximum quantity of printed output that can be generated by the job. The lines parameter l may be given in the form of dL, or eK, or f, where d, e, and f are integers. dL is taken to mean d lines of print, eK means e thousands of lines, and f (an integer without any modifiers) is also taken to mean thousands of lines. The parameter l may be omitted, whereupon a default value of 2K lines is assumed.

c --- an integer specifying the estimated maximum number of cards that may be punched by the job. If the cards parameter c is omitted, a default value of 0 (i.e., no punched output) is assumed.

p --- the estimated maximum drawing time needed to produce output on the Calcomp Plotter. The plot parameter p may be specified in the same manner as described above for the time parameter. If the parameter p is omitted, a default value of 0 (i.e., no plotted output) is assumed.

m --- the user's name; this field need not be put in quotes unless a special character (or a blank) is used. The length of this field must be less than or equal to 20 characters. Usually this field should contain the programmer's last name optionally followed by his first name. The name parameter m must be specified.

3. The REGION parameter

The keyword parameter REGION is used to specify the amount of main storage which is to be allocated to the job. If the REGION parameter is not specified on the Job card, the job is assigned a default REGION size of only 50K (K=1024) bytes. Although a few programs, including the IBM utility programs, will run in 50K, most processors at this installation require larger amounts of main storage. REGION requirements for most processors used at this installation are given in the CCIS memo SI14 "REGION Requirements for Common Processors at SUCC". Copies of this memo are available in the AID Office.

The user requests the amount of main storage required for his job by coding

REGION=nK

where n is an even integer. This requests n 1024-byte areas of main storage. Rules for punching the REGION specification are given in section 5 of this memo.

The maximum main storage allocation that can be obtained depends on the time of day at which the job is to be run and these limits may change periodically as the configuration of the supervisor is changed. The job may be submitted at any time, but if, when the job would normally execute, the requested main storage allocation cannot be provided, the job is held in queue until the requested amount of storage is available. If execution of a job is initiated but a request for main storage cannot be satisfied, then the job is terminated with a SYSTEM=804 or SYSTEM=80A error code. A REGION parameter specification for a job step on an EXEC card is ignored at this installation.

4. Additional parameters

In addition, two optional keyword parameters may be specified following the name field. They are:

MSGLEVEL=0	---	to suppress printing of JCL statements.
COND=(specs)	---	to determine bypassing of job steps.

Rules for punching these specifications are given in the next section.

Coding MSGLEVEL=0 on the Job card suppresses the printing of all JCL statements except the Job card and those JCL statements in error. If the MSGLEVEL parameter is omitted, then MSGLEVEL=1 is assumed at this Center which results in all JCL statements being printed on the second page of the output. For more information and other options you can specify using MSGLEVEL refer to the IBM Job Control Language manual, Form GC28-6704.

Coding COND=(specs) on the Job card, where specs are as described in the IBM manual mentioned above, allows the user to determine whether subsequent job steps are to be executed.

5. Rules for punching the Job card

The rules for punching the Job card are:

- // must appear in card columns 1 and 2,
- the job name must begin in column 3,

- the verb, JOB, must appear next (separated from the job name and the accounting field by blanks),
- the set of parameters in the accounting field must be enclosed in parentheses as shown,
- parameters must be separated from each other by commas; no blanks are allowed,
- if certain parameters of the accounting field are omitted and other accounting field parameters follow, then the comma following the omitted parameter must still be punched,
- if REGION, MSGLEVEL or COND are coded they should follow the programmer's name and be separated from it by a comma. If more than one of these are coded they should be separated from each other by a comma,
- blanks are permitted only before and after the verb, JOB, and within the single quote marks of the name field, or following the last parameter on the card, and
- if all the information does not fit on one card, the user may continue onto the next card subject to the usual continuation rules for JCL cards; i.e., he must stop keypunching the card at a comma separating parameters, punch // in columns 1 and 2 of the next card, and begin the next parameter between columns 4 and 16, inclusive.

6. Estimation of parameters

The following considerations should govern the user's estimates for the time, lines, cards and plot parameters in the accounting field. First, the estimates should not be underestimated since the job is automatically cancelled as soon as any one of these estimates is exceeded with the probable results that the run is completely wasted. On the other hand, the estimates should not be excessive overestimates, since jobs with smaller estimates, particularly smaller time estimates are given higher priority in the job queue within the machine. Thus, the safe policy is to specify generous estimates on the early runs and then to refine those estimates for later runs on the basis of the job statistics for the earlier runs which appear on the first page of the printed output. (Warning: Untried programs should not be given large time estimates since they may become trapped in an infinite loop).

The REGION parameter must be coded in most cases, since the default main storage allocation (50K as of April 10, 1973) is not sufficient for most jobs. For more information concerning the

REGION parameter and main storage requirements for the various compilers, utility programs and prepared packages of programs, the user should obtain a copy of the CCIS memo SI14 mentioned earlier in this memo. One may estimate how much storage a FORTRAN program needs for execution by adding 8 to 12K to the size listed in the program's linkage editor map. In order that several jobs may be processed concurrently, the REGION request should be the smallest that is sufficient for the job. However, until the main storage requirements of a job have been determined, the user should request a REGION size that will both accommodate the job and be the largest available for a convenient part of the day.

7. System response to Job card errors

The user's account number or mannumber and the department code or the account code for every job are automatically checked by the operating system. If either is invalid, the job is not run. Accounts are also checked for sufficient funds. A time estimate message "INSUFFICIENT FUNDS" appears on the first page of the output. This is caused by a time estimate that, if actually used, would more than use the rest of your account. This misfortune can be avoided if the user keeps track of the status of his account which appears near the top of the first page of each printed output. Any keypunch error on the Job card also causes the job to fail. Other system error messages follow:

1. INVALID CONTINUATION CARD
2. INVALID DELIMITER
3. MISSING ACCOUNT NUMBER
4. MISSING DEPARTMENT CODE
5. DEPARTMENT CODE EXCEEDS FOUR CHARACTERS
6. TOO MANY ESTIMATES
7. INVALID ESTIMATES
8. INVALID PROGRAMMER NAME
9. PROGRAMMER NAME EXCEEDS TWENTY CHARACTERS
10. INVALID DIGIT
11. NUMBER EXCEEDS NINE DIGITS
12. INVALID OPERAND
13. PARAMETER EXCEEDS EIGHT CHARACTERS

14. NONCE ERROR - this error indicates that you have tried to do something that as of this time is unsupported by SUOS or OS or both. Any other occurrence of this message should be brought to the attention of the AID Office consultant.

15. OPERATOR CANCELLED INPUT

16. READER INPUT DEVICE ERROR

17. MISSING CONTINUATION CARD

18. MISSING JOB NAME

19. JOB NAME EXCEEDS EIGHT CHARACTERS

20. MISSING OPERAND

21. TIME OR PLOT ESTIMATE TOO LARGE

22. EXCESSIVE CONTINUATION

23. TOO MANY SYSIN DATA SETS

Error codes SYSTEM=804 and SYSTEM=80A, which are generated by OS after execution has been initiated, may mean that the REGION specification was incorrect.

If one of these errors should occur when you are trying to run a Job, or for an explanation of what these errors mean, check with an AID Office consultant.

8. Passwords for user account numbers

The user's account number may be passworded to restrict usage of the account number to only authorized users. The account code or password replaces the user's department code on the Job card and during terminal sign-on. The account code is limited to 1 to 4 alphabetic characters. This feature is supplemental to the department code and has no effect on users who do not request an account code.

Those wishing to use account codes should contact the Business Manager, Room 121 MH, x2775, to specify the desired account code. The user should note that it is his responsibility to protect his account number and code from unauthorized persons. These two items appear only on the user's Job card. They do not appear on any printed output or terminal displays produced in the Center.

9. Examples

Example 1.

```
//EX#1  JOB  (234,YUP),JOHNSON
```

The job name for this job is EX#1. Johnson is the programmer. In this example all defaults are in effect. At most he can use two minutes of equivalent time, produce two thousand lines of output. If he exceeds these limits or tries to punch cards or produce a plot, his job will be automatically cancelled by the operating system.

Example 2.

```
//PLOTTEST  JOB  (16820043,JUMP,,,300,20),'JOHN DOE',REGION=120K
```

In this example JOHN DOE has used the default values for equivalent time and number of lines. However, he can now punch up to 300 cards and use 20 minutes of plotter time. Note that the output will be placed in the 'J' bin and not the 'D' bin.

Example 3.

```
//$$MONEY$  JOB  (1234,FIVE,1H10M5S,100L),'GREEN SAM',REGION=300K
```

This job can use a maximum of one hour, ten minutes, and five seconds of equivalent time. He is expecting less than 100 lines of printed output. The output will be placed in the correct bin (last name order).

10. THE SUOS VARIABLE VERB JOB CARD

A new type of JOB card will be available under SUOS as of August 19, 1974. It is called variable verb JOB card. This new type of job card allows you to put the name of any catalogued procedure* in the verb field of the JOB card instead of the verb 'JOB' (thus the name "variable verb"). SUOS will then generate an EXEC card for you and substitute 'JOB' back onto the JOB card. The EXEC card will have the procedure name in both the label field and the operand field (see example below). This JOB card allows you to omit the 'EXEC' card completely. An additional feature of using this type of JOB card will cause SUOS to check a special default REGION library. If the catalogued procedure you are using is included in this library, SUOS will automatically use the nonstandard default REGION found in this library(See Dispatcher or AID Office for a listing of this library) unless the REGION parameter is specified on the JOB card. Therefore, if the procedure is listed in the REGION library, you can omit both the EXEC card and, in most cases, the REGION parameter and your job will run

in an appropriate amount of storage instead of the standard default of 50K. A word of caution is in order here, as in some applications even the nonstandard default REGION is too small. For example, a large PL/C program with several hundred statements may not run in the default REGION of 120K. It is the programmer's responsibility to realize that his/her program is a special case and that the REGION parameter needs to be specified.

