Department of Computer Science

Technical Report CS-73-03

February 1973

THE CLASSPAK SYSTEM

APL Routines for the Automatic Manipulation of Classroom Data

by

Patrick C. Fischer

This work was supported by the National Research Council of Canada, Grant No. A-5549.

0. Foreword.

Since the CLASSPAK system was first documented in Technical Report CSTR 1008, November, 1970, there has been a steady demand for the CLASSPAK manual. The original report is now out of print.

Although it had been hoped to incorporate a number of suggestions into a CLASSPAK3, this version has not yet been written. The major goal here would be to allow use of CLASSPAK for very large classes. What needs to be done depends critically upon the workspace sizes and file system available in an installation. In the interim, this slightly revised version of the manual is being issued.

CLASSPAK 1 and CLASSPAK2 are currently available to qualified users at the University of Waterloo, the University of Guelph and Lakehead University. In the first instance, the workspaces are available in Library 3 as depicted in Example 0, page 7. In other cases, potential users should check with the consultants at their own installation. Mathematical Faculty users at Waterloo who are not running on the 360/44 should ask the author for information on how to obtain a copy of CLASSPAK.

The only major change to the system in the past two years has been the addition of the GRAPH function. A description of this appears on page 17. A few typographical errors have been corrected in the earlier material and few bugs have been removed from the system.

1. Introduction.

The CLASSPAK system has been designed to make it possible to enter, manipulate and retrieve information relevant to student performance in a course from a selectric teletypewriter terminal (e.g. IBM 2741 or Datel 30). The system is implemented in the APL language, but only a minimal number of system commands need be learned in order to use CLASSPAK.

The data is organized as given in Fig. 1. Student membership in one or more sections is indicated by a single bit in a Boolean matrix, and student marks are recorded in a matrix of integers. The number of student names, number of sections, number of columns for marks, and the number of characters stored in a student name are all system parameters which are set by the user before any data is entered. (The system names for these variables are given in Sect.7, but need never be used directly.)

In general, reference may be made either to row, section and column numbers or to the appropriate names and titles, in which case the system will automatically look up the corresponding numbers. The precise syntax of the language is given in later sections. Function commands are mnemonic, e.g., EDIT, LIST, ALTER, ANALYZE have the meanings one would expect.

There are two versions of CLASSPAK, which differ only in the way the mark information is stored. Version 1 will handle up to approximately 150 students with 24 mark columns, and a single mark entry may be any integer between -9 and +999. -1 through -9 have special meanings such as no mark, DNW, ABS, EXC, etc.). Version 2 can handle approximately twice as many students. However, the mark entries are restricted to the range -9 through +100. Again, the marks must be integers.

Since the system works in the interactive mode and a reasonable amount of diagnostics are present, it is suggested that the best way to become familiar with the system is to experiment with it. A reasonable amount of success has been achieved by persons who have not even had the early version of this manual, and have therefore learned directly from the responses of the computer.

