
E M P L

USER'S

MANUAL

July 18, 1977

Erik T. Mueller
Britton House
Roosevelt, NJ 08555
(609) 448-2605

© 1977 by Erik T. Mueller

LOADING MML

1) Enter the following checksum loader into memory. (All locations are split octal.)

037	061	LXI SP
321	037	
322	037	
323	016	MOV 0,0
324	000	MVI A,020
325	076	
326	020	OUT 156
327	323	
330	156	LXI M,0
331	041	
332	000	
333	000	
334	000	
335	000	
336	000	
337	037	LOOP OLL INTP
338	037	
339	037	
340	037	
341	167	MOV M,A
342	201	ADD 0
343	117	MOV 0,A
344	043	INX H
345	174	MOV A,H
346	026	ORI 026
347	026	
350	302	JNZ LOOP
351	332	
352	332	
353	315	OLL INTP
354	306	
355	037	
356	271	CMP 0
357	302	JNZ CSER
358	357	
360	037	
361	037	
362	063	DI
363	303	GOOD JMP GOOD
364	303	
365	037	
366	156	INTP IN 156
367	156	
370	346	AMI 020
371	020	
372	302	JNZ INTP
373	306	
374	037	
375	333	IN 157
376	157	
377	311	HET

2) Put tape into cassette recorder.
 3) Load tape 037 3211: Run
 4) Start MML LED goes on. MML is being read into memory.
 5) When the address lights change to 037 367 and the INTR LED goes out, the read is complete. If the address lights instead indicate 037 377 with the INTR LED still on, there has been a checksum error and the user must try again.

Initializing MML as I/O system

- 1) Store the address of the first byte available for the workspace.
 025 260 LLL {Low byte of address} — 000
 025 261 HHH {High byte of address} — 052
 025 244 LLL — 052
 025 245 HHH — 052
 (Since MML resides from 000000 to 025377, the first available byte would normally be 026000 unless the user wishes to put I/O or other subroutines right after MML.)
- 2) Store the address of the first byte available for the workspace * 2.
 000 105 LLL — 002
 000 106 HHH — 052
 (This would normally be 026002.)
- 3) Store the address of the last byte available for the workspace.
 000 001 LLL — 377
 000 002 HHH — 157
 002 335 LLL — 377
 002 336 HHH — 157
 016 276 LLL — 377
 016 277 HHH — 137
 023 137 LLL — 377
 023 140 HHH — 157
 023 331 LLL — 377
 023 332 HHH — 157
 (In an 8-bit system, this should be 037 377 if I/O subroutines are right after MML or in a monitor somewhere else in memory.)
- 4) Store the address of your INPUT subroutine.
 000 011 LLL — 000
 000 012 HHH — 026
 Your INPUT subroutine should get a character from the terminal and put it in the accumulator (A). Bit 7 should be 1. The character should not be echoed, as MML will use your OUTPUT subroutine to echo it later.
 Be **frugal**! Every character that is changed.
 If you **resides** at 025 000, be changed.
- 5) Store your data input from keyboard port number.
 This is the address for control 0. (Program Interrupt)
 012 330 HHH — 001

- 6) Store the address of your OUTPUT subroutine.
 015 072 LLL — 026
 Your OUTPUT subroutine should print the character in 1 on your terminal. Bit 7 is 1. No registers in 1 on your terminal may be changed.
 When OR (215) is output, your subroutine should also output a LF (if necessary on your terminal.) Remember that in this case, a OR must be left in A upon exiting the subroutine.
 OUTPUT register at 026000.
 Store the address of your subroutine to clear the screen (if applicable.)
 014 333 LLL — 045
 014 334 HHH — 046
 If you have no such subroutine, store 014 333 000.
 334 000
- 8) Store the character which will cause your terminal to backspace (or the character that you wish to be echoed when Control B is typed.) This would be a Outser Left on a CRT terminal.
 015 260 iix — 377
- 9) Store the character which will cause your terminal to forward space without changing the character on the screen. (This will be echoed when a Control F is typed.) This would be a Outser Right on a CRT terminal, or simply a space (260) on a printing terminal.
 015 302 iix — 040 (space)
- 10) Store the address of your monitor or operating system. (used for the)QUIT command)
 025 263 LLL — 070
 025 264 HHH — 070
- 11) Make a new tape of MPL for future use.

MPL is now ready to run!

READING FILE

- 1) To enter MPL and create a clear workspace, start execution at 000 000.
- 2) To exit MPL, type)QUIT.
 To re-enter MPL leaving the workspace intact, start execution at 000070.
- 3) To save a workspace on tape:
 a) Type)QUIT. (Quit must be used to exit MPL, as it stores essential data in workspace for backup.)
 b) Dump memory onto tape starting at the location at 025 260 (L) 025 261 (H), and stepping at the location at 023 027 (L) 023 028 (H).
- 4) To load a workspace from tape:
 a) Load workspace back in memory from tape.
 b) Start execution at 000070.

Note: If the workspace is being loaded into a different part of memory than it was originally in, follow the following directions.

- a) Load workspace back into memory from tape.
- b) Store the new end of tape address at the location at 025 260 (L) 025 261 (H).
- c) Start execution at 000070.