One minor restriction has been imposed on the user of this special JOB card. Comment continuation(and only comment continuation)is not acceptable, whereas on a normal JOB card it is.An attempt to do so will result in your job being failed by SUOS with the message NONCE ERROR (which means you attempted something unsupported by SUOS and/or OS). In all other ways the variable verb JOB card conforms to the standard rules of JCL.

Example 1

```
//VVTEST#1   PLC  (1234,ABCD),NAME
```

This job card will expand to the following JCL, and since no REGION is specified, this job will run in 120K (the default for PLC).

```
//VVTEST#1 JOB     (1234,ABCD),NAME
//PLC   EXEC   PLC
```

*See a listing of SYS1.PROCLIB for the procedures available.

Example 2

```
//VVTEST#2  SPSS   (1234,ABCD),NAME,REGION=150K
```

This will expand to:

```
//VVTEST#2   JOB   (1234,ABCD),NAME,REGION=150K
//SPSS  EXEC   SPSS
```

Note that if a PARM must be specified (as is often the case with some programs such as SPSS) you cannot use the variable verb JOB card since the PARM must be entered on the EXEC card.

AID Office

GETTING A DEC-10 FILE PUNCHED ONTO CARDS

As the DECsystem-10 does not have a card punch, a file must be transferred to the IBM370/155 which does have a card punch.

Below is a sequence of events in order to punch out a DEC-10 file. The user must purchase or arrange to use a magnetic tape.

The following instructions are for a disk file named "FOO.DAT" and a tape registered with the number "999999". The tape will be transfered as a "non-labeled" tape.

(1) Obtain a magnetic tape.

(2) Mount the tape on the DEC-10:

```
.MOUNT MTB: FRED: /REELID:999999/VID:'owner 5 MIN'/WE
```

(3) Copy the file to tape, changing the character set as you go:

```
.R CHANGE
>RETAIN              < retains commands until you explicitly execute them.
>FRED:FOO.DAT/MODE:EBCDIC/REC:80/BLOCK:1/INDUSTRY=DSK:FOO.DAT

              { If you make a mistake, retype the line, that's
              { what RETAIN is used for.

>RUN                  < execute the change program.
              { Change messages will appear here
>EXIT
.DISMOUNT FRED:
```

(4) Now run the following batch job on the IBM370:

```
//PUNCH  JOB  (acct,pswd,,,cards),name,REGION=80K
//       SETUP  TAPE=(999999,NL)  owner
//       EXEC MULT3
//SYSIN  DD   UNIT=(TAPE,,DEFER),
//            VOL=SER=999999,
//            LABEL=(1,NL),
//            DCB=(RECFM=F,BLKSIZE=80)
//
```

(5) Pick up punched output in Rm. 122, Machinery Hall.

If you have any questions, come back to the AID Office. If something is or appears incorrect, please tell us. The AID may modify the above to suit the situation. Thank you.

'8460000 DATA' APPEND N TO RHO

SYRACUSE UNIVERSITY COMPUTING CENTER

INFORMATION SERIES

DOCUMENT	TITLE
UA34-1236	TAPEDUMP
DATE	
05/03/79	

AUTHORS: Linda Webb
Thomas J. Pritchard
UPDATE: Jan B. Wilson
Matthew K. Greene

AUDIENCE: System 370 Users

LEVEL: Intermediate

COMPUTER(S): IBM 370/155

REFERENCES:
1. GI11 "Job Cards"
2. SI12 "The SETUP Statement"
Incorporates 'TAPEDUMP' portion of UA15-1048, "TAPESCAN and TAPEDUMP."

ABSTRACT:

TAPEDUMP is a batch utility which can be used to dump all or part of a disk or tape data set. It can also examine the labels on standard labeled tapes. This document describes how to use the utility, explains the optional parameters, and gives some examples showing how TAPEDUMP can be used.

6 pages

$0.00

TAPEDUMP: A Program to Dump Sequential Data Sets

I. Introduction

TAPEDUMP is a general batch utility program for the IBM System/370 written at and supported by the S.U. Computing Center. It can perform one of three primary functions each time it is invoked:

- Dump the contents of all or part of a sequential file.
- Print the contents of the labels on a standard labeled tape.
- Count the numbers of blocks in a sequential file.

The functions are controlled by optional parameters (discussed below); the default is to print the contents of an entire file. TAPEDUMP can be invoked by a cataloged procedure of the same name.

TAPEDUMP recognizes only physical records, or blocks. It does not differentiate between the logical records which comprise a block.

The TAPEDUMP output is by physical record; the record number and its length are given before each block printed. (Additional messages are produced when applicable.) Each block is printed in the standard ABDUMP format. At the left is the 6-digit hexadecimal offset, from the start of the block, of the first character printed on that line followed by 8 groups of 8 hexadecimal digits representing 32 data bytes from the block. On the right is the character interpretation of those 32 bytes with unprintable characters represented by a period(.).

II. JCL and Deck Setup

General form:

```
//      Job card (See CCIS Memo GI11.)
//  SETUP TAPE=(b,k)
//  EXEC  TAPEDUMP,SER=t,LAR=NL
//
```

Notes:

1. The SETUP statement, which is a special feature of SUOS, is required for tape jobs or the use of non-resident disk volumes. (See CCIS Memo SI12.)

2. b on the SETUP card stands for the tape's bin number. If the tape's bin number (b) is not the same as its serial number (t), both should be included on the SETUP card.

3. k on the SETUP card is to be replaced by either NL for a non-labeled tape or SL for a standard labeled tape.

4. t on the EXEC card stands for the tape's serial number. For standard labeled tapes this number must be the same as the volume serial number in the magnetic tape label, which is not necessarily the same as the number on the outside of the tape reel. For non-labeled tapes the serial number should be the same as the number on the outside of the tape reel.

5. Certain other optional parameters, which are described in Notes (6) through (10) should be coded on the EXEC card as circumstances dictate. They should be coded in any order following the SER parameter, separated from it and from each other by commas, and have no embedded blanks. Coding may have to be continued onto subsequent cards according to the standard JCL rules.

6. For standard labeled tapes the data set name (dsname) for the appropriate file must be coded as

   DSN='dsname'

7. The program ordinarily processes, or begins processing at, the first file on the tape. To process, or begin processing at, some other file (say f), the user should also code the following parameter on the EXEC card:

   FL=f

8. If the tape has no standard label, parameter

   LAB=NL

   should also be coded on the EXEC card.

9. A program control parameter with the form

   PARM='q'

   may also be coded on the EXEC card. It is described in section III.

10. For a data set on disk, the following parameters are needed:

    SER=t,DEV=d,DSN='dsname'

    where t is the volume serial number of the disk pack, d is the device specification number and dsname is the data set name. LAB and FL parameters should not be coded. If the data set is cataloged, the t and d may be omitted, but the SER= and DEV= must still be coded. The parameter would then be:

    SER=,DEV=,DSN='dsname'

11. The default values for the various parameters are:

    LAB=SL,FL=1,DSN=NONAME,DEV=TAPE,DEN=4,TRTCH=,PARM=

III. Program Options - The PARM Field

A. To dump an entire file:

You may dump an entire file by omitting the PARM field on the EXEC card.

B. To dump specific records:

You may restrict the number of records to be dumped by coding on the EXEC statement:

```
PARM='nnTOmm'
```

where nn is the starting record number and mm is the stopping record number. (Note: A record in TAPEDUMP is a block.)

C. To obtain a listing of record sizes and whether an I/O error occurred:

You may inhibit the actual dump by coding on the EXEC statement:

```
PARM='nnTOmm,NODUMP'
    or
PARM=',NODUMP'
```

There are no spaces in the PARM specification. The comma is important.

D. To list all volume and data set labels:

You may use TAPEDUMP to list all volume, data set, and user labels from any standard labeled (SL) tape or standard user labeled (SUL) tape by coding on the EXEC statement:

```
PARM=',LABELS'
```

The comma is important. There are no spaces in the PARM specification.

IV. Program Restriction

A. TAPEDUMP stops processing with the LABELS or NODUMP option for an NL tape when two consecutive tape marks are encountered. TAPEDUMP will continue past an empty data file on an SL or SUL tape. Termination occurs when two consecutive tape marks are found following trailer labels.

