MASTER

樂
語

SPEAKEASY-3

The SPEAKEASY System

S. Cohen

Physics Division

Argonne National Laboratory, Argonne, Illinois 60439

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

The facilities of Argonne National Laboratory are owned by the United States Government. Under the terms of a contract (W-31-109-Eng-38) between the U. S. Atomic Energy Commission, Argonne Universities Association and The University of Chicago, the University employs the staff and operates the Laboratory in accordance with policies and programs formulated, approved and reviewed by the Association.

MEMBERS OF ARGONNE UNIVERSITIES ASSOCIATION

The University of Arizona	Kansas State University	The Ohio State University
Carnegie-Mellon University	The University of Kansas	Ohio University
Case Western Reserve University	Loyola University	The Pennsylvania State University
The University of Chicago	Marquette University	Purdue University
University of Cincinnati	Michigan State University	Saint Louis University
Illinois Institute of Technology	The University of Michigan	Southern Illinois University
University of Illinois	University of Minnesota	The University of Texas at Austin
Indiana University	University of Missouri	Washington University
Iowa State University	Northwestern University	Wayne State University
The University of Iowa	University of Notre Dame	The University of Wisconsin

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately-owned rights.

MASTER 樂語

SPEAKEASY-3

The SPEAKEASY System

S. Cohen*

Physics Division

Argonne National Laboratory

Argonne, Illinois

February 15, 1972

The author has a limited number of copies for general distribution. Anyone who desires a copy should contact him directly; the PHY Informal Reports are not handled through the ANL Technical Publications Department.

NOTICE
This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

*Work performed under the auspices of the U. S. Atomic Energy Commission.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

per

TABLE OF CONTENTS

	<u>Page</u>
I. INTRODUCTION	3
II. NEW OPERATIONAL FEATURES	4
A. SIZE	5
B. VOCABULARY	8
C. HELP	10
D. NAMES	12
E. HENCEFORTH	13
F. OBJECT	15
G. MELD	17
H. CONSTRAIN/SELECT	21
I. ECHO/NOECHO	23
J. LIST/NOLIST	23
K. MARGINS	24
III. LIBRARY FACILITIES	26
A. The LINKULE libraries	31
B. The PROCEDURE libraries	31
C. The HELP library	32
D. The DOCUMENT library	33
E. LIBINDEX	33
F. LIST	34
G. USE	34
H. CREATE	35
IV. LOGICAL AND RELATIONAL OPERATOR NOTATION	35
V. INTERACTIVE SPEAKEASY	38
A. The EDIT Mode	38
B. INPUT/PAUSE/STOP	43
VI. VERSIONS	44
A. STANDARD	44
B. BABY	45
C. GRAPHEZ	45

	Page
D. CONSOLE	45
E. SPEK2250	46
F. TSO	46
VII. ACKNOWLEDGMENTS	47
VIII. REFERENCES	48

ABSTRACT

SPEAKEASY has continued to evolve and grow in power. The current production versions, referred to as SPEAKEASY-3, contain many features that have not been formally documented. This paper is intended to provide the needed documentation for the most important of these features. Individual new words that are now available are explained and illustrated in accompanying sample runs. Because of the growing use of the language in interactive situations, emphasis has been placed on explaining facilities oriented to such usage. The power of a library-directed programming language is discussed, and the concept of the SPEAKEASY system is introduced. This report is a supplement to previously available documentation.

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

I. INTRODUCTION

The capabilities of the current production versions of SPEAKEASY are greatly enhanced over those available at the time of the last documentation.¹ From the viewpoint of the users, the growth has been in the direction of increases in the available facilities and an enlarged vocabulary. Actually there have been several major revisions in the structure of the processor during this period, but none of these have been directly apparent to the user community.

This report is intended to formally describe some of the features now available. In part this means describing the new words in the language. A section of this paper therefore deals with increases in the vocabulary. As will be seen, many of the features of the language now reflect the coordination of broad-based library facilities with the processor. The evolution of SPEAKEASY into a library-directed processor has had profound impact on its growth. Part of this report therefore deals with the library facilities now available and discusses their importance.

No new computational capability is apparent in the increased vocabulary described here; such improvements are now added to the so-called LINKULE libraries. These libraries of compiled FORTRAN subroutines are the means by which all new computational features and most other features are added to the processors. The libraries are discussed only briefly here because a separate report² dealing specifically with their capabilities is available.

Other libraries of the system, such as the documentation libraries and the libraries of stored SPEAKEASY decks, are dealt with in general. Specific information about the contents of these libraries can be obtained by use of the features that are available in the language and are described in this report.

A change in the specifications of the SPEAKEASY language is being introduced in Sec. IV below. This modification of the notation for logical and relational operators eliminates some previously restricted words and therefore will benefit all users. During a long transition period in which either notation can be used, existing programs can be converted to the new specifications. This conversion is not likely to be difficult, however, since the change is being introduced to eliminate conflicts that only a few users have encountered.

The use of SPEAKEASY in interactive sessions is growing in popularity now that TSO (the Time Sharing Option) is available to many institutions. Section V therefore explains the use of the facilities of this language that are particularly adapted to this operation and also describes the EDIT mode of operation.

The various processors currently available at Argonne are described. The choice of the default parameters for each processor is detailed. This serves both as a guide to the users of the particular processors and as an indication of the possible choice for other installations.

II. NEW OPERATIONAL FEATURES

This chapter describes some of the more important words and concepts that have been added to SPEAKEASY since its original documentation. In writing SPEAKEASY documentation, the general approach has always been to describe the particular operation clearly and concisely. The operation is then illustrated by using the actual processor to display the effect. It is felt that this form of documentation lends itself to development of the language and provides users with some direct contact with the applications of the language.

As with all documentations, it is difficult to describe an operation concisely and at the same time indicate in detail the applications to which that operation is particularly suited. This report makes no attempt to do the latter; a separate report illustrating applications of the language and some of the tricks that have been developed is planned.

Each of the sections that follow is headed by a major keyword now available in SPEAKEASY. The operation of that word is described, and the section concludes with a sample of the use of the facility in an illustrative run with one of the currently available processors. In most cases the normal batch version is used because the format clearly labels the input.

Many of the words described in this section are members of the LINKULE library of the system. Some are at present built into the basic processor. Most words now are gradually being shifted from the processor to the libraries. Users of SPEAKEASY should be unaware of such changes except when specific inquiries are made about the contents of the libraries.

A. SIZE

SPEAKEASY has a built-in dynamic-storage allocator that carefully manages the space available for information defined during execution. The SPEAKEASY processors themselves are designed so that parts are dynamically brought into the computer as dictated by demands of the particular application program. Unfortunately, it is not yet possible to automatically divide the space available in the computer between these two uses. While default sizes for each of the above are normally supplied, it is desirable that the space allocated to each be based on the actual need.

The first card in the SPEAKEASY deck can be used to inform the processor of the amount of data space needed in the run. The rest of the allocated space for the job is then available for other uses. This first card will be interpreted as a size-selecting card if it carries any of the following words: SIZE, MAIN, LCS, or AUTOCORE. If none of these occur, the default size selection for that particular processor is assumed and the first card is interpreted as a normal SPEAKEASY statement.