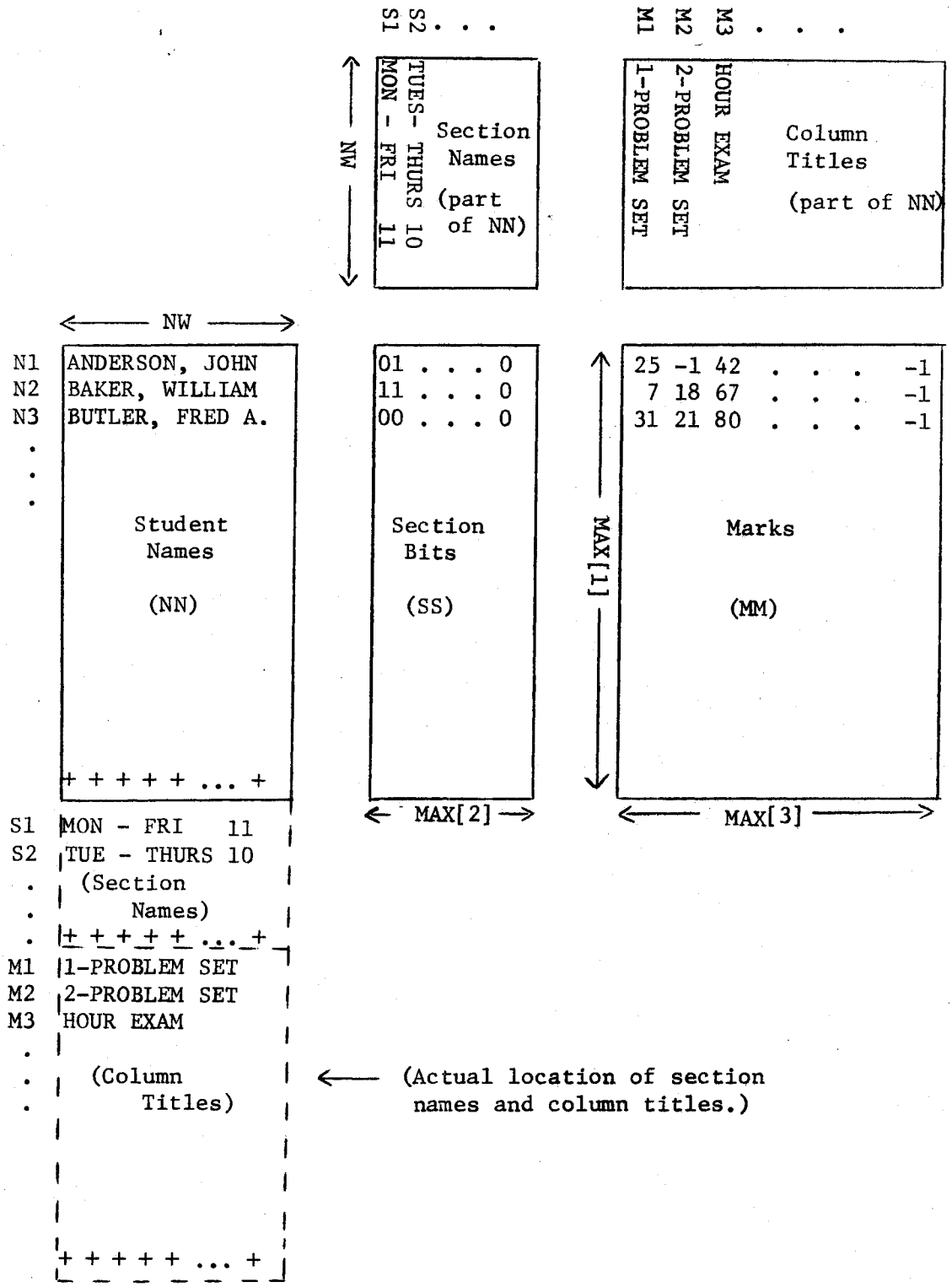

Fig. 1 - Data Organization in CLASSPAK

2. Preliminaries.

In order to use the CLASSPAK system, one does not need to know APL notation. However, there is a rock-bottom minimum of knowledge required, namely how to log on and off the APL system and to load and save workspaces. The IBM APL manual or the abbreviated Waterloo manual "A Taste of APL" contain the necessary information. The following APL commands are necessary to use CLASSPAK:

```
)number:lock          (log-in sequence)
)LOAD   wsname:wslock
)SAVE
)OFF
```

Do not forget to)SAVE the appropriate workspace after a session before signing off. [See also the commands)CONTINUE,)WSID .]

Once a copy of the CLASSPAK system has been loaded into a user's workspace, the function INITIALIZE must be called to set up the variables pertaining to the particular class being handled. INITIALIZE asks for maximum numbers of students, sections and mark columns to be stored and the number of characters to be allowed for each name or title. The initialization will then be performed if there is sufficient space to do so, and a request will then be made that the user)ERASE INITIALIZE SETCONST DESCRIBE. This should be done in order to gain additional workspace. INITIALIZE will never need to be called again for the same class. If insufficient workspace is available, INITIALIZE will say so and do nothing. (See Example 0.)

It is possible but complicated in APL (and not possible directly in CLASSPAK) to revise maxima upwards after data has been entered. Therefore, some allowance should be made for late registrants, etc. On the other hand, if the name-width and other parameters cannot be trimmed sufficiently to satisfy INITIALIZE, version 2 should be considered, or the course data should be split into two separate workspaces.

The following syntactic categories relate to the permissible inputs of the functions in CLASSPAK. Some of them are defined recursively in terms of others, e.g., $\underline{V}_R$ and $\underline{U}_I$.

$\underline{I}$: a positive integer

$\underline{V}$: a vector of positive integers, i.e., $\underline{V}$ ::= $\underline{I}$ | $\underline{V}$ $\underline{I}$

$\underline{I}_Z$: a non-negative integer, i.e., $\underline{I}_Z$::= $\underline{I}$ | 0

$\underline{V}_Z$: a vector of positive integers or a single zero,
i.e., $\underline{V}_Z$::= $\underline{V}$ | 0

$\underline{E}$: N, S, or M (student Name, Section, Mark column,
i.e., $\underline{E}$ ::= N | S | M

$\underline{V}_R$: a vector describing a list of row indices (student names);
i.e., one of:

$\underline{V}_R$::= ALL N | ROWS $\underline{V}_Z$ | SECT $\underline{I}_Z$ | N, $\underline{V}$ | MIDDLE $\underline{V}_R$ | MIDDLE $\underline{I}$,$\underline{V}_R$ | RANK $\underline{U}_I$ | RANGE $\underline{U}_I$ | ($\underline{V}_R$) AND ($\underline{V}_R$)

$\underline{V}_S$: a vector describing a list of section indices; one of:

$\underline{V}_S$::= S | S, $\underline{V}$ | ALL S | MIDDLE ALL S

$\underline{V}_M$: a vector describing a list of mark column indices; one of:

$\underline{V}_M$::= M | M, $\underline{V}$ | ALL M | MIDDLE ALL M

$\underline{U}_I$: a vector referring to a single (section or mark) column and several rows; one of:

$\underline{U}_I$::= 0 | $\underline{I}$ | $\underline{V}_R$ | $\underline{I}$, $\underline{V}_R$