4.1

Page Organization

07	000	061
001	377	LXI SP
002	037	
003	041	LXI M.O
004	000	
005	000	
006	076	MVI A,07H
007	074	
010	315	CALL DTP
011	046	
012	076	
013	078	MVI A,36H
014	315	
015	046	
016	046	CALL DTP
017	077	
019	016	
020	016	MVI O,O
021	000	
022	176	LD.
023	315	MOV A,M
024	046	CALL DTP
025	037	
026	201	ADD C
027	117	MOV O,A
030	043	INX M
031	174	MOV A,M
032	376	OPI 026
033	026	
034	302	JNC LP
035	022	
036	037	
037	171	MOV A,O
040	315	
041	043	CALL DTP
042	037	
043	301	SLEP.JMP SLEP
044	043	
045	037	
046	365	PDB
047	333	PTF.
048	156	IN 156
050	156	
051	346	AJI 040
052	040	
053	302	JNC VT
054	047	
055	037	
056	361	PDP PAB
057	323	OUT 157
060	157	
061	311	RST

[illegible]

```

0010 *****
0020 *      I/O FOR EMPL/N.S. DOS INTERFACE      *
0030 *      STEVE WILLIAMS 8/18/77 BYPA      *
0040 *****
0050 *      IN ADDITION TO THIS APPENDATION TO ERIK M
0060 *      EMPL, SEVERAL PATCHES HAVE BEEN MADE TO T
0070 *      ITSELF. THESE ARE DOCUMENTED IN THE EMPL
0080 *      SUPPLIED WITH YOUR COPY. THIS I/O PACKAGE
0090 *      PATCHES MENTIONED CONSTITUTE THE STANDARD
0100 *      PALO ALTO EMPL/NORTH STAR CONFIGURATION,
0110 *      INTENDED FOR THE SOLE USE OF THE CUSTOMER
0120      ORG      1600H      ;LIVES AT END OF EMPL.
0125 PSW      EQU      6      ;KLUDGE FOR SC5.
0130 DCHIN     EQU      2010H   ;DOS INPUT CHARACTER.
0140 DCHOUT    EQU      2000H   ;DOS OUTPUT CHARACTER.
0150 DOS      EQU      2008H   ;DOS RESTART.
0160 CR      EQU      00H
0170 LF      EQU      0AH
0175 BS      EQU      8
0180 *
0190 *      INPUT CHARACTER.
0200 *
0210 CHIN      CALL DCHIN      ;GET A CHARACTER FROM DOS.
0220          ORI      80H      ;SET HIGH BIT.
0230          RET              ;RETURN.
0240 *
0250 *      OUTPUT A CHAR
0260 *
0270 CHOUT     PUSH PSW          ;SAVE A
0280          PUSH B              ;SAVE B
0290          ANI      7FH       ;STRIP TOP BIT
0300          CPI      CR        ;CHAR A CR?
0310          CJZ      LFEEED    ; YES:DO A LINEFEED FIRST.
0312          CPI      7FH       ;USE THIS AS A BACKSPACE.
0314          JNZ      NOTES    ;TRANSLATE.
0316          MVI      A,BS
0320 NOTES     MOV      B,A      ;CHAR TO B FOR DOS.
0330          CALL DCHOUT        ;DO OUTPUT.
0340          POP      B          ;RESTORE REGS.
0350          POP      PSW
0360          RET
0370 LFEEED    MVI      A,LF      ;RETURN
0380          CALL CHOUT          ;GET A LINEFEED.
0390          MVI      A,CR      ;PRINT IT.
0400          RET              ;GET CR BACK.
0410 *
0420 *      AT-SIGN ,CRLF. INSTEAD OF CLR$CR.
0430 *
0440 CLR$CR    MVI      A,40H     ;AT-SIGN.
0450          CALL CHOUT
0460          MVI      A,CR      ;CRLF.
0470          JMP      CHOUT      ;PRINT AND RETURN FROM THER
9998 *      END

```

EMPL - an 8080 APL

Ballton House,
Rochester, NY 14655
(c) Copyright 1977 by Erik Mueller

8K EMPL is a micro APL for the Intel 8080. The current version fits in the first 5,632 bytes of memory. All the special symbols and operators of APL have been adapted to the ASCII character set.

The following description explains EMPL. Examples are used to clarify all operations.

EMPL has two modes: The normal Execution Mode in which all instructions are executed immediately, and the Definition Mode which permits the user to enter functions.

When in Execution Mode, the computer indents five spaces and waits for input.

When an arithmetic expression and EMPL will respond with the result.

2-5-3

-1 (Computer responses are underlined.)

All expressions are evaluated by the right-to-left rule.

That is, each operator takes as its right argument everything to its right. For example, in the above problem, EMPL first evaluates 2-3 which equals 3, then it evaluates 2-3 which

equals 3, then it evaluates 2-3 which equals 3.

The range is 132767--double-byte integer arithmetic is used.

Variable names may be any length of alphabetic characters.

They are assigned values with the "=" (typed as Control I.)

SPEED=2059

SPEED - 59

One dimensional arrays are called vectors in EMPL. A variable

may be specified to be a vector of any length provided sufficient

memory is available.

For example, if

5 -4 2 33

A=

then

6 -3 3 34

A+1

will result in

23 -2 4 0

A vector operator (such as "+", "-", "*", "/", etc.) takes two arguments,

one on the left and one on the right, as compared with a scalar

operator such as "A" (absolute value), or "J" (random), which

takes only one argument to its right.

Constants may be numeric or string vectors. Since the

internal representation of both is the same, operations may be

performed between the two types. To print a vector as a string,

type "J" before it.

X=STRING

01 24 22 73 78 71

STRING

1X+1

UNQUOTE

"A" is a description of some of EMPL's unique operators follows:

"A" is used to create a vector of consecutive integers between

1 and the right argument.

1 2 3 4 5

"A" gives the length of the right argument.

5

"A" returns the smaller of two numbers; "A" returns the greater.

-1 2

"A" gives the remainder when the right argument is divided

by the left.

2.5

"A" returns the condition of the right argument. If the condition is true, otherwise they return a 0.

0

"A" chains together two vectors.

1 0 1

HI=1 2 3 4

NI=19 512

XX=HI,NI

1 2 3 4 5 6

AI=GEN

BI=NATION

QATERNATION

18 2 40

18 2 10**4

72 8 40

18 2 40

18 2 10**4

18 2 40

18 2 10**4

18 2 40

18 2 10**4

18 2 40

18 2 10**4

18 2 40

18 2 10**4

18 2 40

18 2 10**4

18 2 40

18 2 10**4

18 2 40

18 2 10**4

18 2 40

18 2 10**4

18 2 40

18 2 10**4

"%" performs the Dyadic Scalar Operator % from right to left on all the elements of the right argument. Thus, it treats the left argument as a vector, type "SYMBOL".