If the size of data storage is being specified, the card should read

SIZE = n

SIZE = n, MAIN

SIZE = n, LCS

or AUTOCORE

Here n is the number of kilobytes of storage to be set aside for SPEAKEASY data (1 kilobyte is approximately 120 user-defined numbers). For most applications the card will read

SIZE = n

For machines with LCS (large-core storage)* the data area may be placed in LCS by including the word LCS in the size selection card. Alternatively, in such machines the selection of LCS space can be automatically determined by the space allocation on the job card by use of the word AUTOCORE. This word means that the largest amount of LCS that is available is to be used for this job. The word MAIN is a default if SIZE is specified and has been included here only for completeness of notation.

Initially users will not know how large a data area to assign to a specific job. The default values should be used in such cases (i.e., the user should omit the size-selection card entirely).

*The Argonne computer has 1 megabyte of LCS available.

At the completion of each job, the last line of output gives three pieces of information: the current amount of space being used (labeled NOW), the peak amount needed (labeled PEAK), and the amount of space allocated (labeled ALLOCATED). These numbers are the approximate numbers of kilobyte involved. The next run of this same job should be given a SIZE specification some 10% larger than the indicated peak value. (The extra space provides for some increase in efficiency of operation.)

Part of some actual SPEAKEASY runs are shown below. These indicate the use of SIZE and shows the form of the final line of output.

```

      SPEAKEASY 3D BETA  7:55 PM  7/20/72
INPUT...SIZE=10
INPUT...MARGINS(1,80)
INPUT...X=MATRIX(5,5:INTEGERS(1,25))
INPUT...Y=X*X
INPUT...PRINT Y

      Y (A 5 BY 5 MATRIX)
      215   230   245   260   275
      490   530   570   610   650
      765   830   895   960  1025
      1040  1130  1220  1310  1400
      1315  1430  1545  1660  1775
* * * * *
CORE USED  1 K NOW,  1 K PEAK,      ALLOCATED  10 K
* * * * *
      SPEAKEASY 3D BETA  8:00 PM  7/20/72
INPUT...SIZE=15
INPUT...MARGINS(1,80)
INPUT...X=GRID(0,10)
INPUT...Y=SIN(X)*COS(X)+X*EXP(-X)+2*X*X
INPUT...AVERAGE(Y)
AVERAGE(Y)  =  67.116
* * * * *
CORE USED  2 K NOW,  5 K PEAK,      ALLOCATED  16 K
* * * * *
```

B. VOCABULARY

The single word VOCABULARY asks the processor to list the currently defined words in its vocabulary. Most of the words are described in the SPEAKEASY manual¹ or are added synonyms for these words. Others are described in this report. The rest have not been documented except internally in the system. A brief description of any of these words can be obtained by use of the special word HELP (described in the next subsection).

Changes in response to the word VOCABULARY can be expected. Eventually it is planned that the combination of this word and the word HELP will be the means by which users are led to all of the available information in the system. One form of the response given on the next page is typical of current processors.

INPUT...VOCABULARY

ABS	CUMPROD	GT	MAT	PRODROWS	SPHBES
ACCURACY	CUMSUM	HELP	MATRIX	PROGRAM	SPHBESN
ACOS	DATA	HENCEFOR	MAX	PUNCH	SQRT
ACOT	DEC	HLABEL	MAXOF	QLEG	SUM
ADDGRAPH	DEGRID	HSCALE	MAXOFCOL	QUIT	SUMCOLS
AFAM	DELETE	HSIZE	MAXOFROW	RANDOM	SUMROWS
AMAT	DERIV	IF	MELD	RANKED	SUMSQ
AND	DERIVATI	IMAG	MFAM	RANKER	SUMSQCOL
ARRAY	DET	IMAGPART	MIN	READ	SUMSQROW
ARRAY2D	DIAGELS	INSERT	MINOF	REAL	SYMMAT
ASIN	DIAGMAT	INTEG	MINOFCOL	REALPART	TABULATE
ASYMMAT	DIFFEQ	INTEGERS	MINOFROW	REAL4	TAN
ATAN	DMAT	INTEGRAL	MOVE	REAL8	TIME
AUTOCORE	DOMAIN	INTERP	MYDOCS	RECLASS	TOTALINT
AUTOPRIN	DUMP	INTERPOL	MYHELP	REPAIR	TOTINT
AUTOTAB	ECHO	INTPART	MYLINKS	RETURN	TRACE
AVERAGE	EDIT	INTS	NAMES	ROOTS	TRANSP
A1D	EIGENSYS	INT2	NE	ROWARRAY	TRANSPOS
A2D	EIGENVAL	INT4	NEUMANN	ROWMAT	UMAT
BESSEL	EIGENVEC	INVERSE	NEWGRAPH	ROWMAX	UNITMAT
BESSELK	END	KEEP	NEWPAGE	ROWMIN	UPPERTRI
CGAMMA	ENDAUTOP	KEPT	NOCOLS	RUN	USE
CLEAR	ENDLOOP	LE	NOECHO	SELECT	USEMEMBE
CLEARDAT	EQ	LENGTH	NOELS	SETGAUSS	VARIABLE
COLARRAY	ERF	LIBINDEX	NOLIST	SETINFIN	VEC
COLMAT	ERFC	LIBNAMES	NOROOT	SETJACOB	VECTOR
COLMAX	ERRORS	LINKLIB	NOROWS	SETLAGUE	VFAM
COLMIN	EXECUTE	LIST	NOT	SETLEGEN	VLABEL
COLWIDTH	EXP	LISTHEAD	NOZEROS	SETLIB	VOCABULA
COMPILE	FIN	LISTMEMB	OBJECT	SETNULL	VSCALE
CONJ	FOR	LOADDATA	OMITCLAS	SETPLOT	VSIZ
CONJUGAT	FRAC	LOC	ONERROR	SIGN	WHERE
CONSTRAI	FRACPART	LOCMAX	OR	SIGNIFIC	WHEREVER
CONTINUE	FREE	LOCMIN	ORDERED	SIMEQ	WHOLE
COPY	GAMMA	LOCS	ORDERER	SIN	WRITE
COS	GE	LOG	PLOTSYMB	SINH	ZEROS
COSH	GEIGEN	LOGGAMMA	PRINT	SIZE	
COT	GOTO	LOWERTRI	PRINTCLA	SMAT	
CREATE	GRAPH	LT	PROD	SNOOP	
CREATEME	GRID	MARGINS	PRODCOLS	SPACE	

C. HELP

The vocabulary of SPEAKEASY is constantly being augmented. While documentation such as this report can be used to communicate new features, it is highly desirable to provide a less formal and more rapidly altered means of describing specific words in the language. In interactive usage, it is also extremely important that both the vocabulary and the documentation for each word be easily accessible.