$\underline{U}_S$: a vector referring to several sections and/or several rows; one of:

$\underline{U}_S$::= 0 | $\underline{V}_R$ | $\underline{V}$ | $\underline{V}$, $\underline{V}_R$ | $\underline{V}_S$ | $\underline{V}_S$, $\underline{V}_R$

$\underline{U}_M$: a vector referring to several mark columns and/or several rows; one of:

$\underline{U}_M$::= 0 | $\underline{V}_R$ | $\underline{V}$ | $\underline{V}$, $\underline{V}_R$ | $\underline{V}_M$ | $\underline{V}_M$, $\underline{V}_R$

In general, when the column or row indices are not known, they may be omitted and the system will ask the user for the appropriate titles or names and look up the indices internally. However, in APL some input is always required for a function, so if no indices of any kind are present, 0 should be given as input. The use of 0 as an input should be restricted to such a case, although the system is somewhat tolerant of misapplication of this rule.

3. Functions Which Enter or Manipulate Alphabetic Information.

At times the system will ask for alphabetic input in order to enter new data (ADD) or to specify uniquely which students and sections or mark columns are involved. When data is called for in the singular, one item should be entered. When data is called for in the plural, e.g., STUDENT NAMES, it is permissible to enter more than one item at a time, separated by semicolons. In this case, if the rightmost character entered is a semicolon, more input will be called for after the input in the current line has been processed.

ADD E (Examples 1-3)

This function adds student names, section names or mark column titles to those already entered. Items must contain only blank, period, comma, A-Z, 0-9 and -(minus sign for hyphen), and must be no longer than the name-width (NW) set during INITIALIZE.

DELETE E (Examples 4-5)
DELETE E, I

If an integer I is present in a call for DELETE and it is a valid E-index, the appropriate row, section or mark column will be involved. Otherwise, alphabetic input will be requested, and an E-item having its initial segment matching the alphabetic input will be used. In the event that no items match or more than one match, DELETE will object. DELETE is fail safe in that the designated item is printed out in full and the user must then verify that deletion is to proceed by typing 'D' or 'Y'. If the name (or title) is deleted, the associated section bits and marks are also deleted.

EDIT E (Examples 6-7)
EDIT E, I
EDIT E, I_Z, K ($K < NW$).

If K is omitted, the meaning of the argument is the same as for DELETE. An E-item designated uniquely will be printed out. If no

change is then desired, hitting carriage return will terminate EDIT. Any non-blank entry will replace the original name (or title). Section bits or marks associated with the item will be unaltered.

If K is present and non-zero, then the first K characters of the E-item are left unchanged and editing begins with the K+1-st position. In this case a carriage return or blank input will cause blanks to replace the K+1-st and subsequent characters in the name or title.

LIST E, V (Examples 8-11, 13)
LIST E
LIST ALL E
LIST V_R (e.g., LIST SECT 2)

LIST prints out the E-items involved, but not the associated section bits or marks. If LIST E is used, alphabetic input will be required and all items matching the input will be listed. This makes it possible to LIST all student names beginning with B, etc.

SORT E (Examples 12-13)
SORT E, K (K < NW)

If K is present, the E-items are sorted in alphanumeric order on the first K characters. Otherwise they are sorted on all characters. The associated section bits and marks are permuted along with the name (or title) items. SORT must not be interrupted during the permutation phase. It would be desirable to execute a ")SAVE" immediately before using SORT to protect the data base in the event of an accidental disconnect, etc. If a WSFULL is encountered, see Section 7.

```
      EXAMPLE  0

      )LOAD 3 CLASSPAK1
SAVED  25.40.02 11/05/70
      )WSID TEST1
WAS 3 CLASSPAK1

      INITIALIZE
TYPE IN MAXIMUM NUMBERS FOR NAMES, SECTIONS, AND MARK COLUMN TITLES:
□:
      100 16 24
--NUMBER OF CHARACTERS IN A NAME OR COLUMN TITLE:
□:
      20
INITIALIZATION COMPLETE.
PLEASE )ERASE INITIALIZE SETCONST DESCRIBE

      )ERASE INITIALIZE SETCONST DESCRIBE



      EXAMPLE  1

      ADD S
--SECTION NAMES TO BE ADDED:
MONDAY 11AM
MONDAY 11AM ↔ SECTION 1
      ADD S
--SECTION NAMES TO BE ADDED:
FRIDAY
FRIDAY ↔ SECTION 2



      EXAMPLE  2

      ADD N
--STUDENT NAMES TO BE ADDED:
SMITH, ALBERT B.;BARNUM, P.T.;BAKER, RICHARD A.;
SMITH, ALBERT B. ↔   ROW   1
BARNUM, P.T. ↔   ROW   2
BAKER, RICHARD A. ↔   ROW   3
--STUDENT NAMES TO BE ADDED:
HUMPERDINCK, ENGLEBERT;JONES, ROBERT F.; DUCK, DONALD;
HUMPERDINCK, ENGLEBE  TRUNCATED TO 20 CHARACTERS.  RETYPE IF DESIRED.
HUMPERDINCK, ∈.
ILLEGAL CHARACTER IN  HUMPERDINCK, ∈..  PLEASE RETYPE.
HUMPERDINCK, E.
HUMPERDINCK, E. ↔   ROW   4
JONES, ROBERT F. ↔   ROW   5
DUCK, DONALD ↔   ROW   6
--STUDENT NAMES TO BE ADDED:
BROWNING, ELIZABETH B.
BROWNING, ELIZABETH   TRUNCATED TO 20 CHARACTERS.  RETYPE IF DESIRED.

BROWNING, ELIZABETH ↔   ROW   7
```