To find the largest element in a vector, type "MAXIMUM".

APAS

To write a program (or function), the user must enter the program in the Definition Mode.

APAS

The computer now prompts with a "P" indicating that it is in Definition Mode.

Editing is now like a BASIC editor: type a line to insert it in the function; type the line number alone to delete the line; type an old line over to replace the old one.

Type "Q" to list the function; type "R" to renumber by a specified amount.

APAS

For example, the user types in a program to print the first 10 terms of Pascal's triangle.

APAS

For example, the user types in a program to print the first 10 terms of Pascal's triangle.

For example, the user types in a program to print the first 10 terms of Pascal's triangle.

For example, the user types in a program to print the first 10 terms of Pascal's triangle.

For example, the user types in a program to print the first 10 terms of Pascal's triangle.

APAS

APAS

Conditional branches are accomplished by the following:

If the condition `TEST` is true, the computer will branch to `LINE`. Otherwise, the computer will proceed to the next line.

Strings inside double quotes may be used to print comments.

APAS

Strings that contain spaces are used to separate items, and a semicolon at the end will suppress output.

Strings may be used to create a string as if it had been typed in apostrophe mode.

APAS

Specific elements of a vector may be assigned values.

APAS

APAS

APAS

When "Q" is assigned a value, the value is printed on the terminal.

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

APAS

```

ASK[-1]*
PROG [20] A
2210 P223

```

An error has occurred in line 10 of ASK. Since the execution of ASK hasn't been completed, the computer still has the return address saved. (line 20 of PROG) Now whenever a=10 instruction be typed, execution will continue at line 20 of PROG.

The program with the "*" to the right of it is the current suspended program. All "=" (branch) statements refer to this program. All program indicators that it is currently in execution mode. All programs that are suspended are pending execution. The program to return to next 16K of memory the list. It is a good idea to clear the State indicator often to prevent excess garbage from taking up memory. The State indicator is automatically cleared when entering Definition Mode, and after ERROR 5097 (workspace full.)

Here are some sample functions to illustrate EXRT's use.

```

(1) Generate a prime numbers:
[10] AP=PR END
[20] P1=1;A=1
[30] TEST: =10*ENK<AP
[40] ADD: =10*P+2;P=P+2
[50] =TEST
PR 10
2 3 5 7 11 13 17 19 23 29

```

```

(2) Compress a vector:
[10] AYS=STC COMP VCTR
[20] EX: AYS=AYS (VCTR/ID)*\STC[IDX
[30] =EK(IDX=IDX+1)*\AVCTR
4 1 0 1 1 0 COMP V5

```

```

(3) Sort a group of numbers:
[10] SORT=STC COMP V5
[20] LBI: =10*\A\US
[30] WHICH=US=5US
[40] ORD=ORD, WHICH COMP US
[50] US=(WHICH) COMP US
[60] =1B
5077 -2 1 -1 0 2
-2 -1 0 1 2

```

When FACT 3 is typed, a function called PAOF is searched for. When it is found, VL is set equal to 3 and the function is executed. Upon completion of the execution, the computer uses the variable specified to the left of the assignment operator as the result.

Here is an example of a defined dyadic function.

```

[10] RS=NO
[20] =20*\RS=RS, VTR)<SZ
[30] RS=RS\3Z

```

This function is used to restructure the vector on the right and that it has the length indicated on the left.

```

5 2 3 5 5 2 3

```

ARG to reenter Definition mode of an old function, just type its name.

```

ARGO

```

2 Parentheses are used to override the normal order of evaluation

```

(3-5)*7

```

5 The following commands may be used inside a program, or in Execution Mode.

```

)CLEAR - Clear the workspace.
)XNS - List the names of all functions in the workspace.
)STP - Return to execution mode.
)ERAS - Erase object. Erase the specified variable or function.
)SI - Display the State indicator.
)PUR - Clear the State indicator.
)QUIT - Return to user's monitor.
The State indicator is the stack of the return addresses of all programs whose execution is pending.
)Suppose a program is written, PROG, which calls for the execution of another one, ASK:
[10] ASK
[20] "OK"

```

```

)ASK
[0] AT THERE
PROG
ASK[10]HI THERE
ERROR 2810
The computer prints the program (or function) and line in which the error occurred.

```


HOW TO CONVERT ENPL TO RUB OR AN APL TERMINAL		
1) STORE	000 020	303
	021	117

UNILL IS THE ADDRESS OF YOUR OUTPUT SUBROUTINE
AS DESCRIBED IN A) ON PAGE 33

UNELL IS THE ADDRESS OF YOUR OUTPUT SUBROUTINE
AS DESCRIBED IN 6) ON PAGE 2.2.
THIS DISABLES THE CONVERSION ON OUTPUT OF CONTROL N.

2)	STORE	000	175	220
		001	352	224
		002	025	222
		003	245	227
		004	143	224
		010	314	226
		010	330	223
		013	057	222
		014	211	227
		014	253	227
		015	123	227
		016	334	220
		020	300	220
		021	343	226
		021	344	223
		021	345	225
		021	372	224

THIS ADAPTS ENPL TO NEW INTERNAL CODES DETAILED IN 3).