V. Examples

A. To dump all of the first file on a standard labeled 9-track tape 324761.

```
//      Job  Card
//  SETUP TAPE=(324761,SL)
//  EXEC  TAPEDUMP,SER=324761,DSN='TESTDATA'
//
```

B. To dump an entire disk data set on a 3330 disk unit.

```
//      Job  Card
//  EXEC  TAPEDUMP,SER=SU0002,DEV=3330,
//            DSN='COMP0935.TESTDATA'
//
```

C. To dump records 10 through 20 from the second file of a 9-track tape (116372) with standard labels.

```
//      Job  Card
//  SETUP TAPE=(116372,SL)
//  EXEC  TAPEDUMP,SER=116372,PARM='10TO20',FL=2,DSN='ABC.F2'
//
```

D. To dump the first 20 records of an 1600 BPI tape (111234).

```
//      Job Card
//  SETUP TAPE=(111234,NL)
//  EXEC  TAPEDUMP,SER=111234,LAB=NL,DEN=3
//            PARM='1TO20'
//
```

E. To list the record sizes for the second file of a non-labeled 9-track tape 3264 and determine whether I/O errors occurred.

```
//      Job  Card
//  SETUP TAPE=(3264,NL)
//  EXEC  TAPEDUMP,SER=3264,LAB=NL,FL=2,PARM=',NODUMP'
//
```

F. To list all volume and data set labels from 9-track tape 205151.

```
//      Job Card
//  SETUP TAPE=(205151,NL)
//  EXEC  TAPEDUMP,SER=205151,PARM=',LABELS',LAB=NL
//
```

G. To dump the first 2 blocks of a cataloged disk data set (volume location unknown):

```
//  Job Card
//  EXEC TAPEDUMP,SER=,DEV=,DSN='MYLIB',PARM='1TO2'
```

APPENDIX

Cataloged Procedure TAPEDUMP

```
//  PROC  DEV=TAPE,SER=NONAME,FL=1,LAB=SL,
//              DEN=4,TRTCH=,DSN=NONAME
//DUMP  EXEC  PGM=TAPEDUMP
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  UNIT=(&DEV,,DEFER),
//             DISP=SHR,
//             VOL=SER=&SER,
//             LABEL=(&FL,&LAB,,IN),
//             DSN=&DSN,
//             DCB=(RECFM=U,BLKSIZE=32760,DEN=&DEN,TRTCH=&TRTCH)
```

SYRACUSE UNIVERSITY COMPUTING CENTER

INFORMATION SERIES

DOCUMENT	TITLE
TX2-1111	S O S A Line Oriented Text Editor Used on the DECsystem-10
DATE	
12/13/78	
AUTHOR:	John Thornton

AUDIENCE: DECsystem-10 Users

COMPUTER(S): DECsystem-10

LEVEL: Part 1 - Introductory
Part 2 - Intermediate

REFERENCES: Supersedes CCIS Memo P21-1082, titled as above.

ABSTRACT:

This memo is in two parts. Part 1 is designed as a brief introduction to the most basic commands of SOS. It is intended that Part 1 will give the novice user enough information to use SOS for file creation and simple editing.

Part 2 is for the more advanced user. It describes all of the SOS commands in some detail.

25 Pages

$0.50

INTRODUCTION TO SOS TEXT EDITOR

CONTENTS

INTRODUCTION TO THE SOS TEXT EDITOR

SOS is a line-oriented text editor that allows users to create and edit files on the DECsystem-10. SOS has three modes of operation: input mode -- for creating new files, edit mode -- for correcting a file line by line, and alter mode -- for editing character by character within a line.

SOS begins by assigning a number to each line of the file. When an editing action is to be taken, the line to be edited is referenced by its line number. SOS commands are simple to use. Normally, a command is a single letter followed by a line number or a range of line numbers. SOS allows you to create new lines, get rid of unwanted lines and correct parts of existing lines.

The SOS monitor command is used to run the SOS editor. The same command is used both for creating and for editing files. If the filename used does not exist in your directory, then SOS assumes you want to create a new file (Input mode). If, however, the file was previously created, then SOS assumes you want to edit the file (Edit mode).

This memo is in two parts. Part 1 is designed as a brief introduction to the most basic commands of SOS. It is intended that Part 1 will give the novice user enough information to begin using SOS for file creation and for simple editing. Part 1 begins with definitions of some words and symbols used in this memo. Then the most common commands of the three modes are explained.

Part 2 is for the more advanced user. It describes several topics and the SOS commands involved. The topics include backup files, paging and pointer movement. There is a great deal of information included in Part 2. Therefore, it may be desirable to study topics as they are needed. The appendices are designed to be quick reference charts for the commands and parameters of SOS.

PART 1

DEFINITIONS

escape — escape is entered by pressing the escape key (labelled ESC) when working at a DECwriter, or by typing control-[(left bracket). NOTE: When the escape key is pressed, the computer will type a $ indicating that escape was entered. However, the $ key does not give the same results as pressing the escape key. All $'s in this memo represent pressing the escape key.

<cr> represents a carrier return, i.e., pressing the RETURN or NEWLINE key. This key should be pressed at the end of each line that is typed, whether a command or input.

<space> means the space bar is pressed.

<tab> means the tab key is pressed.

<bksp> means the backspace (labelled BKSP) key is pressed.

<line feed> means the line feed is pressed or alternatively control-J is typed.

<line number> Any other item that is enclosed in angle brackets is to be replaced by the appropriate value of whatever is described. (All the angle bracket items used in this memo are defined in Appendix A). For instance, <line number> should be replaced by the line number desired. Another example is:

<string> A string is any combination of characters (letters, numbers, blanks, punctuation marks, etc.). A string can contain zero or more characters and is normally ended by pressing the escape key.

prompt SOS generally types out a symbol to indicate that it is waiting for input or for a command (input when the prompt is a line number; a command when the prompt is an *). NOTE: alter mode does not give prompts.

Lines in the file can be referenced by either of the following:

1. <line number> - an integer between 1 and 99999.
2. <line number>:<line number> - to refer to a range of lines.
   100:400 means lines 100 through 400.

INPUT MODE

This mode is used to type in a new file (program, data or text). To enter this mode, give the SOS command followed by an unused filename (for instance, SOS TESTB.FOR). SOS will respond with a line number, after which the next line of the file is typed. To exit from input mode, press the escape key; SOS will then enter edit mode.

To summarize, writing a new file involves three steps:

1. give the SOS monitor command.
2. type in the file, line by line, after prompts.
3. press the escape key to leave input mode.

EXAMPLE: (underlined information was typed by SOS)

```
.SOS ONE.FOR
INPUT: ONE.FOR
00100  <TAB> I=1
00200  <TAB> TYPE 10,I
00300  10<TAB>FORMAT(I5)
00400  <TAB>END
00500  $  (THE ESCAPE CHARACTER)
*
```

EDIT MODE

Edit mode is entered either by giving the SOS command followed by the name of a previously written file, or by pressing the escape key while in input mode. Edit mode will type out asterisks as prompts.

EDIT MODE COMMANDS

P Print (on the terminal) the lines indicated.

```
P100
P100:500
```

I Insert the line indicated. (If the line number already exists, SOS will insert between that line and the following line). NOTE: SOS will continue to insert lines, incrementing by 100, until the next line is reached or until the escape key is pressed.

```
I100
I350
```

D Delete the indicated lines from the file.

```
D100
D700:1000
```

R Replace the indicated lines. Equivalent to deleting the lines, then inserting the same lines.

```
R100 (equivalent to D100 then I100)
R400:700 (equivalent to D400:700 then I400)
```

S Substitute one string for another. The format is S<old string> $<new string>$<line number>, where each string is ended by pressing the escape key. The effect is that the new string is substituted for the old string in the line indicated. NOTE: If the old string occurs more than once in the line indicated, each occurrence will be substituted. For more information on this command, see Chapter IV of Part 2.

E Exit from SOS (return to the monitor) and store the file.

W "Save World," i.e., store the file as it is now, but continue editing.

ES Strip (remove) the line numbers, then exit from SOS and store the file. This command must be used for data files. If not used, the line numbers remaining as part of the file will be treated as data. Note that if any editing is done, the line numbers may be different when SOS is reentered.

A Alter the lines indicated, one by one (enter alter mode).

ALTER MODE

This mode is entered by giving the alter command while in edit mode. While in alter mode, one can move back and forth within the line, changing, deleting and inserting characters in the line. This mode maintains a pointer which points to a character in the line. The pointer may be moved by using any of the first four commands below. The following four commands act on or before the character indicated by the pointer. NOTE: Alter mode does not give prompts, nor does it print the user's commands.

ALTER MODE COMMANDS

<space> Pressing the space bar will move the pointer forward one character.

<bksp> Pressing the backspace key will move the pointer one space backward.

l (the letter l) This command will print the line and return the pointer to the beginning of the line.

ns Skip to the nth occurrence of the next character typed.

```
2sc  (Skip to the 2nd c after the pointer.)
st   (Skip to the 1st t after the pointer.)
3ss  (Skip to the 3rd s after the pointer.)
```

nc Change the next n characters (Following the command type in the n new characters.)

```
ca      (Change the next character to an a.)
2cab    (Change the next 2 characters to ab.)
5cx-y/z (Change the next 5 characters to x-y/z.)
```

nd	Delete the next n characters. d 3d
i	Insert, before the next character, whatever is typed next. inserting when the escape key is pressed.
nr	Replace n characters, i.e., delete n characters (nd), then insert (i).
<cr>	Exit alter mode (return to edit mode).

EXERCISES

The left column below is a series of commands to the various modes of SOS. (The SOS command is, of course, given to the monitor.) The right column describes the results of typing the corresponding lines of the left column. This exercise is designed to help the novice gain experience in the use of SOS.

type this:	to do this:
	INPUT MODE
SOS FILEA.TST	Create a file with 5 lines in it.
AAAAA	
BBBBB	
CCCCC	
DDDDD	
EEEEE	
$	(Press the escape key, not the dollar sign.)
	EDIT MODE
P300	Print the 3rd line.
P300:500	Print lines 300 through 500.
I100	Insert a line after line 100.
ABABAB	
D400	Delete line 400.
ES	Exit, stripping the line numbers off.
SOS FILEA.TST	Reenter SOS (edit mode).
P100:500	Print the whole file. (Note that the line numbers have changed.)
R200	Replace line 200.
ZZZZZ	
D300:400	Delete 2 lines.
I300	Insert new lines 300 and 400.
ABCD	
ABCD	
I700	Insert lines 700,800,. . .
FFFFF	Note that if there is room for the insertion of
GGGGG	another line (incrementing by 100), then SOS
HHHHH	will insert another line.
$	(Press the escape key, not dollar sign.)