EXAMPLE 3

```
ADD M
--COLUMN TITLES TO BE ADDED:
P1-PROB SET 1;P2-PROB SET 2;HOUR EXAM;AVG
P1-PROB SET 1 ↔  COLUMN 1
P2-PROB SET 2 ↔  COLUMN 2
HOUR EXAM ↔  COLUMN 3
AVG ↔  COLUMN 4
```

EXAMPLE 4

```
DELETE N
--STUDENT NAME TO BE DELETED:
DUCK
DUCK, DONALD          ↔   ROW   6.  DELETE?
Y
--DELETED--
```

EXAMPLE 5

```
DELETE N
--STUDENT NAME TO BE DELETED:
BA
BA   AMBIGUOUS.  THE FOLLOWING MATCH:

BARNUM, P.T.
BAKER, RICHARD A.
--STUDENT NAME TO BE DELETED:
BAK
BAKER, RICHARD A.     ↔   ROW   3.  DELETE?
N
-NOT DELETED-
```

EXAMPLE 6

```
EDIT S,0,6
--SECTION NAME TO BE CHANGED:
DRIDAY
DRIDAY  NOT FOUND.
--SECTION NAME TO BE CHANGED:
FRI
FRIDAY                ↔ SECTION 2
      ↑ CORRECTION:
 10AM
FRIDAY 10AM ↔ REVISED SECTION 2
```

```
      EXAMPLE  7

      EDIT M,4
AVG                    ↔  COLUMN 4
↑ CORRECTION:
AVERAGE
AVERAGE ↔ REVISED  COLUMN 4
```

```
      EXAMPLE  8

      LIST ALL M

NO.  COLUMN TITLES  (4)

001  P1-PROB SET 1
002  P2-PROB SET 2
003  HOUR EXAM
004  AVERAGE
---END---
```

```
      EXAMPLE  9

      LIST N
--STUDENT NAMES:
H;BA;F
BA  AMBIGUOUS.
F  NOT FOUND.

NO.  STUDENT NAMES  (3)

004  HUMPERDINCK, E.
002  BARNUM, P.T.
003  BAKER, RICHARD A.
---END---
```

```
      EXAMPLE  10

      LIST SECT 1
SECTION 1, MONDAY 11AM

NO.  STUDENT NAMES  (0)
```

```
      EXAMPLE  11

      LIST MIDDLE 4, ALL N

NO.  STUDENT NAMES  (3)

004  HUMPERDINCK, E.
005  JONES, ROBERT F.
007  BROWNING, ELIZABETH
---END---
```



```
      EXAMPLE  12

      SORT N
DO NOT INTERRUPT PERMUTATION PHASE.
SORT OF STUDENT NAMES COMPLETED.
```

```
      EXAMPLE  13

      LIST ALL N

NO.  STUDENT NAMES  (6)

001  BAKER, RICHARD A.
002  BARNUM, P.T.
003  BROWNING, ELIZABETH
004  HUMPERDINCK, E.
005  JONES, ROBERT F.
006  SMITH, ALBERT B.
---END---
```

4.4. Functions Which Manipulate Section Bits

The syntactic forms $\underline{U}_I$ and $\underline{U}_S$ (see Section 2) denote the catenation of column and row indices. If indices of either kind (or both) are missing, the system will ask for the appropriate alphabetic input and attempt to match initial segments of the appropriate name (or title) items in the list.

INCREASE SECTION $\underline{U}_I$ (Example 14)

The bits in the designated rows of the section involved are set to 1. Other bits are unchanged.

DECREASE SECTION $\underline{U}_I$ (Example 15)

The bits in the designated rows of the section involved are set to 0. Other bits are unchanged.

RESET SECTION $\underline{U}_I$

The section involved is cleared, and the bits in the designated rows are then set to 1.

COMPLEMENT SECTION $\underline{U}_I$ (Example 16)

The section involved is set to the complement of the set of designated rows.

SETALLO SECTION $\underline{I}_Z$, N, 1

The section involved is cleared. N, 1 is used as a dummy argument.

SETALL1 SECTION $\underline{I}_Z$, N, 1

The section involved is set to ALL N. N, 1 is used as a dummy argument.

INTERSECT SECTION $\underline{U}_I$ (Example 17)

The section involved is replaced by the intersection of itself and the set of designated rows.

SECEDIT $\underline{U}_S$ (Example 18)

This function lists the names and section numbers of the sections involved (unless $\underline{U}_S$ is of the form $\underline{V}_S$ or $\underline{V}_S$, $\underline{V}_R$, in which case the section names are suppressed). For each row involved, the name and present section bits are listed (if all bits involved are 0, they are omitted). New bits to replace the old ones are requested. Inputs of the wrong length or type are rejected. To leave the line as it is, type "SKIP". To quit before all rows have been processed, type "→".