3) ENPL		A P L		ASSTL		YOU		ENPL		A P L		ASSTL		YOU	
1	1	241	1	337	1	241	1	275	0	243	0	275	0	243	0
2	2	246	6	254	6	246	6	283	6	254	6	283	6	254	6
3	3	277	7	277	7	277	7	333	7	277	7	333	7	277	7
4	4	336	6	336	6	336	6	324	6	336	6	324	6	336	6
5	5	253	3	253	3	253	3	211	3	253	3	211	3	253	3
6	6	255	5	255	5	255	5	284	5	255	5	284	5	255	5
7	7	252	2	252	2	252	2	285	2	252	2	285	2	252	2
8	8	257	7	257	7	257	7	210	7	257	7	210	7	257	7
9	9	236	6	236	6	236	6	250	6	236	6	250	6	236	6
10	10	223	3	223	3	223	3	251	3	223	3	251	3	223	3
11	11	225	5	225	5	225	5	300	5	225	5	300	5	225	5
12	12	274	4	274	4	274	4	228	4	274	4	228	4	274	4
13	13	231	1	231	1	231	1	247	1	231	1	247	1	231	1
14	14	216	6	216	6	216	6	227	6	216	6	227	6	216	6
15	15	276	6	276	6	276	6	220	6	276	6	220	6	276	6

ENPL \ 0 0 0 0 LANGUAGE SUMMARY

(Notes: ENPL can be run either in ASCII or the
APL character set.)

APL	ASCII	ENPL Category
$\overline{Y}$	Y	Monadic Scalar Operators
$\overline{Y}$	Y	Minus Y
$\overline{Y}$	Y	The absolute value of Y
$\overline{Y}$	Y	Not Y
$\overline{Y}$	Y	Random from 1 to Y
$\overline{Y}$	Y	Monadic Mixed Operators
$\overline{Y}$	Y	Vector of consecutive integers from 1 to Y
$\overline{Y}$	Y	The length of Y
$\overline{Y}$	Y	Dyadic Scalar Operators
$\overline{Y}$	Y	X plus Y; X or Y
$\overline{Y}$	Y	X minus Y
$\overline{Y}$	Y	X times Y; X and Y
$\overline{Y}$	Y	X divided by Y
$\overline{Y}$	Y	Minimum of X and Y
$\overline{Y}$	Y	Maximum of X and Y
$\overline{Y}$	Y	Remainder of X/Y
$\overline{Y}$	Y	X less than Y
$\overline{Y}$	Y	X less than or equal to Y
$\overline{Y}$	Y	X greater than Y
$\overline{Y}$	Y	X greater than or equal to Y
$\overline{Y}$	Y	X equal to Y
$\overline{Y}$	Y	X not equal to Y
$\overline{Y}$	Y	Dyadic Mixed Operators
$\overline{Y}$	Y	Y catenated to X
$\overline{Y}$	Y	Elements of X at locations Y
$\overline{Y}$	Y	Composite Operators
$\overline{Y}$	Y	The Operation f performed from right to left on all the elements of Y
$\overline{Y}$	Y	Special Operators and characters
$\overline{Y}$	Y	Assign Y to X
$\overline{Y}$	Y	Print X as a string
$\overline{Y}$	Y	Print literal text
$\overline{Y}$	Y	Execute a string
$\overline{Y}$	Y	Branch to X
$\overline{Y}$	Y	Indexed variable assignment
$\overline{Y}$	Y	Numeric Input/Output
$\overline{Y}$	Y	String Input/Output
$\overline{Y}$	Y	String vector
$\overline{Y}$	Y	Expression in parentheses
$\overline{Y}$	Y	Constant negation

APL	ASCII	ENPL Category
$\overline{Y}$	Y	Definition Mode Commands
$\overline{Y}$	Y	List function
$\overline{Y}$	Y	Number
$\overline{Y}$	Y	Return to execution mode
$\overline{Y}$	Y	Function Definitions
$\overline{Y}$	Y	Monadic function; no result
$\overline{Y}$	Y	Monadic function with result
$\overline{Y}$	Y	Dyadic function with result
$\overline{Y}$	Y	Commands
$\overline{Y}$	Y	Clear workspace
$\overline{Y}$	Y	Display function names
$\overline{Y}$	Y	Display variable names
$\overline{Y}$	Y	Clear state indicator
$\overline{Y}$	Y	Erase function or variable
$\overline{Y}$	Y	Clear screen
$\overline{Y}$	Y	Display state indicator
$\overline{Y}$	Y	Return to execution mode
$\overline{Y}$	Y	Return to user's monitor
$\overline{Y}$	Y	Special Control Characters
$\overline{Y}$	Y	Control B Backspace
$\overline{Y}$	Y	Control C Stop
$\overline{Y}$	Y	Control F Forward space
$\overline{Y}$	Y	Control M " "
$\overline{Y}$	Y	Control I " "
$\overline{Y}$	Y	Control M " "
$\overline{Y}$	Y	Control P " "
$\overline{Y}$	Y	Control S " "
$\overline{Y}$	Y	Control T " "
$\overline{Y}$	Y	Control U " "
$\overline{Y}$	Y	Control V " "
$\overline{Y}$	Y	Control M " "
$\overline{Y}$	Y	Control X Cancel line
$\overline{Y}$	Y	Control Y " "
$\overline{Y}$	Y	Control N " "

ENPL Range = 13767

COMPILED SAMPLE EXECUTIONS AT APL CHARACTER SET

```

(1) GENERATE 4 PRIME NUMBERS:
VP-PR END
(1) P-1-T-1
(2) TEST:~0~ENDSP
(3) ADD:~ADD~1~0~P-T-T-2
(4) P-P-T
(5) ~TEST
V
PM 10
2 3 5 7 11 13 17 19 23 29
(2) COMPRESS A VECTOR:
VARS-SICT COMP VCTR
(1) IDI-1-ABS-10
(2) BR:ANS-ANS. (VCTR(IDI)~1~SICT(IDI
(3) ~&~(IDI-IDI+1)~&VCTR
V
1 0 1 1 0 COMP 15
1 3 ~
(3) SORT A GROUP OF NUMBERS:
(THIS FUNCTION USES COMP)
WORD-SORT UNS
(1) ORD-10
(2) LB:~0~1~P-UNS
(3) WHICH+UNS=L/UNC
(4) ORD-ORD, WHICH COMP UNS
(5) UNS-~(WHICH) COMP UNS
(6) ~LB
V
SORT ~2 1 ~1 0 2
~1 0 1 2
A SORT ~SORT~
OHCT
(4) REVERSE A VECTOR:
VAV-REV VC
(1) RV-VC((1+VC)-1)VC
V
JREV~I THERE~
FAHRT~I~
V
REV 15
5 4 3 2 1
(3) MAKE THE LEFT ARGUMENT TO THE RIGHT ARGUMENT.
(THIS MUST BE SINGLE NUMBERS.)
VEX-SE RAIS BUN
(1) EX++/SE+~1~BUN
V
2 RAIS 5
32