R100 123451234512345	Replace line 100 with some numbers.
A100	Enter alter mode for line 100.
	ALTER MODE - Note that commands do not print.
<space><space>	Move to the 3rd character of the line.
<bksp>	Move back to the 2nd character.
l	(the letter l) Print out the line. Return the pointer to the start of the line.
2s3	Skip to the 2nd 3 of the line.
c0	Change it to a zero.
d	Delete the next character.
sl	skip to the next 1.
i000$	insert 3 zeroes before the 1. (End the insertion by pressing the escape key.)
<cr>	End alter mode, return to edit mode.
P100:900	Print the entire file.
E	Leave SOS.

PART 2

CHAPTER I - BACKUP FILES AND EXITING SOS

When SOS is called upon to edit a file, SOS makes a copy of the file. All of the editing that you ask SOS to do is done on the copy. The copy (the edited file) does not exist in your storage area until you ask SOS to store it. When you give the E or ES commands, SOS changes the name of the original file (by putting a Q in its extension). Then it stores the edited version with the original filename.

By leaving the original file stored and working on a copy, SOS has a "backup" file, a file you can go back to if anything goes wrong. There are several variations of the E command which allow manipulation of these two files.

EQ means, "Exit to the monitor and forget any editing that I have done," i.e., leave the file as it was before the SOS command was given. (Quit while you're ahead.) To prevent catastrophes, SOS will ask, "Really?" making sure there is no mistake. If you then respond with anything but "Y" or "YES," SOS will return to edit mode (giving an * prompt). NOTE: If you give the EQ, YES combination for a file you are creating, the file will cease to exist, since you never stored it.

EB is like the E command, except that no backup file is created. Only the edited version exists after returning to the monitor.

E:<filename> This command allows you to make the name of the edited file different from the original. The unedited version keeps its name; the filename given in the command is used for the edited version.

```
E:FILE27.FOR
E:SAMPLE.DAT
```

ED Asks SOS to delete the file entirely, i.e. after this command is processed, the file being worked on will no longer exist.

^C (Control-C) This is another way to exit from SOS. SOS will respond by typing "YES? (type H for help):", asking you to choose one of six options:

```
C ---continue (cancel ^C request).
E ---like E in edit mode.
EQ---like EQ in edit mode.
M ---Return to the monitor. SOS can be restarted
     by typing CONT to the monitor.
H ---type this list of options.
R ---reenter edit mode. (Terminate infinite
     searches, etc.)
```

SUMMARY OF EXIT COMMANDS

E	exit, store the edited file.
ES	exit, but remove line numbers.
E:<filename>	exit, but change filename.
EQ	quit, do not store edited file.
EB	exit, but no backup file.
ED	delete the file and return to the monitor.
^C	problem exit.

CHAPTER II - POINTER MOVEMENT AND LINE REFERENCES

SOS continually maintains a pointer, an indication of which line was last referenced. A period (.) may be used to indicate this current line, i.e., you may print the current line by giving the edit mode command "P." This pointer is moved whenever a command refers to a different line, i.e., P1200 moves the pointer to line 1200. Another way to move the pointer is with the point command. The command consists of a period, followed by a line number. The pointer is moved to the line indicated, i.e., the command ".2700" moves the pointer to line 2700.

You can also reference lines near the pointer by using the formats .-n or .+n to indicate the nth line before or after the current one (.+1 is the next line; .-10 is the tenth line before the current one). Probably the most common commands are P.+1 and P.-1, so SOS has special forms to simplify these commands. Pressing the

line feed key (or control-J instead) prints the next line (equivalent to P.+1); pressing the escape key when in edit mode prints the previous line (equivalent to P.-1). Two other commonly referenced lines are the first and the last lines of the file. Just as the period refers to the current line, an up-arrow (^) means the first line, and an asterisk (*) indicates the last. You can also refer to lines by ^+n or *-n.

Now you can refer to lines of a file in several ways. You can use the line number or one of .^*, optionally with a +n or -n. You can refer to a range of lines by using the format <line>:<line>, where the colon separates two line specifications. One more way to describe a range of lines is to indicate a number of lines beginning with a particular line by using the format <line>!n.

```
EXAMPLES:  P.!5      Print this line and the next four.
           P^:*      Print the entire file (Page 1 of a paged file).
           P.-2:+.2  Print 5 lines beginning 2 before current line.
           D.        Delete the current line.
           D*-1      Delete the next to last line.
           D^!3      Delete the first three lines.
```

SOS simplifies looking at segments of a file with a special form of the print command. The print command with no arguments (i.e., "P") prints the current line and the next 15 (i.e., equivalent to "P.!16").

If you wish to print lines without their line number, we can use any form of the print command with the S option. This option is obtained by adding ",S" to the end of the command. For instance, you can say P100:900,S or P350!15,S or P,S. NOTE: After this command, the asterisk may not print out. Push <cr> to get it.

You may want to move the pointer to the line where a particular word or phrase is. The find command (F) accomplishes this. The format is F<string>$<range>, where<string> is the word or phrase to be found, followed by the escape character, followed by the range of lines to be searched. The command instructs SOS to search for the word or phrase indicated, to print the first line where it occurs, and to move the pointer there. If no occurrence is found, SOS types "%SEARCH FAILS." For example, FWIRE$100:1500 means find the first occurrence of the word "WIRE" in lines 100 through 1500.

Omitting the range from the F command causes the search to cover from the current line until the end of the file, i.e., <range> is assumed to be ".:*". If no arguments are given, SOS will search for the next occurrence of the same <string> as used in the last F command.

The find command also allows multiple search strings. If you wish to search for two strings, include both strings, separated by a carriage return. (The second string is ended by pressing the escape key). This format of the command searches for the first occurrence of one string or the other.

The find command also has several options available. Each option is obtained by following your command with a comma and the appropriate letter (or number), e.g. FABC$^:*,A. The following are the available options:

A Enter alter mode when the line is found, position the pointer immediately before the string that was found.

D Decide mode. After finding a line, SOS will type out "D*", allowing you to then choose the next action to be taken. Responses include <space> (meaning "OK, continue searching", if the, n option below was used); G (meaning "OK, but no more searching"). The other responses are A,I,K,M,R,X with the meanings described in this list.

E Find an exact match of the string being sought. Normally SOS does not differentiate between upper and lower case letters. With this option SOS will not find a match unless it is an exact match.

I Insert a line immediately following the line found.

K Delete the line found.

M Insert a page mark immediately before the line found.

N Only type back the line number of the line found.

n (where n is a number) find n lines containing the string.

R Replace the line found.

X Extend the line found, i.e. add to the end of the line.

SUMMARY OF POINTER MOVEMENT COMMANDS

.<line>	Move the pointer.
F<string>$<range>	Find the string and move pointer.
F<string><cr><string>$<range>	Find one string or the other.
<line feed>	Print the next line (P.+1).
escape	Print the previous line (P.-1).
P	Print 16 lines (P.!16).

CHAPTER III - INSERTION

The insertion of lines in SOS can be a tricky business, but it helps to keep these rules in mind:

1. If the line number indicated does not exist, SOS will insert a line with the number given.
2. If the line number indicated does exist, SOS will insert a line between that line and the next line.

3. If SOS can insert a second (,third,...) line while incrementing by 100 (or current increment, see below), it will continue inserting until another line will not fit. This can occur if several lines were previously deleted near the insertion point.
4. If an insert command is given for the end of the file (i* or equivalent), SOS enters input mode, i.e. continues inserting until the escape key is pressed.

NOTE: These rules apply to the replace command as well.

There are several special formats of the insert command:

I<line>;<increment> — Insert some lines incrementing by the value given instead of by 100.

```
I1200;25
I300;5
```

I<line>,<increment> — Like above but the new increment is permanent, i.e., use the new increment value in any subsequent insert commands.

```
I700,10
I1500,200
```

I<line>;!n — Insert n lines at position indicated. SOS will calculate an appropriate increment.

```
I2650;!7
I100;!20
```

I/n — Insert a page mark at the end of page n and do an I100 for a new page. This command simplifies the insertion of pages, each beginning with line number 100. (See Chapter VIII)

```
I/13
I/2
```

I — If no line number is given, SOS will use the last number used, i.e. SOS will insert a line immediately following the most recently inserted line.

I* — Since * indicates the last line of a file, I* means insert after the last line, i.e. add lines to the end of the file.

SUMMARY OF INSERT COMMAND FORMATS

I<line>	Insert at or after the line indicated.
I<line>;n	Insert with temporary increment n.
I<line>,n	Insert with permanent increment n.
I<line>;!n	Insert n lines after <line>, compute increment.
I/n	Insert a new page immediately after page n.
I	Insert after most recently inserted line.
I*	Insert at the end of the file.

CHAPTER IV - SUBSTITUTION

Rather than replacing an entire line or getting into alter mode to correct the line, the correction can be done by substitution. The substitute command allows you to specify a string of characters that are in error and the string that is correct, then have SOS do the work:

s<old-string>$<new-string>$<range>

where $ represents the escape character. Every occurrence of the old string in the range indicated is replaced by the new string.

For instance, sINAT100:300 specifies that every occurrence of the character combination "IN" in lines 100 through 300 is to be replaced with "AT". You must always take care to give a specific enough <old-string> so that no unwanted substitutions are made. In the example given, each word IN is replaced by AT. However, words like LINE and COPYING (if present) would also be changed (to LATE and COPYATG).

The substitute command allows multiple substitutions in one command. Several <old-string>'s and the same number of <new-string>'s can be included in the command, each terminated by a carriage return except that each group (new and old) is terminated by the escape characters. In this case, SOS will substitute the first new string for each occurrence of the first old string, the second new string for each occurrence of the second old string, and so on. For example: SWERE<cr>WAS$ARE<cr> IS$100:500

With all these various formats, it is easy to goof while typing a substitute command. Since this command includes the escape character, we need a special way of cancelling a fouled up command and that is to type two control-G's (producing two "bells").