SECBITS $\underline{U}_S$ (Example 19)

The names and section bits in the designated rows and columns are listed. If $\underline{U}_S$ is of the form $\underline{V}_S$ or $\underline{V}_S$, $\underline{V}_R$, a rather lengthy header of section titles is suppressed; otherwise it is printed. (In order to print the header when printing all section columns, use "SECBITS 1↓ALL S, ALL N" .)

EXAMPLE 14

```
      INCREASE SECTION 1, ALL N
RESULT IN SECTION 1, MONDAY 11AM
```

EXAMPLE 15

```
      DECREASE SECTION 1
--STUDENT NAMES:
HUM;JONES
RESULT IN SECTION 1, MONDAY 11AM
```

EXAMPLE 16

```
      COMPLEMENT SECTION 2, SECT 1
SECTION 1, MONDAY 11AM
RESULT IN SECTION 2, FRIDAY 10AM
```

EXAMPLE 17

```
      INTERSECT SECTION ROWS 1 2 3
--SECTION NAME FOR RESULT:
MON
RESULT IN SECTION 1, MONDAY 11AM
```

EXAMPLE 18

```
      SECEDIT 0
--SECTION NAMES:
MONDAY;FRIDAY
--STUDENT NAMES:
JO;L;S
L  NOT FOUND.

MONDAY 11AM
FRIDAY 10AM                1  2

2 ITEMS.

JONES, ROBERT F.           0  1
□:
      SKIP
SMITH, ALBERT B.
□:
      1 1
---END---
```

EXAMPLE 19

```
      SECBITS ALL S, ALL N


NAMES              (  6) 1 2

BAKER, RICHARD A.        1 0
BARNUM, P.T.             1 0
BROWNING, ELIZABETH      1 0
HUMPERDINCK, E.          0 1
JONES, ROBERT F.         0 1
SMITH, ALBERT B.         1 1
---END---
```

5. Functions which Manipulate Marks

All mark-manipulating functions take arguments either of the form $\underline{U}_I$ or $\underline{U}_M$. Any subset of rows and any subset of columns of the mark matrix can be specified by the appropriate catenation of column and row indices. If indices of either kind (or both) are missing, the system will ask for the appropriate alphabetic input and attempt to match it to initial segments of the name (or title) items in the name-title list.

ENTER $\underline{U}_I$ (Example 20)

Names corresponding to the designated rows are listed, 5 at a time. If any marks are already present in the designated column, these are also listed. Numeric inputs are then called for. Inputs of the wrong length or size are rejected. (Version 1 and version 2 have different size criteria; see Section 1.) To leave the entire group as it is, type in "SKIP". To quit before all rows have been processed, type "→". It should be noted that ENTER is really a columnwise version of ALTER since it can process areas in which some marks are already present.

ALTER $\underline{U}_M$ (Example 21)

Marks may be altered for several columns, one row at a time. First, the titles and numbers of the mark columns involved are listed (unless $\underline{U}_M$ is of the form $\underline{V}_M$ or $\underline{V}_M$, $\underline{V}_R$, in which case the column titles are suppressed). For each row involved, the student name and present marks are listed. (If all mark positions are blank, i.e., contain $\bar{1}$, they are omitted.) New marks to replace the old ones are requested. Inputs of the wrong length or size are rejected. To leave the row as it is, type "SKIP". To quit before all rows have been processed, type "→".

Note: the symbol N has a value of $\bar{1}$, it may be used in giving input to ENTER or ALTER to mean "no mark", e.g., 35 24 57 $\bar{1}$ 68 is equivalent to 35 24 57,N, 68.

PRINT $\underline{U}_M$ (Examples 22, 23, 26)

The names and marks in the designated rows and columns are listed. If $\underline{U}_M$ is of the form $\underline{V}_M$ or $\underline{V}_M$, $\underline{V}_R$, the header is suppressed. (In order to print the header when printing all mark columns, use PRINT 1↓ALL M, $\underline{V}_R$.)

```
      EXAMPLE  20

      ENTER 1, ALL N
 COLUMN 1, P1-PROB SET 1

6 ITEMS.

BAKER, RICHARD A.
BARNUM, P.T.
BROWNING, ELIZABETH
HUMPERDINCK, E.
JONES, ROBERT F.
□:
      25 28 27 17 30

SMITH, ALBERT B.
□:
      22
---END---
```

```
      EXAMPLE  22

      PRINT SECT 1
SECTION 1, MONDAY 11AM
--COLUMN TITLES:
P;HOUR
P  AMBIGUOUS.

NAMES          (  4)  P1-P  P2-P  HOUR
                        1    2     3

BAKER, RICHARD A.       25   22    88
BARNUM, P.T.            28   20    75
BROWNING, ELIZABETH     27    -     -
SMITH, ALBERT B.        22    -     -
---END---
```

```
      EXAMPLE  21

      ALTER 2 3
--STUDENT NAMES:
BA;JONES
BA  AMBIGUOUS.

P2-PROB SET 2
HOUR EXAM                2  3

3 ITEMS.

BAKER, RICHARD A.
□:
      22 88
BARNUM, P.T.
□:
      2075
INPUT WRONG LENGTH OR SIZE.
□:
      20 75
JONES, ROBERT F.
□:
      N,66
---END---
```

```
      EXAMPLE  23

      PRINT ALL M, ROWS 1 2 5 5 4
BAKER, RICHARD A.       25   22    88    -
BARNUM, P.T.            28   20    75    -
JONES, ROBERT F.        30    -    66    -
JONES, ROBERT F.        30    -    66    -
HUMPERDINCK, E.         17    -     -    -
---END---
```