The basic vocabulary of the processor is printed out by the command VOCABULARY described previously. Each word in this vocabulary and other words in the language are described in a concise operational form and are available on demand by use of the key word HELP. The input statement

```
HELP xxxx
```

requests that the processor print out a brief description of the word xxxx. The documents available in this manner are not restricted to those listed in the vocabulary but are actually a separate document library that is appended to the various processors.³ By using a standard sequence of SPEAKEASY statements, one can obtain a catalog of this library. That is, the names of the documents available by use of the word HELP will be printed in response to the input

```
CATALOG = LIBINDEX (HELP)
TABULATE (CATALOG)
```

The entire set of documents may be printed out by use of the following SPEAKEASY program:

```
PROGRAM LISTHELP
CATALOG = LIBINDEX (HELP)
FOR I = 1, NOELS (CATALOG)
SPACE(2)
```

```

HELP OBJECT (CATALOG(I))
ENDLOOP I
END
EXECUTE LISTHELP

```

A series of statements similar to the above is itself stored in the SPEAKEASY procedure library and can be invoked by the statement

```
USE MEMBER LISTHELP
```

This single statement will result in the complete listing of the available HELP documents.

Information necessary to make use of the HELP processor itself is available by the statement

```
HELP HELP
```

A few of the actual HELP documents are shown below.

INPUT...HELP MATRIX

MATRIX(N,M:) defines an N by M matrix.

If no additional arguments are present the matrix has all elements set to zero.

A shortened form is MAT.

MATRIX(N,M:I,J,...) defines an N by M matrix with preset elements. The elements are set row by row using the values I,J,... or the elements of I,J... if they are structured objects. If a complex element is encountered, then a complex matrix is defined. If all the elements are not specified by the element list, the unspecified elements are set to zero.

INPUT HELP HELP

HELP contains a brief description of the SPEAKEASY words. An alphabetical listing of the words in the HELP library may be obtained by the statements

```
CATALOG=LIBINDEX(HELP)
TABULATE(CATALOG)
```

To obtain the description of a single SPEAKEASY word, XX, the input statement is

```
HELP XX
```

All the words with a description of each may be obtained by the single statement

```
USE MEMBER LISTHELP
```

A one line description of a SPEAKEASY word may be obtained by the statement

```
LIST MEMBER XX OF LIBRARY HELP TO 1.
```

D. NAMES

The single word NAMES requests the processor to print out the names of all of the currently defined SPEAKEASY objects. This is extremely useful in the interactive mode since it provides the user with the means of recollecting the names of variables that he has previously defined. Combined with the implied print statement, the user can quickly examine any currently defined object.

A section of programming in which objects are defined and freed is shown below. Between each such command the processor was requested to display the defined names.

```

INPUT...Z=5; X=GRID(0,10,2)
INPUT...W=30.4
INPUT...Y=Z*W
INPUT...NAMES
      CURRENTLY DEFINED NAMES
      Y , Z , X , W
INPUT...WORDS=LIBINDEX(HELP)
INPUT...N=NOELS(WORDS)
* * * * *
* PROCEDURE LISTHELP
* TABULATE WORDS
* FOR I=1,N
* USE MEMBER OBJECT(WORDS(I)) OF LIBRARY HELP TO 1
* ENDOLOOP I
* END
* * * * *
INPUT...NAMES
      CURRENTLY DEFINED NAMES
      Y , Z , X , W , WORDS , N , LISTHELP (A PROCEDURE)
INPUT...FREE LISTHELP
INPUT...NAMES
      CURRENTLY DEFINED NAMES
      Y , Z , X , W , WORDS , N
INPUT...FREE WORDS, N
INPUT...NAMES
      CURRENTLY DEFINED NAMES
      Y , Z , X , W
INPUT...CLEAR
INPUT...NAMES
      CURRENTLY DEFINED NAMES

```

E. HENCEFORTH

Although every effort has been made to choose the names of SPEAKEASY functions with care, it is often desirable to provide alternatives. Previously this was done by adding synonyms (for example MAT is equivalent to MATRIX, VEC is equivalent to VECTOR, etc.). A more flexible facility has now been provided to enable users to define their own synonyms at execution time. The statement

HENCEFORTH X IS Y

means that:

Anytime hereafter, if the word X is encountered treat that word as if it were the word Y (the word Y is still usable).

The statement

HENCEFORTH X IS X

stops the redefinition of X for future use.

There are only a very few words in SPEAKEASY that may not be given synonyms by this method. They are the words

EXECUTE, PROGRAM, PROCEDURE, DATA,
END, FOR, ENDLOOP, QUIT, LOADDATA,
GOTO

Note that HENCEFORTH itself can be given a synonym by this method. In addition, the word IS in the expression is an arbitrary word chosen to make an easily remembered sentence, any word can be used. For example, the sequence

HENCEFORTH TREAT IS HENCEFORTH
TREAT S AS SIN
TREAT C AS COS
TREAT LET AS HENCEFORTH
LET M BE MATRIX

is acceptable. The statement

HENCEFORTH X IS 'SIN(X)'

is usable with the restriction that the literal quantity can have at most eight letters. Any of the normal SPEAKEASY character set except an apostrophe is allowed. This makes it possible to obtain output that is particularly neat.

HENCEFORTH is a convenient way of compacting a user's program by eliminating repetitious typing of long words. It is particularly important when combined with the operation OBJECT described later in this document.

The use of HENCEFORTH is illustrated below.

```

INPUT...HENCEFORTH M IS MATRIX
INPUT...Y=M(2,2:1,5,8 4)
INPUT...PRINT Y
  Y (A 2 BY 2 MATRIX)
    1  5
    8  4
INPUT...HENCEFORTH TREAT IS HENCEFORTH
INPUT...TREAT P AS PRINT
INPUT...TREAT LET AS TREAT
INPUT...LET V BE VECTOR
INPUT...W=V(4:1,2,3,4)
INPUT...P W
  W (A VECTOR WITH 4 COMPONENTS)
    1  2  3  4

```

F. OBJECT

It is often desirable to be able to generate the names of objects. This is particularly true when defining large numbers of objects, each of which is dependent on particular parameters of a calculation. A general operation for generating names has made available.

It should be remembered that names in SPEAKEASY can have at most 8 characters. This restriction holds for names that are generated by the mechanism described here. Other restrictions such as the use of special characters do not exist since the names are generated internally in the processor.

The expression

OBJECT(I, J, K, . . .)

may occur anywhere in a SPEAKEASY statement. It means that the arguments I, J, K, etc. are to be used to generate a name that is to be substituted for this expression.