```

SAMPLE DIALOG IN EMTL RUNNING APL CHARACTER SET:

```

EMTL 1.0
CLEAR US
V/110
55
P/1~123
BCDFGHIJKLMNOPQRSTUVWXYZ
VAV-REV VC
~10 RV-VC((1+VC)-1)VC
VD
VAV-REV VC
(10) RV-VC((1+VC)-1)VC
V
F REV ~TESTING~
CHITSET
AVC
TESTING
JST
RVV~1~
2015 FREE
VPASCAL
P-1
VPASCAL
ERROR 3027
~10 P-1
~20 ~PRINT~#20P~(0,P)~P.0
~15 PRINT:P
~SIC
VD
VPASCAL
(10) P-1
(20) PRINT:P
(30) ~PRINT~#20P~(0,P)~P.0
V
N+8
PASCAL
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1.
)FMS
MTV:PASCAL:
TEST~1~3000
EMMR 5097
TEST
EMMR 2010

```


276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	784	785	786
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

[illegible]

E M P L

USER'S

MANUAL

July 18, 1977

Erik T. Mueller
Britton House
Roosevelt, NJ 08555
(800) 448-2605

© 1977 by Erik T. Mueller

MEMORY MAP

1) Enter the following checksum loader into memory.
(All locations are opti total.)

037	061	LXI SP
321	320	
322	320	
323	016	MOV 0,0
324	000	
325	000	MTI A,020
326	076	
327	020	OUT 156
330	323	
331	156	
332	041	LXI M,0
333	000	
334	000	
335	315	LOOP CALL INTP
336	345	
337	037	
338	373	
339	167	HI
340	201	MOV M,A
341	117	ADD 0
342	043	MOV 0,A
343	043	INX H
344	174	MOV A,H
345	376	CPI 026
346	026	
347	302	JNZ LOOP
350	302	
351	335	
352	315	CALL INTP
353	315	
354	306	
355	037	CMP 0
356	271	JNZ CSN
357	302	CSN
360	357	
361	037	BI
362	363	GOOD JMP GOOD
363	303	GOOD
364	363	
365	037	INTP
366	156	
367	156	
370	346	ADI 020
371	020	
372	302	JNZ INTP
373	366	
374	037	
375	333	IN 157
376	157	
377	311	HIT

- 2) Put tape into cassette recorder.
- 3) Start tape 037 321; Run
- 4) Start tape INTP LED goes on. MPFL is having read into memory.
- 5) When the address lights change to 037 367 and the INTP LED goes out, the read is complete. If the address lights instead indicate 037 377 with the INTP LED still on, there has been a checksum error and the user must try again.

Input/Output MPFL and MPFL SYSTEM

- 1) Store the address of the first byte available for the workspace.
025 260 LIL {Low byte of address} — 000
025 261 HIL {High byte of address} — 052
025 244 LIL — 000
025 245 HIL — 052
(Since MPFL resides from 000000 to 023377, the first available byte would normally be 026000 unless the user wishes to put I/O or other subroutines right after MPFL.)
- 2) Store the address of the first byte available for the workspace + 2.
000 002 HIL — 052
000 102 HIL — 052
(This would normally be 026002.)
- 3) Store the address of the last byte available for the workspace.
000 001 LIL — 377
000 002 HIL — 137
002 335 LIL — 377
002 336 HIL — 137
016 276 LIL — 377
016 277 HIL — 137
023 137 LIL — 377
023 140 HIL — 137
023 331 LIL — 377
023 332 HIL — 137
(In an 8K system, this would be 037 377 if I/O subroutines are right after MPFL or in a monitor somewhere else in memory.)
- 4) Store the address of your INPUT subroutine.
000 011 LIL — 000
000 012 HIL — 026
Your INPUT subroutine should get a character from the terminal and put it in the Accumulator (A). Bit 7 should be 1.
The character should not be echoed, as MPFL will use your OUTPUT subroutine to echo it later.
No registers other than A may be changed.
INPUT resides at 026000.
- 5) Store your data input from keyboard port number.
This is used to check for control 0. (Program Interrupt)
012 330 LIL — 001

- 6) Store the address of your OUTPUT subroutine.
 015 072 LIL — 046
 015 073 HHH — 026
 Your OUTPUT subroutine should print the character in A on your terminal. Bit 7 is 1. No registers including A may be changed.
 When a CR (215) is output, your subroutine should also output a LF (14) necessary on your terminal.)
 Whenever you use the CR or LF, CR must be left in A and the LF must be left in the subregister of the subregister.
 OUTPUT register at 026006
 015 074 LIL — 045
 014 334 HHH — 026
 If you have no such subroutine, store 014 333 060.
 334 000
- 7) Store the address of your subroutine to clear the screen (if applicable).
 014 333 LIL — 045
 014 334 HHH — 026
 If you have no such subroutine, store 014 333 060.
 334 000
- 8) Store the character which will cause your terminal to backspace (or the character that you wish to be echoed when Control A is typed.)
 015 260 LIL — 045
 015 261 HHH — 026
 This would be a CR (215) on a CRT terminal.
 334 000
- 9) Store the character which will cause your terminal to forward space without changing the character on the screen. (This will be echoed when a Control F is typed.)
 This would be a CR (215) on a CRT terminal, or simply a space (240) on a printing terminal.
 015 262 LIL — 045
 015 263 HHH — 026
- 10) Store the address of your monitor or operating system.
 (This is the address of the monitor or operating system.)
 025 263 LIL — 045
 025 264 HHH — 026
- 11) Make a new tape of MPL for future use.