There are also several options for the substitute command. These options are used by ending the command with a comma and a letter. SOS normally does not differentiate between upper and lower case characters. The ,E option asks for an exact match of upper and lower case. If ,N is used, SOS will only type out the line numbers of lines that are changed. There is also a decide option (,D). This option says that you wish to confirm each substitution before

it is actually changed. When a substitution is made, the line is printed out with the change, then SOS types "D*" as a prompt and waits for a response. Allowed responses are:

<delete>	meaning cancel this substitution (continue searching the other lines in the range
<space>	meaning allow this substitution (continue searching).
G	meaning allow this substitution (no more searching.
E or Q	cancel the substitution and return to edit mode.
A	enter alter mode for the line (continue searching).

SUMMARY OF SUBSTITUTE COMMAND FORMATS

s<old string>$<new string>$<range>	substitute new-1 for old-1,
s<old string-1> <old string-2> <old string-3>$<new string-1> <new string-2> <new string-3>$<range>	substitute new-1 for old-1, new-2 for old-2,...
s<old string>$<new string>$<range>,N	only type out line numbers
s<old string>$<new string>$<range>,D	confirm every substitution
s<old string>$<new string>$<range>,E	substitute only for exact match

CHAPTER V - COPYING LINES

If a group of lines are found to be in the wrong place in your file, or if a group of lines are repeated in the file (or in another file), then you need to use one of SOS's copy operations. There are three similar but distinct types of copy operations in SOS: transfer a range of lines from one point in a file to another point; duplicate a range of lines from a different point in the file; duplicate a range of lines from a different file.

T<destination>,<range>	Place a copy of the lines indicated at or after the destination line and delete the original lines. If necessary, SOS will compute an appropriate increment. T1800,8500:9300
C<destination>,<range>	Similar to the Transfer command, except that the original lines are not deleted. C1800,8500:9300
C<destination>=<filespec>,<range>	The lines of the file indicated are copied to the destination in

the current file. <filespec> represents the appropriate DECsystem-10 file specification.

C1800=FILE23.DAT,2000:3000

If SOS cannot find an increment that will allow all the lines to be copied, it will copy all the lines, numbering them 100,200,300,....Then the message "incl=order" is printed indicating that there is a line number problem. This problem is straightened out by renumbering the lines (See Chapter VII).

If you cannot remember exactly which lines you want to copy from a different file, SOS will allow you to look at the file before doing the copy. The following format:

C<destination>=<filespec>/s

allows you to enter the second file in READONLY mode. READONLY means that you can print lines or enter alter mode to examine a line, but you cannot use any SOS command which will change the file (SOS prompts with "C*" to remind you). When you know which lines you want to copy, give the E command to return to your editing file. SOS will then ask "Source lines=" so that you can complete the copy operation. If you decide not to copy any lines while reading the file, type EQ to abort the copy command.

SUMMARY OF COPY COMMANDS

T<destination>,<range>	Transfer some lines.
C<destination>,<range>	Copy some lines
C<destination>=<filespec>,<range>	Copy from another file.
C<destination>=<filespec>/S	Copy, but first examine <filespec> READONLY mode.

CHAPTER VI - JOINING LINES

When working with text files, there are customarily times when a line is too long or too short. SOS has a pair of "join" commands which help to solve this problem. One join command is in edit mode and the other is in alter mode.

J<line> (edit mode) Combine the line indicated with the following line. The second line is deleted.

```
J1300
J.
```

j (alter mode) Take all of the characters to the right of the pointer and insert them at the beginning of the next line, making the current line shorter and

the following line longer. After the command is executed, SOS will be in alter mode for the following line with the pointer at the beginning of the line.

SUMMARY OF JOIN COMMANDS

J<line>	edit mode join. Combine two lines into one.
j	alter mode join. Join rest of line to next line.

CHAPTER VII - RENUMBERING LINES

It is sometimes desirable or necessary to renumber the lines of a file. For instance, if a great deal of editing has been done, lines have been inserted and deleted, the line numbers may no longer be as convenient as 100, 200, 300, ... On the other hand, some other regular numbering system may be desired. Either change can be accomplished with the Number commands:

N<increment>,<range>,<first line number>

Number the lines in the range indicated. Begin numbering with the first line number given and increment by <increment>. If no arguments are given (e.g., "N"), SOS assumes standard line numbers are desired (increment = 100, first line number = 100, range is the entire file).

```
N
N100,685:862,700
N5,^:*,5
```

NA<increment>,<range>

Add the increment value given to each of the line numbers in the range indicated. This command is normally used when so much inserting has been done between two lines of the file that there is no more room for insertion.

```
NA50,2005:2100
```

When numbering is done to a paged file (See Chapter VIII), SOS begins each page with <first line number>, that is, the (standard) first lines of each page are each numbered 100. If you would rather have the line numbers continue incrementing over page marks, use the following command:

NP<increment>,<range>,<first line number>

Number the lines in the range indicated beginning with <first line number> and incrementing by <increment>. Do not reset to <first line number> at beginning of pages. i.e., keep incrementing. Again no argument means standard values. (See Chapter VIII for an explanation of the second example).

```
NP
NP10,^/1:*/5,10
```

The normal line numbering scheme for paged files has the first line of each page labelled 100, the second of each page 200, and so on. If a page mark is removed, therefore, the resulting page will have two line 100's, two line 200's, etc. SOS indicates this problem by stating, "%OUT OF ORDER." The cure for this problem is to renumber the lines. (See Chapter VII on renumbering.)

SUMMARY OF PAGING COMMANDS

M<line>	Mark a new page.
K/<page number>	Kill the page mark.
K/<page#>:/<page#>	Kill the range of page marks.

CHAPTER IX - CASE CHANGES

When editing text files, you may need to work with both upper and lower case letters. Normally this is accomplished by giving the SET TTY LC monitor command before entering SOS. When doing just one or two changes, however, it is common to forget to give the command. So, SOS has some new commands to change the case of letters. There are three new edit mode commands and three new alter mode commands. These commands allow you to change lower case letters to upper case, to do the reverse, or to invert the case of each letter encountered. Note that these commands affect only the letters encountered; they have no effect on digits or special characters.

In edit mode, the commands act on each letter in the range of lines that you specify. The command VL, followed by a range of lines, changes all letters to lower case. The command VU changes all letters to upper case. The command VV inverts the case of each letter. That is, any lower case letters encountered are changed to upper case and vice versa.

VL100:200	change all letters in lines 100 through 200 to lower case.
VU700:1000	change all letters in lines 700 through 1000 to upper case.
VV400:600	invert the case of all letters in lines 400 through 600.

In alter mode, there are three analogous commands. However, these commands function character by character rather than line by line. > is the command to change the next character (if it is a letter) to lower case, while < does the reverse. Either of these can be preceded by a number to repeat for more than one character. i.e. 4> means to change all the letters among the next four characters to lower case. If a negative number is used, the command functions towards the left. The invert case command is V, which means: if the next character is a a lower case letter, make it upper case and vice versa. Again a number can be used to repeat the command to the right (or left if negative). For example, -7V means to invert the case of the previous 7 characters.

SUMMARY OF NUMBER COMMANDS

N<increment>,<range>,<first line number>	renumber the lines in <range>.
NA<increment>,<range>	add to each line number.
NP<increment>,<range>,<first line number>	N, but increment at page mark.

CHAPTER VIII - PAGING

SOS allows you to group the lines of a file into pages. This ability is normally used for text files. Another reason for understanding paging is for editing paged files, such as those produced by RUNOFF, the DEC-10 text formatter.

Each page of an SOS file can contain any number of lines, as you have seen from the one-page files you have been dealing with. When editing paged material, line numbers refer to the current page, unless a different page is indicated. The page is indicated by a slash followed by the page number. For instance, p100/2 says to print line 100 of page 2. Any subsequent commands now refer to page 2 until another page number is given. SOS puts a "page mark" at the beginning of each page. The mark is only printed if it is inside the range of lines being printed. Note that ^ and * refer to the first and last lines of the current page, i.e., P^:* will print only the current page. Just as "^", "." and "*" refer to the first, the current and the last lines of the page, /^ refers to the first page of the file, /. to the current page, and /* to the last page. An entire paged file may be represented as ^/^:*/*. If you wish to refer to one entire page (to print, delete, etc.), no line indication is needed, i.e., "P/3" will cause all of page 3 to be printed.

The following commands allow you to separate material into pages or to "unpage" paged material:

M<line>

Insert a page mark immediately before the line indicated. The page number of any lines following the new page mark is increased by 1.

```
M5100
M6375
```

K/<page number>

Delete the page mark indicated. Any larger page numbers are decreased by 1. The page indicated and the one before it are merged into one page. You may also delete a range of page marks with the format:

```
K/<page#>:/<page#>
K/3
K/4:/6
K/10
```

SUMMARY OF CASE CHANGE COMMANDS

edit mode:
VL<range> change all letters in range to lower case.
VU<range> change all letters in range to upper case.
VV<range> invert the case of all letters in range.

alter mode:
(-)n> change all letters among the next (previous)n characters to lower case.
(-)n< change all letters among the next (previous)n characters to upper case.
(-)nv invert case of all letters among the next (previous)n characters.

CHAPTER X - MISCELLANEOUS EDIT MODE COMMANDS

The remaining operations of SOS edit mode are: listing lines on the line printer, extending lines (inserting at the end of the line), and an edit-execute sequence for program files. Some variations of the Save World command are also described here.