Each of the arguments can be either a literal quantity or a non-negative number. For literals, the expression itself is used. For numbers, the integer part of the number is used as a literal. The various arguments are then joined together to make up the name. Thus

OBJECT ('A', 3, 'B') becomes A3B

OBJECT ('A', 3, 'X', 4) is A3X4

If X = 'XX'; Y = 'Y'; I = 4 then

OBJECT (X, I, 3 Y) becomes XX43Y

The use of literal quantities in expressions means that strange looking names can be generated. Thus if X = '*4' then

OBJECT (X, X) becomes *4*4

The power provided by the use of the word OBJECT is greatly

enhanced when combined with the word HENCEFORTH. The expression

HENCEFORTH X IS OBJECT ('X', I)

means that the variables X1, X2, X3, etc. are specified when the current value of the variable I is 1, 2, 3, etc. The following series of statements illustrates one of the uses of the feature to obtain a rather elegant printout which would not otherwise be possible.

```
INPUT...PI=2 ACOS(0);X=GRID(0,2 PI,PI/40)
INPUT...N=3;HENCEFORTH Y IS OBJECT('SIN(',N,'X')')
INPUT...Y=SIN(N X)
INPUT...TABULATE(X,Y)
```

X	SIN(3X)	X	SIN(3X)	X	SIN(3X)
0	0	2.1206	.078459	4.2412	.15643
.07854	.23345	2.1991	.30902	4.3197	.38268
.15708	.45399	2.2777	.5225	4.3982	.58779
.23562	.64945	2.3562	.70711	4.4768	.76041
.31416	.80902	2.4347	.85264	4.5553	.89101
.3927	.92388	2.5133	.95106	4.6338	.97237
.47124	.98769	2.5918	.99692	4.7124	1
.54978	.99692	2.6704	.98769	4.7909	.97237
.62832	.95106	2.7489	.92388	4.8695	.89101
.70686	.85264	2.8274	.80902	4.948	.76041
.7854	.70711	2.906	.64945	5.0265	.58779
.86394	.5225	2.9845	.45399	5.1051	.38268
.94248	.30902	3.0631	.23345	5.1836	.15643
1.021	.078459	3.1416	0	5.2622	-.078459
1.0996	-.15643	3.2201	-.23345	5.3407	-.30902
1.1781	-.38268	3.2987	-.45399	5.4192	-.5225
1.2566	-.58779	3.3772	-.64945	5.4978	-.70711
1.3352	-.76041	3.4558	-.80902	5.5763	-.85264
1.4137	-.89101	3.5343	-.92388	5.6549	-.95106
1.4923	-.97237	3.6128	-.98769	5.7334	-.99692
1.5708	-1	3.6914	-.99692	5.8119	-.98769
1.6493	-.97237	3.7699	-.95106	5.8905	-.92388
1.7279	-.89101	3.8485	-.85264	5.969	-.80902
1.8064	-.76041	3.927	-.70711	6.0476	-.64945
1.885	-.58779	4.0055	-.5225	6.1261	-.45399
1.9635	-.38268	4.0841	-.30902	6.2046	-.23345
2.042	-.15643	4.1626	-.078459	6.2832	0

G. MELD

Some SPEAKEASY programs include a sequence such as:

```

FOR I = 1, NOELS(X)
FOR J = 1, NOELS(Y)
F(I, J) = some function of X(I) and Y(J)
ENDLOOP J
ENDLOOP I

```

Such looping, though logically correct, defeats many of the optimizing features of the language. Users have been warned of the consequence of doing element by element operations that do not make use of the built-in algebra for structured objects. For a pair of nested loops as illustrated, the array algebra of the language is usually sufficient to enable one to eliminate the loops easily.

The problem of multidimensional arrays with more than two independent variables is not easily dealt with. For this purpose, a new concept and a new word has been added to the SPEAKEASY language. Both the word and the concept are due to Richard Kimmel. The word MELD provides a major advance in the capabilities of the language.

Although MELD is a straightforward operator, it differs from others that have previously been met in SPEAKEASY since it redefines the arguments occurring in the statement. Thus

```
MELD(I, J, K)
```

in fact alters the structure of I, J, and K.

The arguments in the MELD must be 1-dimensional objects. In essence, the MELD operation is a simple and direct means of providing a revised set of objects in which every element of each of the objects in the argument list is associated with every element of every other object in that list. In fact, as will be seen,

this is a simple way of describing a multidimensional space. A few examples are sufficient to show how the operator acts. (It is much easier to show the operation than to describe it in words.) If I is a 2-component 1-dimensional array and J is a 3-component array, e.g., if

$$I = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad J = \begin{pmatrix} 0 \\ 5 \\ 7 \end{pmatrix},$$

then MELD (I, J) alters both I and J and produces two 6-component arrays

$$I = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \end{pmatrix}, \quad J = \begin{pmatrix} 0 \\ 5 \\ 7 \\ 0 \\ 5 \\ 7 \end{pmatrix}.$$

Similarly if I, J, and K are

$$I = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad J = \begin{pmatrix} 1 \\ 2 \\ 4 \end{pmatrix}, \quad K = \begin{pmatrix} 8 \\ 9 \end{pmatrix},$$

then MELD (I, J, K) redefines them all to be the 12-component arrays

$$I = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}, \quad J = \begin{pmatrix} 1 \\ 1 \\ 2 \\ 2 \\ 4 \\ 4 \\ 1 \\ 1 \\ 2 \\ 2 \\ 4 \\ 4 \end{pmatrix}, \quad K = \begin{pmatrix} 8 \\ 9 \\ 8 \\ 8 \\ 9 \\ 8 \\ 9 \\ 8 \\ 9 \\ 8 \\ 8 \\ 9 \end{pmatrix}.$$

Note that after melding, as in the last example, any function of I, J, and K can be written in a straightforward manner. Thus

$$F = 3 * I + 2 * (I + J) - K * I * \text{SIN} (PI * K)$$

is allowed. The function is then effectively evaluated in the 3-dimensional space spanned by the three original arrays. A table of the values of I, J, K, and the resultant could be obtained by

TABULATE (I, J, K, F)

Up to ten arguments are permitted in a single MELD call. Care should be taken not to produce unreasonably large arrays by this method. For example, if ten arrays of only 3 components each are melded, then each of the resultant arrays would have 3^{10} elements. This would, in fact, far exceed the capacity of any available SPEAKEASY processor.

Melding combined with the use of structured indices leads to other capabilities that are often needed. If one melds the indices of an array rather than its elements, then it is possible to carry out later operations on this and correlated arrays.

The use of the MELD operation is shown below. The evaluation of the function W as a function of the three independent variables I, J and K is carried out by means of melding.

INPUT...I=ARRAY(5:1 2 3 4,5);J=ARRAY(3:7,8,9);K=ARRAY(2:0,6)

INPUT...TABULATE(I,J,K)

I	J	K
1	7	0
2	8	6
3	9	
4		
5		

INPUT...MELD(I,J,K)

INPUT...TABULATE(I,J,K)

I	J	K	I	J	K
1	7	0	3	8	6
1	7	6	3	9	0
1	8	0	3	9	6
1	8	6	4	7	0
1	9	0	4	7	6
1	9	6	4	8	0
2	7	0	4	8	6
2	7	6	4	9	0
2	8	0	4	9	6
2	8	6	5	7	0
2	9	0	5	7	6
2	9	6	5	8	0
3	7	0	5	8	6
3	7	6	5	9	0
3	8	0	5	9	6

INPUT...W=3*I+4/J-K

INPUT...TABULATE(I,J,K,W)

I	J	K	W	I	J	K	W
1	7	0	3.5714	3	8	6	3.5
1	7	6	-2.4286	3	9	0	9.4444
1	8	0	3.5	3	9	6	3.4444
1	8	6	-2.5	4	7	0	12.571
1	9	0	3.4444	4	7	6	6.5714
1	9	6	-2.5556	4	8	0	12.5
2	7	0	6.5714	4	8	6	6.5
2	7	6	.57143	4	9	0	12.444
2	8	0	6.5	4	9	6	6.4444
2	8	6	.5	5	7	0	15.571
2	9	0	6.4444	5	7	6	9.5714
2	9	6	.44444	5	8	0	15.5
3	7	0	9.5714	5	8	6	9.5
3	7	6	3.5714	5	9	0	15.444
3	8	0	9.5	5	9	6	9.4444

H. CONSTRAIN/SELECT

As has been mentioned, MELD produces correlated elements in several objects. It is often then necessary to make selections based on certain constraints between the elements of these objects. An operation is provided to carry out such selections for all of the objects and other correlated objects simultaneously. For example, the command

CONSTRAIN (A, B, C, D : (A + B.GT.C) .AND. B.NE.C)

means that the logical expression to the right of the colon is to be constructed. The objects to the left of the colon are truncated to leave only elements corresponding to true values in the logical expression. For example, if X, Y, and Z describe positions in 3-space, the statement

CONSTRAIN (X, Y, Z:X**2+Y**2+Z**2.LT.R**2)

would eliminate any point described by X_i , Y_i , and Z_i lying outside the sphere of radius R. Note that all arguments must be 1-dimensional or scalar, and all 1-dimensional objects must have the same length.