ANALYZE $\underline{U}_M$ (Example 24)

The marks in the designated rows are analyzed for each column involved. Given for each column are the number of marks, number with no mark, highest mark, 75th percentile, median, 25th percentile, lowest mark, mean and standard deviation.

COMPUTE $\underline{U}_M$ (Examples 25, 26)

The inputs for COMPUTE designate a list of columns and a list of rows. The first member of the column list denotes the location of the results, and the other members designate the arguments. COMPUTE asks for a numeric constant to be added to each result and for a vector of coefficients to be applied to the entries in each of the argument columns. It then cycles through each of the designated rows in turn and computes the result for that row using

$$f(x_1, x_2, \ldots, x_k) = c + \sum_{i=1}^{k} a_i x_i$$

where c is the constant, and the a_i's are the coefficients. Negative mark entries are treated as zeros in this computation.

It is possible to use a mark column as both argument and result column, viz. COMPUTE 1 1, ALL N. In this case a linear transformation of the marks in column 1 would be stored in column 1.

It is also possible for a user to write his own function for COMPUTE. When the constant term is requested, if the user types in "SKIP", COMPUTE will call a user defined function called FUNCT. In this case, as COMPUTE cycles through the designated rows, it will feed each row vector of marks to FUNCT and receive the output from FUNCT. After rounding to the nearest integer and checking for proper size, the result will be stored in the results column as before. Specifications for a user-defined FUNCT are given in Sect. 7, part H.

APL numeric inputs may be expressions. In example 25 this property is used to have P1, P2 and HOUR EXAM contribute 20, 20 and 60, respectively to AVERAGE. It is assumed that the maximum possible marks were 30, 30 and 100, respectively.

EXAMPLE 24

ANALYZE 1 2 3, *ALL N*

COLUMN TITLE	*P1-PR*	*P2-PR*	*HOUR*
COLUMN NUMBER	1	2	3
NUMBER OF MARKS	6	2	3
HAVING NO MARK	0	4	3
HIGHEST MARK	30	22	88
75TH PERCENTILE	28	22	81
MEDIAN MARK	26	21	75
25TH PERCENTILE	22	20	70
LOWEST MARK	17	20	66
AVERAGE MARK	24.8	21.0	76.3
STD. DEVIATION	4.3	1.0	9.0

EXAMPLE 25

COMPUTE 4 1 2 3, *ALL N*

ARGUMENT COLUMNS 1 2 3

P1-PROB SET 1
P2-PROB SET 2
HOUR EXAM

--CONSTANT:
□:
0
--COEFFICIENTS:
□:
20 20 60 ÷ 30 30 100
RESULT IN COLUMN 4, *AVERAGE*

EXAMPLE 26

PRINT 1↓*ALL M*, *ALL N*

NAMES (6)	*P1-P*	*P2-P*	*HOUR*	*AVER*
	1	2	3	4
BAKER, RICHARD A.	25	22	88	84
BARNUM, P.T.	28	20	75	77
BROWNING, ELIZABETH	27	-	-	18
HUMPERDINCK, E.	17	-	-	11
JONES, ROBERT F.	30	-	66	60
SMITH, ALBERT B.	22	-	-	15

---END---

EXAMPLE 27

LIST (SECT 0) *AND (ROWS* 0)
--STUDENT NAMES:
BA;H
BA AMBIGUOUS.
--SECTION NAME:
FRIDAY
SECTION 2, *FRIDAY* 10*AM*

NO. STUDENT NAMES (6)

004 *HUMPERDINCK, E.*
005 *JONES, ROBERT F.*
006 *SMITH, ALBERT B.*
001 *BAKER, RICHARD A.*
002 *BARNUM, P.T.*
004 *HUMPERDINCK, E.*
---END---

GRAPH U_I (Example 27A)

The marks in the designated rows and the single column specified are sorted and tabulated. In response to "BEGINNING, INTERVAL SIZE, END" the user should type 3 integers. Highest marks will be plotted first if the first number is greater than the third; otherwise the lowest marks will be plotted first. The last interval plotted may be smaller than the interval size. The function automatically changes scale for large classes so that the probability that the horizontal histogram will run off the page is small. The number of mark entries not plotted by virtue of their not being between the beginning and end points is given at the end of the plot.

```
      EXAMPLE 27A

      GRAPH 4, ALL N
--BEGINNING, INTERVAL SIZE, END:
□:
      100 15 0

INTERVAL     PCT 0     5    10    15    20    25    30    35    40    45    50
                 |     |     |     |     |     |     |     |     |     |     |
100 - 86     0.0 |
 85 - 71    33.3 |□□   2
 70 - 56    16.7 |□   1
 55 - 41     0.0 |
 40 - 26     0.0 |
 25 - 11    50.0 |□□□   3
 10 -  0     0.0 |

SUBTOTAL  100.0     6          COLUMN 4, AVERAGE
OTHERS      0.0     0
TOTAL               6
```