MPL is now ready to run!

APPENDIX 9 MPL

- 1) To enter MPL and create a clear workspace, start execution at 000 000.
- 2) To exit MPL, type)QUIT.
 To re-enter MPL leaving the workspace intact, start execution at 000 000.
- 3) To save a workspace on tape:
 a) Type)QUIT. (Quit must be used to exit MPL, as it stores essential data in workspace for taping.)
 b) Dump memory onto tape starting at the location at 025 260 (L) 025 261 (H), and stepping at the location at 023 027 (L) 023 030 (H).
- 4) To load a workspace from tape:
 a) Load workspace back in memory from tape.
 b) Start execution at 000 000.

Note: If the workspace is being loaded into a different part of memory than it was originally in, follow the following directions:

- a) Load workspace back into memory from tape.
- b) Store the new set of tape address at the location at 025 260 (L) 025 261 (H).
- c) Start execution at 000 000.

2.6

037 000 061 LLI 8F
001 377

037	000	061	LI1 82
001	377		
002	037		
003	041		LI1 N.O.
004	000		
004	000		
004	007		MT1 A.O7
004	076		
007	374		
011	046		0AL 07P
012	037		
013	076		MT1 A.34
014	346		
015	315		0AL 07P
016	046		
017	037		
020	016		MT1 0.0
021	000		
022	176		MOV A.N
023	046		0AL 07P
024	046		
025	037		
026	201		AND 0.0
027	117		MOV 0.A
030	043		IRX N
031	174		MOV A.N
032	376		07P 026
033	026		
034	002		
035	022		JR2 LP
036	171		
037	171		MOV A.O
040	315		0AL 07P
041	046		
042	037		
043	303		RELJ.JMP RELJ
044	043		
045	037		
046	365		PUSH PSW
047	313		IR 156
048	046		
049	046		
050	046		
051	046		
052	040		
053	302		JR2 WT
054	047		
055	037		
056	361		POP PSW
057	323		0UT 157
060	157		
061	311		RET

...

IF they meet, US WILL error.

Stacked data

```

K00 = 010 Program
    011 Dyad10 function
    012 Menad10 function
    004 Variable

```

```

0010 *****
1600 0020 * I/O FOR EMPL/N.S. DOS INTERFACE *
1600 0030 * STEVE WILLIAMS 8/16/77 BYPA *
1600 0040 *****
1600 0050 * IN ADDITION TO THIS APPENDIX TO ERIK M
UELLER'S
1600 0060 * EMPL, SEVERAL PATCHES HAVE BEEN MADE TO
HE EMPL
1600 0070 * ITSELF. THESE ARE DOCUMENTED IN THE EMPL
MANUAL
1600 0080 * SUPPLIED WITH YOUR COPY. THIS I/O PACKAGE
, AND THE
1600 0090 * PATCHES MENTIONED CONSTITUTE THE STANDARD
BYTE OF
1600 0100 * PALO ALTO EMPL/NORTH STAR CONFIGURATION,
AND ARE
1600 0110 * INTENDED FOR THE SOLE USE OF THE CUSTOMER
S WE SUPPLY.
1600 0120 ORG 1600H ;LIVES AT END OF EMPL.
1600 0125 PSW EQU 5 ;KLUDGE FOR SCs.
1600 0130 DCHIN EQU 2010H ;DOS INPUT CHARACTER.
1600 0140 DCHOUT EQU 200DH ;DOS OUTPUT CHARACTER.
1600 0150 DOS EQU 202SH ;DOS RESTART.
1600 0160 CR EQU 0DH
1600 0170 LF EQU 0AH
1600 0175 BS EQU 5
1600 0180 *
1600 0190 * INPUT CHARACTER.
1600 0200 *
1600 CD 10 20 0210 CHIN CALL DCHIN ;GET A CHARACTER FROM DOS.
1600 F6 80 0220 ORI 80H ;SET HIGH BIT.
1600 C9 0230 RET ;RETURN.
1600 0240 *
1600 0250 * OUTPUT A CHAR
1600 0260 *
1600 0270 CHOUT PUSH PSW ;SAVE A
1600 0280 PUSH B ;SAVE B
1600 0290 ANI 7FH ;STRIP TOP BIT
1600 FE 0D 0300 CPI CR ;CHAR A CR?
1600 CC 1D 16 0310 CZ LFEED ; YES-DO A LINEFEED FIRST.
1600 FE 7F 0312 CPI 7FH ;USE THIS AS A BACKSPACE.
1611 C2 16 16 0314 JNZ NOTBS ;TRANSLATE.
1614 3E 08 0316 MVI A,BS
1616 47 0320 NOTBS MOV B,A ;CHAR TO B FOR DOS.
1617 CD 0D 20 0330 CALL DCHOUT ;DO OUTPUT.
161A C1 0340 POP B ;RESTORE REGS.
161B F1 0350 POP PSW
161C C9 0360 RET ;RETURN
161D 3E 0A 0370 LFEED MVI A,LF ;GET A LINEFEED.
161F CD 06 16 0380 CALL CHOUT ;PRINT IT.
1622 3E 0D 0390 MVI A,CR ;GET CR BACK.
1624 C9 0400 RET
1625 0410 *
1625 0420 * AT-SIGN ,CRLF. INSTEAD OF CLRSCR.
1625 0430 *
1625 3E 40 0440 CLR$CR MVI A,40H ;AT-SIGN.
1627 CD 06 16 0450 CALL CHOUT
162A 3E 0D 0460 MVI A,CR ;CRLF.
162C C3 06 16 0470 JMP CHOUT ;PRINT AND RETURN FROM THEM.
E.
162F 9998 * END

```

E.M.P.L. on 8080 AFL

Written House
Roosevelt, N.J. 08555
(c) Copyright 1977 by Erik Mueller

8K EMPL is a micro AFL for the Intel 8080. The current version fits in the first 5,632 bytes of memory. All the special symbols and operators of AFL have been adapted to the ASCII character set.