SELECT is similar to CONSTRAIN. The effect of the operator is to truncate (or expand) several correlated 1-dimensional objects by use of a single structured index as the control array.

Thus if

$$I = \begin{pmatrix} 1 \\ 3 \\ 5 \\ 7 \end{pmatrix}$$

and A, B, and C are all 1-dimensional arrays, each with at least seven components, then

SELECT (A, B, C : I)

redefines A, B, C to be 4-component objects with the 1st, 3rd, 5th, and 7th elements of the original objects. Note that in this operation A, B, and C need not be the same length.

Samples of the operations CONSTRAIN and SELECT are shown below.

```
INPUT...I=INTEGERS(3,10)
INPUT...J=INTEGERS(20,13)
INPUT...L= 1 3 4
INPUT...TABULATE I J
```

I	J
3	20
4	19
5	18
6	17
7	16
8	15
9	14
10	13

```
INPUT...SELECT(I,J:L)
INPUT...TABULATE I J
```

I	J
3	20
5	18
6	17

```
INPUT...X=GRID(0,1,.1);Y=X;MELD(Y,X)
INPUT...CONSTRAIN(X Y:X**2+2*Y**2 LT .5)
INPUT...TABULATE X Y
```

X	Y	X	Y	X	Y
0	0	.4	.1	.2	.3
.1	0	.5	.1	.3	.3
.2	0	.6	.1	.4	.3
.3	0	0	.2	.5	.3
.4	0	.1	.2	0	.4
.5	0	.2	.2	.1	.4
.6	0	.3	.2	.2	.4
.7	0	.4	.2	.3	.4
0	.1	.5	.2	.4	.4
.1	.1	.6	.2	0	.5
.2	.1	0	.3		
.3	.1	.1	.3		

I. ECHO/NOECHO

One feature of SPEAKEASY that has proved desirable in batch operation is that of printing the actual input information along with the results. This "echoing" of the input means that the user can clearly see what it was that he asked and can see the response immediately below it.

In other applications, particularly in interactive usage, repeating the input information would be redundant since it would follow immediately below the typed line. The ability to control the echoing therefore has been added to the language. If the statement NOECHO is encountered during execution, the echoing of input data is suppressed. ECHO reinstates it.

One application of this feature in normal batch jobs is to produce more publishable results. Extraneous commands to the processor can be done in the NOECHO mode without their operations appearing on the output.

J. LIST/NOLIST

The listing of SPEAKEASY programs can be controlled in much the same way as that of the ECHO facility in the MANUAL MODE. NOLIST indicates that such programs should not be printed in the output. LIST means that they should. If the options NOLIST and NOECHO are selected, only the actual results produced during execution will be printed—as in the operation of conventional programs such as FORTRAN.

In the interactive operation, NOLIST and NOECHO were selected as the preferred mode of operation. Batch processing, on the other hand, normally uses LIST and ECHO.

K. MARGINS

This is a facility that enables a user to control the width and position of his printed output. This is of particular importance when devices other than line printers are used for output. For instance a teletype has only 72 characters on a line and would not supply the output in the proper format for a printer.

The user specifies the left-hand and right-hand limits of the printout by executing a statement of the form

MARGINS (n, m) where $n < m$

which means that output should be restricted to columns starting at n and ending with m. Alternatively

MARGINS (m) is equivalent to MARGINS (1, m)

All printed output normally has a so-called carriage-control character in its first position. This is not printed but is used to control vertical spacing and to position output at the top of a new page. In some applications it is necessary to suppress such control functions. The statement

MARGINS (0, m) is equivalent to MARGINS (1, m)

except that the carriage-control characters are eliminated.

The SPEAKEASY processor will readjust all of its printout to conform to the specified margins.

The next page shows how the word MARGINS can be used to control the format of output.

```
INPUT...A=AMAT(3:1,2,3)
INPUT...PRINT A
```

```
A (A 3 BY 3 MATRIX)
0 -1 -2
1  0 -3
2  3  0
```

```
INPUT...S=SMAT(3:1,2,3,4,5,6)
INPUT...MARGINS(20,40)
INPUT...PRINT S
```

```
S (A 3 BY 3 MATRIX)
1  2  4
2  3  5
4  5  6
```

```
INPUT...MARGINS(30,120)
INPUT...AS=A*S
INPUT...PRINT ('THE PRODUCT IS 'AS)
THE PRODUCT IS
```

```
AS (A 3 BY 3 MATRIX)
-10 -13 -17
-11 -13 -14
8    13  23
```


III. LIBRARY FACILITIES

The SPEAKEASY-3 processors were developed with the intent of making extensive use of libraries that could be attached to the processors at execution time. Once the means of accessing such libraries was clearly defined, the processors and the libraries could be developed independently rather than in concert.

The consequences of the separation cannot be over-emphasized. The processor developments have by and large been in the direction of providing very basic facilities; their objective is always a smaller executable module with as few specialized features as possible. This is in marked contrast to the developments in the libraries, in which as many new features as possible are sought. One wants the greatest possible capabilities in general, but for each specific application one wants to avoid the difficulties associated with such generality. Libraries answer both needs admirably.

Detached libraries have many additional direct advantages. Users can freely put information into private libraries without fear of altering the operations of the processors. Similarly, new words for the language can be tested and validated by use of standard processors without the usual problems associated with new releases. The transfer of information from private to communal libraries is not a major disturbance to the system. The growth of the over-all language since the introduction of attachable libraries has been dramatic. (Most of the words described in this report are in the libraries of the system—some were in fact added to fill gaps in this writeup itself.)

The growth capabilities of the SPEAKEASY system now rests in the libraries attached to the processors. To a large extent, the processor can now be viewed as an interface between the various libraries as well as between the users and the libraries.

This section of the report describes the various libraries in the system. To some extent these descriptions tend to overemphasize their distinctions. Information (i. e., computational techniques or data) can be entered into the system in a variety of ways. The choice of a particular library or combination of libraries for the storing of information is somewhat arbitrary and may reflect personal whims. The system is sufficiently broad-based to accept several alternatives. There should therefore be no need to be concerned about following rigid rules in selecting the mode of operation.