6. Auxiliary Functions.

Seven functions may be used in conjunction with the functions given in Sections 3-5. In each case the output of an auxiliary function is a vector in the proper format for a vector of row, section or mark column indices. The functions ALL, MIDDLE and AND can generate any of the three types of indices (N, S or M), although, in general, MIDDLE and AND are used only in conjunction with row indices. The other four auxiliary functions only generate vectors of row indices.

ALL E (Examples 13, 26)

All current E-indices are generated.

MIDDLE ALL E	MIDDLE I, ALL E	(Example 11)
MIDDLE E, V	MIDDLE I, E, V	
MIDDLE V_R	MIDDLE I, V_R	

If I is not present, MIDDLE asks for alphabetic input. The index found by matching (or I) is compared against its vector of arguments. The output is the vector of indices obtained by deleting those occurring before the designated index. If the designated index is not in the list, the value of MIDDLE will be E, and this will be interpreted as an empty list. If no index matches the alphabetic input (if called for) the value of MIDDLE is undefined.

NOTE: When using auxiliary functions, e.g., "LIST MIDDLE ALL M", if the value of MIDDLE or of ALL becomes undefined due to lack of valid input, APL will give an error indication (usually SYNTAX ERROR). This does not indicate a CLASSPAK system error. See also Sect. 7, part D.

ROWS V_Z (Examples 17, 23, 27)

If $V_Z = 0$, alphabetic input is asked for and indices of student names matching the input are generated. Otherwise, the indices in V are checked for validity and used directly. If invalid input is detected, the value of ROWS is undefined.

SECT $\underline{I}_Z$ (Example 27)

If $\underline{I}_Z$ is a valid section index, the output of SECT is the vector of row indices corresponding to membership in that section (i.e. having section bits set to 1). If $\underline{I}_Z = 0$, then a section name is asked for as input and matched. If no match is obtained, the value of SECT is undefined.

($\underline{V}_R$) AND ($\underline{V}_R$) (Example 27)

The items conjoined by AND should be enclosed in parentheses. Multiple use of AND, e.g., "LIST (SECT 1) AND (SECT 2) AND (ROWS 4 5)" is permitted.

RANK $\underline{U}_I$ (Examples 28, 30)

As with ENTER, a single column and a set of rows are designated. Missing items will cause the system to ask for alphabetic data for matching purposes. The output of RANK is a vector of row indices such that the associated marks in the column involved occur in descending order as one cycles through the list generated by RANK.

RANGE $\underline{U}_I$ (Examples 29, 30)

RANGE selects from the designated vector of row indices only those indices for which the marks in the designated column lie within the range given (including the end points of the range interval). The range limits are always called for as input; they are <u>not</u> part of $\underline{U}_I$.

The auxiliary functions may be combined in many ways. Thus, the following statement is admissible in the CLASSPAK system:

PRINT ALL M, RANK RANGE (MIDDLE SECT 1) AND (ROWS 0)

```
      EXAMPLE  28

      PRINT 4 3 1 2, RANK 4, ALL N
RANK ORDER FOR  COLUMN 4, AVERAGE

NAMES           (  6)  AVER HOUR P1-P P2-P
                         4    3    1    2

BAKER, RICHARD A.       84   88   25   22
BARNUM, P.T.            77   75   28   20
JONES, ROBERT F.        60   66   30    -
BROWNING, ELIZABETH     18    -   27    -
SMITH, ALBERT B.        15    -   22    -
HUMPERDINCK, E.         11    -   17    -
---END---
```

```
      EXAMPLE  29

      PRINT ALL M, RANGE ROWS R
--COLUMN TITLE FOR RANGE:
AVER
--RANGE LIMITS:
□:
      0 60
MARKS FROM 0 TO 60 IN COLUMN 4, AVERAGE
JONES, ROBERT F.        30    -   66   60
BROWNING, ELIZABETH     27    -    -   18
SMITH, ALBERT B.        22    -    -   15
HUMPERDINCK, E.         17    -    -   11
---END---
```

```
      EXAMPLE  30

      PRINT C, RANK 4, RANGE 4, ALL N
--RANGE LIMITS:
□:
      60 100
MARKS FROM 60 TO 100 IN COLUMN 4, AVERAGE
RANK ORDER FOR  COLUMN 4, AVERAGE

NAMES           (  3)  P1-P P2-P HOUR AVER
                         1    2    3    4

BAKER, RICHARD A        25   22   88   84
BARNUM, P.T.            28   20   75   77
JONES, ROBERT F.        30    -   66   60
---END---
```

7. Miscellaneous Comments.

A. The use of C and R. (Examples 29, 30)

When one of the functions in Sections 4 and 5 is used, the indices of the (section or mark) columns involved are stored in C and those for the designated rows are stored in R. Each of these may be used as an input to the next operation. Since R is of the syntactic form V, it must be used in conjunction with N or with ROWS. Thus, if ALTER has been used to change several rows and columns, one may simply use "PRINT C, ROWS R" or "PRINT C, N, R" in order to list the new marks for proofreading.