The following description explains EMPL. Examples are used to clarify all operations.

EMPL has two modes: The normal Execution Mode in which all instructions are executed immediately, and the Definition Mode which permits the user to enter functions. When in Execution Mode, the computer indents five spaces and waits for input. When in Definition Mode, the computer will respond with the prompt:

-1

(Computer responses are underlined.)

All expressions are evaluated by the right-to-left rule. That is, each operator takes as its right argument everything to its right. For example, in the above problem, EMPL first evaluates 2-3 which equals 5, then it evaluates 2-5 which equals 7. The range is 12767--double-byte integer arithmetic is used. Variable names may be any length of alphabetic characters. They are assigned values with the "=" (typed as Control I.)

SP2521=2059

SP2521=59

2000 One dimensional arrays are called vectors in EMPL. A variable may be represented by a vector of any length provided sufficient memory is available.

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

A1=5 -4 2 33

STRING
X+1
X+1

A description of some of EMPL's unique operators follows: "X" is used to create a vector of consecutive integers between 1 and the right argument.

1 2 3 4 5
X 5
X HELLO
X HELLO

2 3

23 returns the smaller of two numbers; "X" returns the greater.

1 2

4 1 9 3 2 56

2.5 divides the remainder when the right argument is divided by the left.

1 2

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

0 7 2 4 3

chains together two vectors.

1 2 3 4

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

HELLO 1 2 3 4

1 2 3 4 5 6

"1" performs the Dyadic Scalar Operator f from right to left on all the elements of the right argument. Thus, it treats $4*5$ as if it were $1+2+3+4+5$.

To find the largest element in a vector, type $\%MAXOR$.

To write a program (or function), the user must enter EXPR's Definition Node. Type "d" followed by the new name of the function.

2225

The computer now prompts with a "y" indicating that it is in Definition Mode, like a BASIC editor. Type a line to insert it in the function; type the line number alone to delete the line; type an old line over to replace the old one. Type "q" to list the function; type "q" to renumber by a specified amount.

For example, the user types in a program to print the first row of Pascal's triangle.

```
200 P=1
210 P=P+1
215 PRINT P
22 10
```

2225

```
[10] P=1
[20] PRINT P
[30] P=PRINTD=P+(O,P)+P,O
```

Note that "P=" can be used inside any expression.

"PRINT" is used as a label in line 20. Upon exiting Definition Mode, PRINT becomes a variable whose value is 20.

which follows it. Branch to the line given by the expression which follows it. Branch to a null vector is no branch (computer prints the next line), and branch to an undefined line is the end of program.

The user now types a "q" to leave Definition Mode.

22

```
P=5
225
```

Conditional branches are accomplished by the following:

If the condition $\%IF$ is true, the computer will branch to LINE. Otherwise, the computer will proceed to the next line.

Strings inside double quotes may be used to print comments.

```
2225 "q"
```

Strings that semicolons are used to separate items, and a semicolon at the end will suppress CR/LF.

"q" is used to execute a string as if it had been typed in execution mode.

```
2225 PRINT "q"
```

```
2225
```

5. 10. 15. 20. 25

```
A=5
A(3)=17 18
```

```
1. 2. 17 18 5 6
```

When "q" is assigned a value, the value is printed on the terminal.

```
A=5 "q"
```

Then "q" is assigned a value, the value is printed as a string on the terminal.

```
PRINT "q"
```

if "q" is used as an operand in an expression, the computer first requests input from the terminal. "q" enables string input to be used. (The input is taken as a literal.)

```
2225
```

```
[10] WHAT IS YOUR NAME?
```

```
[20] NAME=NAME
```

```
[30] PRINT AN EXPRESSION
```

```
[40] PRINT "q"
```

```
[50] PRINT "q"
```

```
[60] THE ANSWER IS: EXTRA
```

```
2225
```

```
WHAT IS YOUR NAME?
NAME=NAME
PRINT AN EXPRESSION
PRINT "q"
```

```
THE ANSWER IS: 25
```

```
2225
```

```
User defined monadic or dyadic functions are possible.
```

Here is an example of a user defined monadic function to take the factorial of the right argument.

```
[10] P=1
```

```
"FACT" may now be used as any EXPR monadic operator can be used.
```

```
2225
```


		Z80L Range = 72A-767
	<u>Commands</u>	
Clear	Clear Workspace	
PUS	Display function names	
VARS	Display variable names	
FIR	Display state indicator	
CLASS ob	Class appears on variable.	
SR	Class appears	
STOP	Display State Indicator	
	Return to execution Mode	
	Return to monitor	
	QUIT	
	<u>Special Control Characters</u>	
Control G	ScreenSpace	
Control S	Stop	
Control P	Forward space	
Control H	=:	
Control I	:=	
Control M	>=	
Control X	Cancel line	
	<u>Internal Messages</u>	
AIE	String is too large to execute	
B002	Object does not exist	
B003	Object does not have type in expression	
1269	Length of vectors doesn't match	
1466	Illegal Operation in reduction	
2495	Variable not present in indexing	
2539	Length of index doesn't match length of value to be assigned; or variable not declared as variable	
2550	Invalid assignment operand type	
2630	Function not found	
2810	Variable or function not found	
3593	Backspace less than beginning of line buffer	
3581	Forward spacing greater than end of line buffer	
3742	Input over 256 function definition	
3827	Illegal command in definition mode	
4058	Illegal redefinition of monadic function header	
4063	Illegal redefinition of dyadic function header	
5097	Nonspase kill	
5092	Division by zero	

NEW INTERNAL CODE

YOUR OUTPUT ROUTINE SHOULD USE A CONVERSION TABLE TO CONVERT THE LISTED CHARACTER CODE IN 3) TO THE CORRECT CODE FOR YOUR TERMINAL.