Listing. A file or a range of lines will be sent to the line printer by SOS if the L command is given. "L" asks that the entire file be listed, while "L<range>" asks for just a range of lines to be listed. An "S" option allows the suppression of line numbers for the lines listed, that is, "L,S" and "L<range>,S" each specify that a group of lines are to be listed on the line printer without their line numbers.

Extending Lines. A common use of alter mode is to add to the end of existing lines. The sequence of commands might be A100, <tab>(Skip to end of the line), then i (begin inserting). The extend command in edit mode (x) has the same effect as this sequence of commands.

X<range> One by one, print the lines and wait for insertion at the end of each line. You are, at this point, in alter mode, so that pressing the escape key will end the insert and allow alter mode commands to be used.

```
X1250
X.+1!10
```

If you do not want the current line typed out, use the S option, e.g., X300,S. SOS will not type the line but merely wait for input.

Edit-Execute. SOS is very handy for debugging a program, the sequence commonly being: create a program file, execute it, correct it, execute, correct, ... SOS simplifies this process by having an "execute" command in edit mode. After the first execution, you will probably give the monitor command "SOS". (The filename is remembered from the last SOS command.) After all

apparent corrections are made, you may give the edit mode command "G". Just as the execute and SOS monitor commands remember the last file used, the G command instructs the computer to re-execute the last program that was executed (probably the one you are editing).

The command sequence (after the incorrect first execution of a program) could be: "SOS", then whatever editing commands are needed, then "G" to execute the program, repeating until the program works. NOTE; The G command has the same variations as the E command, i.e., GS, GB and G:<filespec>. (See Chapter I for description.)

Save World. The Save World (W) command allows you to store the current version of your file and to continue editing. This command has the same variations that the E command has, i.e., WS, WB and W:<filespec>. (See Chapter I for description.) The W command can be made automatic also. The SAVE option generates an auto-w command after the specified number of edit mode commands. the ISAVE option generates an auto-w command after the specified number of input lines. (See Chapter XI for more description.)

SUMMARY OF MISCELLANEOUS EDIT MODE COMMANDS

L<range>	List the range of lines.
L	List the file.
L,S	List without line numbers.
X<range>	Extend the lines one by one.
X<range>,S	Extend, but do not type out lines.
G (GS, GB, G:<filespec>)	Go and execute.
W (WS, WB, W:<filespec>)	Save world.

CHAPTER XI - MISCELLANEOUS ALTER MODE COMMANDS

In Part 1 of this manual, alter mode commands were classified as three types: pointer positioning, action and finishing commands. Here we introduce five more positioning commands, four more action commands, and three more finishing commands.

Each previously described pointer positioning command has another similar command.

n<space>	Move the pointer n spaces forward (right).
n<bksp>	Move the pointer n spaces backward (left).
p	Type out the line, then return the pointer to the current position.

nw — Skip forward n words. The pointer, after execution will be at the first character of the nth word after the current word.

<tab> — Pressing the tab key causes the pointer to move to the end of the line. If a minus sign is typed first, the pointer moves to the start of the line.

Two of the action commands presented here are alternative methods for replacing and deleting information in the line. Another is a variation of the insert command.

T — Replace one word. This command deletes the spaces to the right of the word, so you may want to reinsert them.

nk<character> — Delete all of the characters up to, but not including, the nth occurrence of the character. -nk<character> will delete to the left of the pointer. If there is no occurrence, the pointer will not be moved.

```
kc    (delete until the first c)
3kt   (delete until the third t)
-5k0  (delete backwards until the fifth 0)
```

nu — Delete n words. More precisely, delete from the pointer to and through the nth group of one or more spaces.

x — Extend the line, i.e. move the pointer to the end of the line and initiate an insert. If this command is preceded by a minus sign, SOS will type out the line number and initiate an insert at the beginning of the line.

Note that nine alter mode commands allow a negative n value to be specified. The nine are D, K, R, S, V, X, <tab>, < and >. Each of these allows correction to the left of the pointer.

SOS has a special feature which allows you to insert a new line while in alter mode. If a <line feed> is entered (by pressing the Line Feed key or Control-J) as part of text for the insert command, SOS will generate a new line number (by incrementing or computing one if necessary). Any additional text will be inserted into the new line. New lines may be continually entered until the escape key is pressed or there is no more room for a new line number. A special form of the alter mode insert commands allows you to specify the increment for the new lines:

ni — Insert text; ended by pressing the escape key. If any new lines are started using the line feed key, use n for the line number increment.

Three additional finishing commands:

e Leave alter mode, but do not print the remainder of the line.
q Quit, leave alter mode, but undo any corrections made.
^u Cancel any corrections just made, but begin the alter again.

SUMMARY OF ALTER MODE COMMANDS

n<space>	Move pointer right.
l	Print pointer to start.
(-)ns<character>	Skip to nth occurrence.
(-)<tab>	Skip to end/start of line.
nc<string>	Change n character
ni<string>$	Insert (increment n).
(-)nk<character>	Delete to nth occurrence.
<cr>	Leave alter mode.
q	Quit, undo corrections.
n<bksp>	Move pointer left.
p	Print, pointer same.
nw	Skip n words.
nu	Delete n words.
(-)nd	Delete n characters
(-)nr<string>$	Replace n characters.
t	Replace one word.
(-)n>	Change n characters to lower case.
(-)n<	Change n characters to upper case.
(-)nv	Invert the case of n characters.
(-)x	Extend the line.
e	Leave without printing.
^u	Quit and begin again.

CHAPTER XII - SWITCHES AND PARAMETERS

SOS has a series of options available. Each of the options may be set as a switch in the "SOS" command. To use a switch, follow the filename with a slash and then the option name, e.g., "SOS FILE27.DAT/EXPERT." More than one switch may be used in one command. The options may alternatively be set while in edit mode. In edit mode the format is to type the underline character (_), followed by the option name. (On ADDS terminals, the left arrow is used.)

There are several options which are set to a value instead of on or off. These are set by following the option name with a colon and the value desired (e.g., increment:10).

OPTIONS:

BAK Create a backup file, i.e., store the unedited version of this file in a file whose extension begins with a Q. This option is the default.

BASIC — The file was written with BASIC (the format of line numbers is different). This is assumed for a file whose extension is .BAS.

BLOCK — Create a line-blocked file. This is the default.

C64 — 64 character set.

C128 — 128 character set.

COMPRESS — Remove form feeds, etc from text to get a "compressed" file.

DECIDE — Confirm any substitution made. Equivalent to the ,D option of the substitute and find commands. (See Chapters II and IV for more explanation.) The default is NODECIDE.

DELETE — Delete the file upon exit from SOS. To prevent tragedies, SOS will ask "Do you really want to delete the file (Y or N)?" before proceeding.

EXACT — Normally SOS does not differentiate between upper and lower case when doing string searches, i.e. the find command for the string THE will "find" The, the, etc. If the EXACT option is turned on, however, SOS will only find an exact match of upper and lower case.

EXPERT — This option causes certain things not to be typed while using SOS. When characters are deleted or replaced in alter mode, they will not be typed out surrounded by slashes as usual. This allows the line to appear as it actually is, without deletion indicators. If an error occurs, just a three-letter abbreviation is typed instead of the entire message. Also no confirmation of questionable commands is asked (e.g., "Massive delete OK?"). The opposite of this option is NOVICE, which is the default.

INCREMENT:n — Change the line number increment to n. Default = 100.

ISAVE:n — Automatically execute a Save World command (W) after every n lines of input. Default = 0 (never).

LOWER — Interpret all letters as lower case instead of upper case. This option does not give both upper and lower case; only the "SET TTY LC" monitor command accomplishes that.

NAME:newname — Change the name of the edited file to the name given. The backup file keeps its original name. Using this option has the same effect as the E:filename command (Chapter I).

NOBAK — Do not create a backup file. The only file will be the edited version. Using this option has the same effect as the EB command (Chapter I).

NOBLOCK — Turn off Block.

NOCOMPRESS — Turn off COMPRESS. This is the default.

NODECIDE — Turn off DECIDE. This is the default.

NODELETE — Turn off DELETE. This is the default.

NOEXACT — Turn off EXACT. This is the default.

NONUMBERS — Do not print line numbers. The lines are numbered as always, but the numbers are not printed. The default is NUMBERS.

NOVICE — Turn off EXPERT. This is the default.

NUMBERS — Turn off NONUMBERS. This is the default.

OLD — This option will cause SOS to store your unedited file with a different extension than the normal backup file (by replacing the first letter of the extension with a Z). If you do a lot of editing, you can store your original file as a Z file, then your normal backup file (Q in extension) is an intermediate version. The Z file remains as the original.

OPTION:option-name — If you have several favorite switches, they may be made defaults (for you) by creating a file named SWITCH.INI with a line in it of the following format:

```
SOS/Switch/Switch...
```

If you have an alternative favorite set of switches, enter a line into SWITCH.INI with the following format:

```
SOS:option-name/Switch/Switch...
```

Now when you enter SOS, you will get your default switches. When you use the OPTION Switch, however, SOS will look in SWITCH.INI for a matching SOS:option-name line and use those switches instead.

PLINES;n — The print command with no arguments (i.e., "p") is normally equivalent to p.!16 (print 16 lines). The number of lines printed is changed to n by this option. Default = 16.

READONLY (or RONLY) (or R)	Enter readonly file. In this mode, any command which would alter the file is illegal. This option cannot be turned off while in SOS. Default is off.
RUN:program-name	Normally, when an edit mode G command is given, the COMPILE system program is run after exiting from SOS. With the RUN option, you can specify a different system program to be run, e.g. RUN:RUNOFF.
SAVE:n	Automatically execute a Save World command (W) after every n edit mode commands. Default=0(never).
SEQUENCE	Turn off UNSEQUENCE. This is the default.
START:n	Change the first line number to n. Default = 100.
STEP:n	Set increment to n. Default = 100.
UNSEQUENCE	Remove line numbers when the file is stored. This option has the same effect as the ES command (Chapter I).
UPPER	Turn off LOWER. This is the default.