Actually there is a growing interconnection between the libraries that involve the SPEAKEASY processor only as a communication module. The exciting aspects of the system at this stage in the growth is in this interconnectivity. Each new feature added to the system enhances the over-all capabilities of the system, not only because of direct contribution but even more by the ways in which it can be interconnected with other facilities already in the system. The potential power of the SPEAKEASY system rests in this limitless growth capability.

Each user community exposed to SPEAKEASY can develop its own library. It can at the same time draw on the other libraries easily. The major problem being faced is not the construction nor the operations of the libraries themselves but rather in the communication between users and between user communities.

This report itself represents a crude method of communicating facilities, some of which have been available within the system for a long time. Other major facilities will no doubt become available before this report is actually distributed. It is for this reason that the current effort in development is directed towards techniques for information retrieval within the system, with emphasis on information available about the system itself. Only a few of the

tools are available at present. They are in the process of being expanded to provide users with the necessary information.

The types of libraries can be divided into two sets, those that could in principle be read directly by people and those that represent stored information that is usable only by the computer. In the former class one can put the various forms of documentation that are associated with this system or that are stored within the system for other reasons. In addition, stored decks of SPEAKEASY statements represent information that is readable by people although it is also intended to be used directly by the computer.

The latter type of library (i.e., computer-readable information) for the most part represents stored compiled computer programs. These are specially designed routines that are compiled independently from the processors and are placed in accessible libraries. These libraries, in which the specialized tools intended to carry out major functions are stored, are the so-called LINKULE libraries.

LINKULES represent the real operational capability of the over-all SPEAKEASY system. They usually are efficient routines for carrying out specific mathematical operations. They are of use, however, only if there are documents that can explain how the operation is used and what it does. For each member in the communal libraries of the system, there must exist documentation. Such documentation is in one of the other libraries of the system and is thus readily available to users of the system.

It is obviously necessary to provide some method for finding the contents of specific libraries. The ability to ask the system for such information is essential so that exploratory searches can be made. Processors to enable the user to carry out such searches have been built into the LINKULE library. These have already been alluded to in the discussion of HELP.

Finally it is desirable that a user needing specific information should be able to carry out systematic searches through all the libraries of the system. This can in fact be done by connecting the facilities of the system described above together by a SPEAKEASY program such as the one described in the HELP section. Thus one sees that the documentation libraries, the LINKULE libraries, and the libraries of stored SPEAKEASY statements can be used together even at this very simple level.

Probably a single LINKULE could be written to carry out all of these functions. However, it would not have the flexibility inherent in the highly modular interconnected approach outlined above.

There are still problems of communication. A user of the language processor still can be unaware of how to get to a specific piece of information, even though that information exists within the system. To attempt to inventory all of the information available in the system, even as it exists today, would be a formidable task. It would surely swamp the user with undesired information. Lest there be a misunderstanding, it is not felt that this is an unsolvable problem. It is just that simple techniques of indexing and of report generation are not satisfactory solutions, and more powerful facilities will have to be added. A major effort in this direction has now been started.

The sections that follow describe each of the types of libraries currently considered part of the system. Others will be added as the need arises. Each section is a rather general over-all description of the purpose of the library. More specific information about the individual members of the libraries are to be found within the system itself. A set of operators for this purpose is described at the end of this section.

All libraries in the system are, in IBM terminology, partitioned data sets (PDS). Each such data set is a collection of members referred to by distinct member names. Although the data

sets themselves are named, in this application we are interested only in the name used to refer to the data set in the SPEAKEASY run. This name by which we refer to the library is called the DDNAME in standard IBM Job-Control Language^{*} and the FILE in TSO usage.[†]

The SPEAKEASY processor assumes that certain libraries are attached and checks to see if other special libraries are available for this run. The specific library names and their contents are as follows.

1) Libraries that are assumed to be attached are:

LINKULES	libraries of operations (compiled FORTRAN routines)
PROCLIB	library of stored SPEAKEASY statements
HELP	library of brief HELP documents
DOCUMENT	library of larger documents

2) Optionally attached libraries:

MYLINKS	private versions of LINKULES
MYPROCS	private library of stored SPEAKEASY statements
MYHELP	private HELP words
MYDOCS	private documents

Any additional libraries can be attached to the system to supplement those listed above. However, it is necessary to

^{*}For standard batch jobs, the cards which have the form

//name DD . . .

are called DD cards. The first field (i. e., the word name above) is the DDNAME referred to and is called the library name in this report.

[†]In TSO runs, the statement

ALLOCATE FILE(name) DA(. . .)

the equivalent way to define the library name.

explicitly communicate the names of such libraries to the processor. For example, adding additional LINKULES can be done by an explicit statement of the form

LINKLIB = 'XXX'

where XXX is an additional library of LINKULES. Similarly, document libraries are addressed by indicating the library name in the reference statement.

A. The LINKULE libraries

These libraries contain packages of compiled FORTRAN subroutines. Each of the members of such a library is available to the processors. If in the execution of any SPEAKEASY statement a word is encountered that has not been previously defined, then the system library is searched for a member with that name. If one is found, then that routine is brought into the computer and control is transferred to it. The calling sequence for these subroutines is designed to enable complete information transfer between processor and the individual LINKULES. The form of this calling sequence and the method of communication to the processor is described in a separate report.²

The user library MYLINKS is of the same form as the system library but represents personal routines. If MYLINKS is an attached library, then it will be searched for a given member prior to the search of the LINKULES library.

B. The PROCEDURE libraries

Instead of distributing listings and copies of commonly used SPEAKEASY decks, a library named PROCLIB has been created

for them. This library is always attached to the SPEAKEASY processors. The procedures in this library are directly available as input to the processor and can be used in any program by inserting a card of the form

USE MEMBER NAME

where NAME is the name of the particular procedure desired.

A user may append his own library of such statements to the system by assigning it the name MYPROCS. His library is then available for use within the run and is considered as part of PROCLIB for that run.

Since members of the procedure library are to be considered part of the generally available resources in SPEAKEASY, a description of the use of each procedure is also included in the HELP library.

C. The HELP library

The members of the HELP library³ are concise documents describing words and features available to the SPEAKEASY processor. These documents are oriented towards the interactive user who is interested only in making use of the facility and not in a detailed description of its internal workings. The intent is to enable the user to quickly find out about a feature so that he may use it in the calculation currently before him.

It is intended that a HELP document will exist for every word used in the processor and for every member attached to it in a system library.

If the user attaches his own library of such brief documents, he should give it the name MYHELP. In this case, all of his documents are also available during that run.

Any specific HELP document can be obtained by an input statement

HELP XXX

where XXX is the name of the desired document.

D. The DOCUMENT library

In many cases the brief HELP documents described above are too concise to explain details about particular words. A larger document library is available for more lengthy descriptions. A member of this library is obtained by use of the statement

DOCUMENT XXX

where XXX is the name of the desired document. The user may attach his own library of documents. It should be given the library name MYDOCS. The library name for the system documents is DOCUMENT.

E. LIBINDEX

The statement

LIBINDEX (name)

defines a literal 1-dimensional array with the members of the named library as components of the array. This array can be used in many ways, the simplest one being to tabulate it. The statements

LINKS = LIBINDEX (LINKULES)

TABULATE (LINKS)

will produce a table containing the names of all the LINKULES in the system library. Similarly

HELPNMS = LIBINDEX (HELP)
 TABULATE (HELPNMS)

will list the names of all of the available HELP documents.