B. Termination of a long printout or computation.

The "Attention" button may be used in APL to terminate printing or execution as well as to correct input. However, APL saves the conditions of an interrupted execution, and one's limited workspace can insidiously disappear through improper use of this feature. In order to avoid this situation, after interrupting a printout, type in a "→". This will clear out the state indicator and restore the available workspace.

C. Termination of a routine when it is asking for input.

If alphabetic input is being called for (in which case the comment ends with a colon or a question mark), CLASSPAK is programmed so that a simple carriage return will terminate the function. If numeric input is being requested (via □ :), type in a "→".

D. VALUE ERROR or SYNTAX ERROR.

These are APL objections because of reference to an undefined symbol (e.g., "PRNIT") or improper APL syntax. Unless APL prints a line such as DECODE[3] followed by a string of APL characters, simply type in a corrected statement. Otherwise first type in a single "→". followed by carriage return.

It is also possible to encounter one of these comments because of failure to supply a CLASSPAK function with the proper data. In LIST SECT 0, if a carriage return is given when SECT asks for a section name, SECT will terminate with no value, and APL will object to the fact that LIST now has an undefined argument. These objections are harmless and can be ignored. Simply type in the next statement for CLASSPAK. In general, CLASSPAK comments terminate in periods, and CLASSPAK requests for input terminate in colons or question marks, while APL comments have no periods or colons and use the word ERROR.

E. INDEX ERROR or LENGTH ERROR.

These are APL comments which probably indicate that improper data has been undetected by the tests in the CLASSPAK subroutines. One way to cause this would be to type "LIST N, 2 3 0 -8 1.6 12345". Any of the last four elements would cause trouble. Such a condition inevitably leaves junk in the state indicator, and a "→" should always be typed after such a comment.

F. WSFULL

This APL comment means that even though there is enough room to store the final results of an operation, there is not enough room to store the intermediate results. In this case, the fact that machine conditions are saved may work to advantage. Several tricks may be tried to complete the operation.

(1) Type "R←C←0", then
"→⌶26" (⌶ is formed by overstriking ⊥ and ⊤)

(2))ERASE any spurious variables left around because of doing ordinary APL operations in the same workspace, then type "→⌶26". Do not erase DATA, any variables in DATA (See Part I.), or any variables with underscored characters in their names.

(3))ERASE a function not currently being used, such as ANALYZE, type "→⌶26" to complete the current operation, clean up the system as described in part G.

(4) Scratch the current operation by typing "→". Then clean up the stem as described in part G.

(5) If all else fails, do a ")SAVE (new wsname)", then note the responses to the inputs "⌶22" and "17 × MAX [1]." The former number is the number of bytes available in the workspace, and it should be larger than the latter number. Then ask for help.

G. Obtaining a copy of the CLASSPAK system.

At the University of Waterloo, both versions of CLASSPAK are stored in APL Library 3. The following sequence will give the user a copy of the system:

```
)LOAD 3  CLASSPAKi   (i = 1 or 2)
DESCRIBE             (this step may be omitted)
)ERASE DESCRIBE      (to avoid overly conservative initialization)
)SAVE                (user's wsname)
```

If one already has a copy of the system with data entered and wishes to obtain a cleaned up workspace and/or the latest version of the system, the following sequence will work:

```
)SAVE     (old wsname)
)LOAD     3 CLASSPAKi  (i = 1 or 2, same as before)
)COPY     (old wsname) DATA
SETCONST
)ERASE INITIALIZE SETCONST DESCRIBE
)WSID     (new wsname)
)SAVE     (new wsname)
```

After one has verified that no disasters have occurred, the old workspace can be dropped.

H. Specifications for FUNCT.

FUNCT must be a one-argument, one-output APL function. The input to FUNCT will be a vector of integers of length RC-1, given in the same order as the listing of argument columns by COMPUTE. The value ¯1 is assigned when marks are missing unless use has been made of the values ¯2 through ¯9. Any other parameters needed by FUNCT must have been entered into the workspace before COMPUTE is called. The output of FUNCT is a single number, which need not be an integer, as COMPUTE will perform the rounding.

I. The variables in DATA.

MAX maximum numbers of student names, sections, mark columns.

ORG 0, MAX [1], MAX [1] + MAX [2].

CNT largest index used for student names, sections, mark columns.

NW name-width (number of characters in a name or title).

NN alphabetic matrix of student names, section names, column titles.

SS boolean matrix of section bits.

MM integer matrix of marks (in version 2, this is actually an alphabetic matrix of characters, in which each character represents an integer from -9 through +100).

NB a boolean matrix in which column 1 represents the active rows, column 2 the active sections, column 3 the active mark columns.