SOS will answer several questions about your job's status. These questions are "asked" while in edit mode by typing an equals sign, followed by the "question," e.g., =ERROR.

=.	What is the current pointer location?
=BAK	Is the BAK option on or off?
=BIG	How many pages in this file?
=BLOCK	Is the block option on or off?
=CASE	What case is SOS in?
=COLUMN	Type out numbers to label columns, simplifying the lining up of text.
=DECIDE	Is the DECIDE option on or off?
=DELETE	Will the file be deleted upon exit?
=DISK (or =DSK)	What is the current storage quota information?
=ERROR	What was the last error message given? (print the full message even when in EXPERT mode).
=EXACT	Is the EXACT option on or off?
=INCREMENT	What is the current increment value?
=ISAVE	What is the current ISAVE value?
=LOCATION	What is the first location in the edit buffer?
=NAME	What will the name of this file be when stored?
=PLINES	What is the current PLINES value?
=RUN	Which system program will be after a G command
=SAVE	What is the current SAVE value?
=SEQUENCE	Is the SEQUENCE option on or off?
=START	What is the current START value?
=STEP	What is the current STEP value?
=STRING	What are the most recently used strings in the find and substitute commands?

APPENDIX A

ANGLE BRACKET DEFINITIONS

Several of the items surrounded by angle brackets merely mean to press a particular key on the terminal keyboard:

<cr>	carrier return (RETURN key or NEWLINE key).
<tab>	tab key.
<space>	the space bar.
<bksp>	the backspace key.
<linefeed>	the linefeed key (also CONTROL J).
<delete>	the DELETE key.

The other items are to be replaced with whatever is described:

<line number>	an SOS line number.
<page number>	an SOS page number.
<increment>	the number to add to a line number to get the number for the next line.
<1st line number>	the line number used for the first line of the file or the first line of each page. In the Number commands, the line number for the first line being renumbered.
<string>	any combination of zero or more characters, usually ended by pressing the escape key.
<filespec>	any valid DECsystem-10 filename and extension.
<character>	any valid ASCII character.
<destination>	any line indication telling where to place copied or transferred information.
<line>	a line indicator. Any of: 1. a line number. 2. . (the current line symbol) or .+n or .-n. 3. * (symbol for last line) or *-n. 4. ^ (symbol for first line) or ^+n.
<range>	indication of a line or a group of lines. Any of: 1. a line indicator (See <line>). 2. <line>:<line>. 3. <line>!n.

APPENDIX B

SWITCHES AND PARAMETERS

(For further description, refer to Chapter XII.)

BAK	Create a backup file.
BASIC	File is in BASIC format.
BLOCK	Line-block the output file.
C64	64 character set.
C128	128 character set.
COMPRESS	COMPRESS the output file.
DECIDE	Confirm any substitution.
DELETE	Delete file on exit.
EXACT	EXACT match of upper, lower case.
EXPERT	Work in expert mode.
INCREMENT:n	Change line increment to n.
ISAVE:n	Auto-w on input.
LOWER	Every letter in lower case.
NAME:newname	Rename the edited file.
NOBAK	No backup file.
NOBLOCK	Turn off line blocking.
NOCOMPRESS	Do not COMPRESS output file.
NODECIDE	No confirmation of substitution.
NODELETE	Turn off DELETE option.
NOEXACT	Don't differentiate upper, lower case.
NONUMBERS	Don't type out line numbers.
NOVICE	Return to NOVICE mode.
NUMBERS	Begin typing line numbers again.
OLD	Create Z backup file.
OPTION:option-name	Get set of Switches from SWITCH.INI
PLINES:n	Change number of lines for "p<cr>".
/R	READONLY mode. Can only be a switch in SOS command.
/RONLY or /READONLY	" " " " "
RUN:program-name	Run program after G command.
SAVE:n	Auto-w in edit mode.
SEQUENCE	Store with line numbers.
START:n	Set first line number.
STEP:n	Set line number increment.
UNSEQUENCE	Store without line numbers.
UPPER	Every letter in upper case.

EDIT MODE "QUESTIONS"

=.	What is current line number?
=BAK	Is BAK option on or off?
=BIG	How many pages in file?
=BLOCK	Is BLOCK option on or off?
=CASE	Give case information.
=COLUMN	Type out column numbering.
=DECIDE	Is DECIDE option on or off?
=DELETE	Is the DELETE option on or off?
=DISK or =DSK	What is current storage quota information?
=ERROR	What was most recent error message?
=EXACT	Is EXACT option on or off?
=INCREMENT	What is current increment?
=ISAVE	What is current ISAVE value?
=LOCATION	What is 1st location in edit buffer?
=NAME	What will filename be?
=PLINES	What is current PLINES value?
=RUN	What system program will a G Command run?
=SAVE	What is current SAVE value?
=SEQUENCE	Is SEQUENCE option on or off?
=START	What is first line number?
=STEP	What is current increment?
=STRING	What were the most recently used strings for the find and the substitute commands?

APPENDIX C

EDIT MODE COMMANDS

		Refer to Page:
.<line>	Move pointer.	8
A<range>	Enter intraline edit mode for lines, one by one.	5
C<destination>,<range>	Copy range of lines to destination.	13
C<destination>=<filespec>,<range>	Copy from another file.	13
D<range>	Delete lines	4
E	End, save the file, return to the monitor.	5
EB	E, but don't save a Backup file.	7
EQ	E, but don't store new (edited) file; Quit.	7
ES	E, but strip line numbers.	5
F<string>$<range>	Find string and move pointer there.	9,10
G	Go, execute last file executed.	18
I<line>	Insert a line or lines.	4,10,11
J<line>	Join this line and the following line.	14
K/<page number>	Kill the page mark indicated.	16
L<range>	List (on LPT) the lines indicated.	18
M<line>	Put a page mark before the line indicated.	16
N<increment>,<range>,<start>	Renumber using the values given.	15
NA<increment>,<range>	Add increment to each line number.	15
NP<increment>,<range>,<start>	N, but don't restart numbers at new page.	15
P<range>	Print lines.	4,8,9,16,23
R<range>	Replace lines.	4,11
S<old string>$<new string>$<range>	Substitute new string for old.	4,12,13
T<destination>,<range>	Transfer lines to destination.	13
VL	Change letters to lower case	17,18
VU	Change letters to upper case	17,18
VV	Invert case of letters	17,18
W	Save World, continue editing.	5,19,22,24
X<range>	Extend, add at end of each line.	18

APPENDIX D

ALTER MODE COMMANDS

		Refer to Page:
n<space>	Move pointer n characters right.	5, 19
(-)<tab>	Move pointer to end (start) of line.	20
n<bksp>	Move pointer n characters left.	5, 19
l	Print rest of line; place pointer at beginning.	5
p	Print rest of line; place pointer at current position.	19
(-)ns<character>	Skip to the nth occurrence of <character>.	5, 20
nw	Skip forward one word.	20
nc<string>	Change n characters to those indicated.	5
(-)nd	Delete n characters.	6, 20
(-)nk<character>	Delete to the nth occurrence of <character>.	20
i<string>$	Insert the string (end with escape).	6, 20
(-)nr<string>$	Replace n characters (end with escape).	6, 20
nu	Delete n words.	20
t<string>$	Replace one word (end with escspe).	20
j	Stick remainder of line at start of next.	14
(-)x	Extend, insert at end (Start) of line.	20
(-)n>	Change n characters to lower case.	17
(-)n<	Change n characters to upper case.	17
(-)nv	Invert case of n characters.	17
<cr>	Exit from alter mode.	6
q	Quit, cancel any editing.	21
e	Exit, but don't print rest of line.	21
^u	Quit and start again.	21

TO USE THIS DISPLAY TERMINAL TO LIST THE QUEUE

1) Sign on using your BATCH Acct. number. To do this, type:

)ON nnnnnnnn:password

    Where nnnnnnnn is your BATCH Account Number, and password is your BATCH password.

2) To list the queue type the display command

    DQ15J

    exactly as it appears here. This will display the queue one page at a time. To get a new page hit enter.

3) After you have finished listing the queue, you must sign off. To do this, simply type

)OFF

If you have any trouble entering commands, hit the RESET button on the display terminal.

'QUE FILENAME/NAME:"~~"

DECSYSTEM-10 STORAGE QUOTAS

When you store files on the DECsystem-10, the monitor records the size of each file for you. The size is measured using a unit called a "disk block" (or simply block), which is equal to 640 characters of information. When you ask for your directory (using the "DIRECTORY" monitor command), each file is listed like the following:

```
PROG1 FOR     2      <057>   19-OCT-76
```

The 2 which appears in this example is the number of blocks needed to store the file PROG1.FOR.

When the DECsystem-10 stores a file, however, it sets aside (i.e. "allocates") a certain amount of storage, normally 5 blocks. If this allocation is used up, another 5 blocks are allocated. When the file is totally stored, the total number of blocks allocated will be a multiple of 5 and may not be entirely filled (NOTE: 2 blocks are used for system information, and, if necessary another 5 blocks allocated). The monitor also records the total allocated size of each file.

Each user is assigned 2 quotas on the amount of storage that can be allocated. You are allowed to store a great deal more information when you are logged-in than you may store after you have logged-out, i.e. your logged-in quota is larger than your logged-out quota. This fact can create a problem when you try to sign off. A message similar to the following will sometimes be typed when you attempt to "KJOB":

```
?DSKA Logged out quota 100 exceeded by 5 blocks
```

The important point to note is that your logged-out quota is in terms of the storage allocated, not the number of blocks actually written.