F. LIST

The statement available to list members of a library has a generalized keyword format. Each keyword encountered is a signal that the next information is to be associated with the keyed option. The keywords, their default values, and their functions are

<u>KEYWORD</u>	<u>DEFAULT</u>	Meaning
MEMBER	INDEX	Select member to be listed
LIBRARY	PROCLIB	Select library to search for member
FROM	1	Start listing from line #
TO	90000	End listing at line #

Any words that are not keywords are ignored. Thus

LIST MEMBER MOON FROM 3 TO 7

will produce a printed copy of lines 3 through 7 of member MOON of PROCLIB.

G. USE

Since decks of SPEAKEASY statements can be stored in libraries, a method must be provided to make these decks available to the processor. Execution of the statement

USE MEMBER memname OF LIBRARY libname

causes the deck of the designated name from the library to be used

as input to the processor. If the library reference is omitted, PROCLIB is assumed. If a library named MYPROCS is attached, it will be searched before looking in PROCLIB for this particular member. When the member has been completely read in, the input will be taken from the normal input device.

Such items as SPEAKEASY programs, series of HENCEFORTH statements, and notes to be printed to the user can all be in such libraries. The use of this facility in private libraries is to supply commonly used constants and SPEAKEASY programs to the processor in a simple way.

H. CREATE

It is possible to create new members of documentation and procedure libraries while running in SPEAKEASY. This is done by the simple command

```
CREATE MEMBER memname OF LIBRARY libname
```

This command indicates that the lines that follow are to be used to define a new library member. The processor itself is passive in this operation. The creation of the new member is terminated by a single word

```
ENDCREATE
```

in the input data. All information between those two statements is saved as the newly created member.

IV. LOGICAL AND RELATIONAL OPERATOR NOTATION

In the original specifications for SPEAKEASY, a special set of restricted keywords were used as logical and relational

operators. This has on occasion caused difficulties because these words cannot be used as names of objects. For this reason, the language specifications have been changed. The introduction of SPEAKEASY 3E (October 1972) will begin a transition period to a logical operator notation similar to the FORTRAN conventions. During the transition period, both the old and new notations will be accepted.

In order to benefit from the new conventions, however, it is necessary to provide users with a means of eliminating the older restricted words. During the period of transition, therefore, a SPEAKEASY statement of the form

.NEW.

will deactivate the restricted words for logical operators. The statement

.OLD.

will reactivate them if it is necessary. The default condition will be .OLD. —at least during the early part of the transition period. The transition period will be a long one—users writing new programs should make use of the new notation and should gradually replace statements using the older words. The logical and relational operators in SPEAKEASY are listed in the following table.

Older form (being phased out)	Newer form (now acceptable)	Meaning
LT	.LT.	Less than
LE	.LE.	Less than or equal to
NE	.NE.	Not equal to
EQ	.EQ.	Equal to
GE	.GE.	Greater than or equal to
GT	.GT.	Greater than
AND	.AND.	And
OR	.OR.	Or
NOT	.NOT.	Not

V. INTERACTIVE SPEAKEASY

An interactive version has been available since the inception of SPEAKEASY. The 2250 version of the language has been used for a variety of calculations. With the introduction of TSO, an interactive version of SPEAKEASY is now becoming available for use by a large community. TSO SPEAKEASY differs from conventional SPEAKEASY only in the interactive capabilities. Users in the MANUAL MODE of operation are able to direct the processor step by step through a calculation. They may examine the information, make corrections to it, and thus proceed directly through the steps to the completed results. In this mode of operation, the system can be viewed as a super desk calculator. Operations on whole arrays of elements can be carried out with a single command. All of the large sets of capabilities of SPEAKEASY are literally at the users' fingertips.

The program mode of operation is similar to the normal batch operation except that the results are instantaneously available. Small SPEAKEASY procedures that will be repeated several times can be programmed during the session at the terminal and used immediately.

Errors occurring in the manual mode are repaired by merely retyping the correct input. In the program mode, it is necessary to edit previously entered information. The EDIT mode is available for this purpose.

A. The EDIT Mode

In batch processing, there is no need to edit the statements of a SPEAKEASY program since they can be altered only after the job has been completed. In interactive usage, on the

other hand, such editing is of great importance. Facilities for this purpose are provided in SPEAKEASY. All processors include these features, but they are not normally used except in an interactive environment.

The EDIT mode is entered automatically when the word PROGRAM, PROCEDURE, or DATA is encountered in the input stream. In the EDIT mode, statements are assigned successive integer reference numbers starting with the number 1. The program mode is left if the single word END is encountered. At this time, the program is compacted and the statements are individually examined for syntax errors. The stored program is then defined as a single object whose name is the name of the program.

While the processor is in the EDIT mode, certain control functions are activated. All such functions are selected by a % in the first field in the input card. The functions are

%LIST	List the entire edit file
%LIST N	List the statement with line number N
%LIST N, M	List statements with line numbers between N and M
%N Statement	Assign this statement the number N
%INSERT N	Insert the statements that follow at N, N + 1, . . .
%INSERT N(i)	Insert the statements that follow at N, N + i, N + 2i, . . .
%	Stop the insert
%DELETE N	Delete statement number N
%DELETE N, M	Delete statements N through M
%MOVE N	Move statement N to the last position
%MOVE N, M	Move statement N through M to the last positions

%MOVE N, M, K	Move statements N through M to K, K + 1, K + 2, . . .
%MOVE N, M, K(i)	Move statements N through M to K, K + i, K + 2i, . . .
%COPY N	Copy statement N into the last position
%COPY N, M	Copy statements N through M into the last positions
%COPY N, M, K	Copy statements N through M into K, K + 1, K + 2, . . .
%COPY N, M, K(i)	Copy statements N through M into K, K + i, K + 2i, . . .

Although integer values are automatically assigned to statement numbers, these numbers can have smaller incremental values. During normal editing, numbers with increments as small as 0.01 are allowed. Thus a statement of the form % COPY 5, 10, 18.9 (.01) is acceptable.

If the process of editing generates a statement number that is identical to a previous one, then the old one is replaced. Care should be taken to protect previous information when performing multiple insertions.

For interactive processing, a second copy of the program is maintained. This copy contains statements in their original form and with the associated statement numbers. In such cases, the processor can be returned to the EDIT mode and a previously defined program can be activated by the statement

EDIT xxxx

where xxxx is the name of the program to be edited. This program can then be corrected by replacing, deleting, or inserting statements.

After it is satisfactorily corrected, the single word COMPILE will return SPEAKEASY to the manual mode. The word RUN is equivalent to the word COMPILE followed by the EXECUTE command.

In the interactive operation of SPEAKEASY, the user is informed of changes in the mode of operation as they occur. Once again, this information is suppressed for normal batch operations. In contrast, the program listing is normally printed in batch operation and is available only on command in interactive operation.

The following sample of the operation of the EDIT mode is a run carried out with the interactive version of SPEAKEASY operating under TSO.


```

:_program fit
  EDIT MODE
:_a=array(noels(x),nfit:); i=integers(1,nfit)
:_a=mfam((x+a)**(i-1)); y=vfam(y)
:_afit=1/(transpose(a)*a)*transpose(a)*y
:_print('    the best fit is ',afit)
:_end
  MANUAL MODE
:_x= 1 2 3 4
:_y=3*x**2+4*x+1
:_nfit=3
:_execute fit
  EXECUTION STARTED
    THE BEST FIT IS
      AFIT (A VECTOR WITH 3 COMPONENTS)
        1 4 3
:_edit
  O.K.-EDIT MODE
:_%list
%1      PROGRAM FIT
%2      A=ARRAY(NOELS(X),NFIT:); I=INTEGERS(1,NFIT)
%3      A=MFAM((X+A)**(I-1)); Y=VFAM(Y)
%4      AFIT=1/(TRANSPOSE(A)*A)*TRANSPOSE(A)*Y
%5      PRINT('    THE BEST FIT IS ',AFIT)
%6      END
:_%5.5 yfit=a*afit ; tabulate x y yfit
:_%run
  EXECUTION STARTED
    THE BEST FIT IS
      AFIT (A VECTOR WITH 3 COMPONENTS)
        1 4 3
      X Y   YFIT
      1 8   8
      2 21  21
      3 40  40
      4 65  65
:_x=1.1 2.03 3.34 4.43 5.16
:_y=2.78 4.4 10.7 19.4 27.1
:_execute
  EXECUTION STARTED
    THE BEST FIT IS
      AFIT (A VECTOR WITH 3 COMPONENTS)
      3.8691 -2.4795 1.3526
      X Y   YFIT
      1.1 2.78 2.7782
      2.03 4.4 4.4095
      3.34 10.7 10.676
      4.43 19.4 19.429
      5.16 27.1 27.087

```

B. INPUT/PAUSE/STOP

In interactive usage of the program mode of SPEAKEASY, it is often desirable to interrupt the computation at specific places but to retain the capability of resuming at that point. Three statements are available in the program mode for this purpose. All are identical in operation; the choice between them is purely subjective. If the first word on a SPEAKEASY statement encountered in the execution mode is INPUT, PAUSE, or STOP, then the entire statement is printed out and the execution of the program is interrupted. The system is put into a mode referred to as the HOLDING mode.

This mode is in fact the MANUAL mode with the added capability of resuming the execution of the SPEAKEASY program at a later time. All of the facilities of the manual mode are available. New objects can be defined, old results can be examined, etc. Whenever the objectives of the interrupt have been met, then the execution can be resumed by entering a statement with one of the words RESUME, CONTINUE, or GO. If one desires to terminate the HOLDING mode, the statement MANUAL places the system in the true manual mode. In operation a statement of the form

INPUT A, B AND C PLEASE

encountered during execution would print out

INPUT A, B AND C PLEASE

and the user might then type in

A = 4; B = 7; C = 27.48*W; RESUME

Note that the operations in the HOLDING mode are completely general and need not be restricted to the implied requests. The only restriction is that the EXECUTION mode itself may not be used. If it is used, then one loses the ability to resume from this point at some later time.

VI. VERSIONS

SPEAKEASY is a general processing language. The modes of operation, even at Argonne, are rather diverse. Card-input, remote-job-entry, and remote-job-output facilities all imply slightly different optimal forms of operation. In contrast, in the truly interactive mode (e.g., on the IBM-2250 console or the newly available TSO version) the user may need a quite different form. Instead of attempting to construct specialized processors for each application, the approach taken has always been to include as many diverse capabilities in the basic processor as possible. Each use can then select the available ones that are most clearly desirable in the application. Specialized input and output requirements are met by isolating them in two or three replaceable modules. These can easily be adapted to special devices such as the 2250.

Several different versions of the SPEAKEASY processor are now available. The computational logic is identical in various versions. They differ primarily in their space requirements and in their efficiency of operation. It is expected that some of these versions will be combined in the near future. The following versions are available.

A. STANDARD

This is the standard production version of the language for batch processing. It is a non-overlaid version that occupies 260K^{*} of core. This version is the fastest running version available

*The size given is a nominal one. It was selected on the assumption that the size of the LINKULES does not exceed 10 K.

but does require the largest amount of computer core. The default settings can be viewed as

```
SIZE = 40, MAIN
MARGINS (1, 128)
LIST
ECHO
```

B. BABY

This is a heavily overlaid version of the above processor and is designed to provide rapid batch turnaround at the sacrifice of execution efficiency. The default settings are

```
SIZE = 40, MAIN
MARGINS (1, 128)
LIST
ECHO
```

This processor requires about 160K to operate.

C. GRAPHEZ

The graphical facilities of the language (for use with the CALCOMP 780 device) are maintained only in this special version. The version is identical to the standard version described above except for these additional features. This version requires 280K to operate.

D. CONSOLE

This version of the language is adapted to the 2250 display console. It is a heavily overlaid version that is designed for

export. The operation of this version is similar to that of the TSO version except that graphical output is directly available. Since the display is on an oscilloscope, a monitor copy of the input and output is also produced on a line printer for later reference. The defaults are

SIZE = 8
MARGINS (1, 70)
LIST
ECHO

E. SPEK2250

This is the production console version of the language for use at Argonne. It is a non-overlaid version, but it is designed for use with LCS as the primary core storage. This tailored version is similar to the CONSOLE version except for the use of LCS.

F. TSO

The adaption of SPEAKEASY to operation with TSO is relatively new. The major difficulty has been one of trimming the processor to a size acceptable for use in TSO regions of normal size. Since each installation chooses this size to meet its specific needs, no single generally acceptable size has yet been established. In decreasing the size of the standard version, the already small version BABY has been further overlaid. The version that is now available will operate in a 120K TSO region. Further decreases in this size are expected; but since further decreases become more and more difficult, it is unlikely that a much smaller version will be

available soon. The default options for this version are

SIZE = 4

MARGINS (1, 128)

NOLIST

NOECHO

VII. ACKNOWLEDGMENTS

SPEAKEASY is a language developed to serve its user community. Complaints and praise by the users, though apparently ignored in the short term have gradually influenced the structure of the language. It is through discussions with users, particularly in the Physics Division that defects and desirable facilities are first noted. The form and variety of the features available is therefore mostly due to the users themselves. I wish to thank those who have expressed their needs and desires and who have thus influenced this development. In particular special thanks is due to Frank Serduke, Martin Vincent and Keith Rich who have been generous with their time and who have been directly involved in many of the major discussions that have influenced the development of the language. Since the language continues to evolve it is hoped that such direct influences will continue.

VIII. REFERENCES

- 1) An Introduction to SPEAKEASY, S. Cohen and C. M. Vincent, Informal Report PHY-1968E, Physics Division, Argonne National Laboratory, 1968.
- 2) SPEAKEASY-3 Linkules and Interfaces, S. Cohen, R. Kimmel, K. Rich and F. J. D. Serduke, Informal Report, PHY-1972C, Physics Division, Argonne National Laboratory, 1972.
- 3) The SPEAKEASY HELP Documents, J. K. Fink, Informal Report PHY-1972G, Physics Division, Argonne National Laboratory, 1972.