When this message occurs, you must delete enough files so that you are at or below your logged-out quota. This can be done by responding "I" to Confirm:. As the computer types out each entry in your directory, respond with "K" to delete the file, "S" to save it or "H" for help. The files one should normally eliminate are those with the following extensions: .Q?? (an extension beginning with Q indicates an SOS backup file), .BAK (another type of backup file), .REL (compiled programs can be easily recompiled) and .TMP (temporary files).

To check your allocated storage, give the monitor command "DIRECTORY/ALLOC". When your directory is typed out, the number of blocks allocated will be typed in place of the number of blocks used.

To check your quotas, give the monitor command "R QUOLST". The response will consist of 4 numbers: the total number of blocks allocated for your storage, the number of unused blocks in your logged-in quota, the number of unused blocks in your logged-out quota and the number of unused blocks on the system.

If your job remains inactive for more than one hour for any reason, the system may attempt to log your job off. If you are also over your logged-out quota, the system will delete your files one by one until below the quota. The order of deletion is the files with extension .TMP, then .REL, then .BAK, then .SAV. If still more must be deleted, the oldest file (earliest creation date) is deleted first.

Procedure for DECsystem-10 Communications Problems

When problems develop in the Syracuse University Timesharing System, the associations between your terminal and the work you were doing may be broken. Below are listed the four major indicators of such problems. In order to re-establish contact with the system, find the response you are getting (which may be no response), and then follow the appropriate directions. If you have questions, call the AID Office (x3424).

NO RESPONSE---If you get no response to the commands that you type: FIRST of all try typing control-C twice. If you still get no response, press the RETURN key periodically until you get the sign-on message, "Syracuse University Timesharing". Normally, you should get this greeting within 5 minutes of losing terminal response. Call the AID Office (x3424) or the Dispatcher (x3245) if you have questions. When you get the sign-on message, follow the PROCEDURE below. If you get no response for 10 minutes, try again later.

?DECSYSTEM-10 NOT AVAILABLE---If you get this message, try to login in about 5 to 10 minutes, following the PROCEDURE below.

?LOGIN---If you this response to a command when you are already logged in, then follow the PROCEDURE below.

^W'S REPEATEDLY TYPED BACK SURROUNDED BY QUESTION MARKS---If you believe that the commands are valid, type the command DSK. If you get the message "?Login", follow the PROCEDURE below. If you get your disk usage information typed out, sign off (type K/F), then check with the AID Office (Room 116 Machinery Hall) or with your instructor to find out what the problem is. Take the print-out of the problem along; it can help in locating a problem.

?PROCEDURE TO FOLLOW:

(1) Type LOGIN followed by your account number, e.g. LOGIN 14,11.

(2) You may get the following message:

[LGNDJS Detached Jobs same PPN]

If not, continue logging in normally and neglect the rest of this procedure. If you do get the message, you will also be asked if you wish to attach to the detached Job (the one you were using before problems developed). Respond YES and continue to the next step.

(3) You will be asked for your password - type it in as usual.

(4) Type a control-C.

(5) If you had any special terminal characteristics set (for example, SET TTY LC), reset them now.

Type CONTINUE. You should now be back where you were when the ..oblems developed. Note that you may have lost the last line you typed before the problem.

```
TIMESHARING 16.26.50 TUE 06/17/80 V1M51X
/LOG
LOG
JOB 34 SYRACUSE UNIVERSITY 603A TTY6
#454,12
Password:

.DIR/F

CH2RAD.DAT      DSKB:   [454,12]
H2O.DAT
EQL1CN.MCH
EQLOCN.MCH
EQNLCN.MCH
V10LOC.MCH
V10NLC.MCH
V11LOC.MCH
V11NLC.MCH
V12LOC.MCH
V12NLC.MCH
V1LOCN.MCH
V1NLCN.MCH
V2LOCN.MCH
V2NLCN.MCH
V3LOCN.MCH
V3NLCN.MCH
V4LOCN.MCH
V4NLCN.MCH
V5L1CN.MCH
V5NLCN.MCH
V6LOCN.MCH
V6NLCN.MCH
V7LOCN.MCH
V7NLCN.MCH
V8LOCN.MCH
V8NLCN.MCH
V9LOCN.MCH
V9NLCN.MCH
CNDDAT.MCH
XPRDT.DAT
CNDO.FOR
CNDDAT.NFD
```

V1NLCN.NPD
V1LOCN.NPD
XPRDT.FOR
DISPLA.FOR
IRCD.FOR
VOA.TMP
ABSORP.FOR

.QUE CNDO.FOR/NAME:"TOWNSEND"
[LPT:CNDO=/Seq:1249/Limit:165, 1 File]

.QUE DISPLA.FOR/NAME:"TOWNSEND"
[LPT:DISPLA=/Seq:1250/Limit:52, 1 File]

.K/F
Job 34 User PRASAD PL [454,12]
Logged-off TTY6 at 16:29:06 on 17-Jun-80
Runtime: 0:00:00, KCS: 5, Connect time: 0:02:35
Disk Reads: 73, Writes: 3, Blocks saved: 666

Charge: $0.12 = $0.05(R) + $0.01(K) + $0.06(C)
Year-To-Date Account Usage: 63%
.
089) SU TIMESHARING 16.29.38 TUE 06/17/80 V1M51X
/LOG
LOG
JOB 34 SYRACUSE UNIVERSITY 603A TTY6
$454,11
Password:

(16-Jun-80)   A new version of SPELL, the spelling correction program,
        is on NEW: for ^O

.DIR/F

HALOBI.NPT   DSKB: [454,11]
BMAT.FOR
HALCAR.TIN
CART.FOR
DICB.MAT
DICHLE.BMA
DICHLE.CIN
DICHLE.NCA
DICHLE.XMA
DICHLE.ZMA
DICX.MAT
FCLBR.BBB
FCLBR.FFF
FCLBR.XXX
FCLBR.ZZZ
FZMAT.DAT
HCLF.FFF
HCLFB.MAT
HCLFX.MAT
POLDX.MAT
POLDX1.MAT
POLDX2.MAT
TRICHL.BBB
TRICHL.FFF
TRICHL.NCC

```
METCYC.DAT
CYCHEX.BIN
CYCHEX.DAT
ALA.BIN
ALA.UBI
UBZM.FOR
ALA.CIN
ALA.DAT
VOADAT.NH3
TEMP.FOR
CNDDAT.NH3
XXXXX.FOR
VOA.FOR
CNDO.FOR
EQNLCN.NH3
V3NLCN.NH3
V3LOCN.NH3
V4NLCN.NH3
V4LOCN.NH3
V5NLCN.NH3
V5LOCN.NH3
V1NLCN.NH3
V1LOCN.NH3
V2NLCN.NH3
V2LOCN.NH3
V6NLCN.NH3
V6LOCN.NH3
VOADAT.NDT
CNDDAT.NDT
V1NLCN.NDT
V1LOCN.NDT
V2NLCN.NDT
V2LOCN.NDT
CNDO.DAT
EQUIB.DAT
V3NLCN.NDT
V3LOCN.NDT


.QUE BMAT.FOR/NAME:"TOWNSEND"
[LPT:BMAT=/Seq:1251/Limit:64, 1 File]

.QUE CART.FOR/NAME:"TOWNSEND"
[LPT:CART=/Seq:1252/Limit:73, 1 File]

.QUE TEMP.FOR/NAME:"TOWNSEND"
[LPT:TEMP=/Seq:1253/Limit:51, 1 File]

.QUE VOA.FOR/NAME:"TOWNSEND"
[LPT:VOA=/Seq:1254/Limit:97, 1 File]

.K/F
Job 34 User PRASAD PL [454,11]
Logged-off TTY6 at 16:32:23 on 17-Jun-80
Runtime: 0:00:00, KCS: 8, Connect time: 0:03:01
Disk Reads: 119, Writes: 3, Blocks saved: 888

Charge: $0.13 = $0.05(R) + $0.01(K) + $0.07(C)
Year-To-Date Account Usage: 80%
.
089) SU TIMESHARING 16.32.55 TUE 06/17/80 V1M51X
/
```

DARBOUT 13 1001 68
angle 83
bell 66
model 78

46 asin°
50 ABL

```
*P^:*
00100                   OPEN(UNIT=30,FILE='CNDDAT.MCH')
00200                   OPEN(UNIT=31,FILE='GIANT.FIL')
00300                   DO 20 J=1,120
00400                   READ(30,2)X,Y,Z,W,U,V
00500                   WRITE(31,2)X,Y,Z,W,U,V
00600         2         FORMAT(6F12.6)
00700        20         CONTINUE
00800                   CLOSE(UNIT=30,FILE='CNDDAT.MCH')
00900                   CLOSE(UNIT=31,FILE='GIANT.FIL')
01000                   STOP
01100                   END
*EB

[DSKB:READ.FOR]
```

```
00100                   OPEN(UNIT=30,FILE='GIANT.FIL')
00200                   OPEN(UNIT=31,FILE='GIAN.FIL')
00300                   DO 20 J=1,13
00400                   READ(30,1)TITLE
00500                   DO 10 I=1,23
00600                   READ(30,2)X,Y,Z
00700                   WRITE(31,2)X,Y,Z
00800                   READ(30,1)TITLE
00900        10         CONTINUE
01000         2         FORMAT(9X,3F12.6)
01100         1         FORMAT(A4)
01200        20         CONTINUE
01300                   CLOSE(UNIT=30,FILE='GIANT.FIL')
01400                   CLOSE(UNIT=31,FILE='GIAN.FIL')
01500                   STOP
01600                   END
*ES
```