NEW I. B. L. ERROR MESSAGES

412 STRING IS TOO LARGE TO EXECUTE
 885 OBJECT ALREADY ERASED
 1082 UNKNOWN OPERAND TYPE IN EXPRESSION
 1466 LENGTH OF VECTORS DOESN'T MATCH
 2266 ILLEGAL OPERATION IN REDUCTION
 2495 VARIABLE NOT PRESENT IN INDEXING
 2529 LENGTH OF INDEX DOESN'T MATCH LENGTH OF VALUE TO BE ASSIGNED
 3534 INDEX GREATER THAN LENGTH OF VARIABLE
 3636 UNKNOWN ASSIGNMENT OPERAND TYPE
 2690 FUNCTION NOT FOUND
 2810 VARIABLE OR FUNCTION NOT FOUND
 3503 BACKSPACE LESS THAN BEGINNING OF LINE BUFFER
 3521 FORWARD SPACING GREATER THAN END OF LINE BUFFER
 3542 INPUT OVER 12 CHARACTERS
 3737 BAD STATE IN FUNCTION DEFINITION
 3827 ILLEGAL COMMAND IN DEFINITION MODE
 4058 ILLEGAL REDISTRIBUTION OF MODARIC FUNCTION HEADER
 4083 ILLEGAL REDISTRIBUTION OF DIADIC FUNCTION HEADER
 5097 WORKSPACE FULL
 5302 DIVISION BY ZERO

HOW TO CONVERT ENPL TO RUB OR AN APL TERMINAL

1) STORE 000 020 263

021 LEL

022 REN

RENEL IS THE ADDRESS OF YOUR OUTPUT SUBROUTINE AS DESCRIBED IN 6) ON PAGE 3.2. THIS DISABLES THE CONVERSION ON OUTPUT OF CONTROL N, I. B. AND Y TO "...", ">", "<".

2) STORE 000 175 220

001 352 224

004 025 222

006 143 226

010 314 226

010 330 222

012 057 222

014 211 227

014 253 227

015 123 227

016 334 220

020 300 220

021 343 226

021 344 223

021 345 225

021 372 224

THIS ADAPTS ENPL TO NEW INTERNAL CODES DETAILED IN 3).

3) ENPL			ENPL			YOU			ENPL			ENPL			YOU		
A P L			ASCII			CODE TYPE			A P L			ASCII			CODE TYPE		
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	
5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	
6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	
8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	
9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	
10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	
11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	
12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	
13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	
14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	
16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	
17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	
18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	
19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	
20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	
22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	
23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	
24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	
25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	
26	26	26	26	26	26	26	26	26	26	26	26	26	26	26	26	26	
27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	
28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	
29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	
30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	
31	31	31	31	31	31	31	31	31	31	31	31	31	31	31	31	31	
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	
33	33	33	33	33	33	33	33	33	33	33	33	33	33	33	33	33	
34	34	34	34	34	34	34	34	34	34	34	34	34	34	34	34	34	
35	35	35	35	35	35	35	35	35	35	35	35	35	35	35	35	35	
36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	
37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	
38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	
39	39	39	39	39	39	39	39	39	39	39	39	39	39	39	39	39	
40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	
41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	
42	42	42	42	42	42	42	42	42	42	42	42	42	42	42	42	42	
43	43	43	43	43	43	43	43	43	43	43	43	43	43	43	43	43	
44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	
45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	
46	46	46	46	46	46	46	46	46	46	46	46	46	46	46	46	46	
47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	
48	48	48	48	48	48	48	48	48	48	48	48	48	48	48	48	48	
49	49	49	49	49	49	49	49	49	49	49	49	49	49	49	49	49	
50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	
51	51	51	51	51	51	51	51	51	51	51	51	51	51	51	51	51	
52	52	52	52	52	52	52	52	52	52	52	52	52	52	52	52	52	
53	53	53	53	53	53	53	53	53	53	53	53	53	53	53	53	53	
54	54	54	54	54	54	54	54	54	54	54	54	54	54	54	54	54	
55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	
56	56	56	56	56	56	56	56	56	56	56	56	56	56	56	56	56	
57	57	57	57	57	57	57	57	57	57	57	57	57	57	57	57	57	
58	58	58	58	58	58	58	58	58	58	58	58	58	58	58	58	58	
59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	
60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	
61	61	61	61	61	61	61	61	61	61	61	61	61	61	61	61	61	
62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	
63	63	63	63	63	63	63	63	63	63	63	63	63	63	63	63	63	
64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	
65	65	65	65	65	65	65	65	65	65	65	65	65	65	65	65	65	
66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	
67	67	67	67	67	67	67	67	67	67	67	67	67	67	67	67	67	
68	68	68	68	68	68	68	68	68	68	68	68	68	68	68	68	68	
69	69	69	69	69	69	69	69	69	69	69	69	69	69	69	69	69	
70	70	70	70	70	70	70	70	70	70	70	70	70	70	70	70	70	
71	71	71	71	71	71	71	71	71	71	71	71	71	71	71	71	71	
72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	
73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	
74	74	74	74	74	74	74	74	74	74	74	74	74	74	74	74	74	
75	75	75	75	75	75	75	75	75	75	75	75	75	75	75	75	75	
76	76	76	76	76	76	76	76	76	76	76	76	76	76	76	76	76	
77	77	77	77	77	77	77	77	77	77	77	77	77	77	77	77	77	
78	78	78	78	78	78	78	78	78	78	78	78	78	78	78	78	78	
79	79	79	79	79	79	79	79	79	79	79	79	79	79	79	79	79	
80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	
81	81	81	81	81	81	81	81	81	81	81	81	81	81	81	81	81	
82	82	82	82	82	82	82	82	82	82	82	82	82	82	82	82	82	
83	83	83	83	83	83	83	83	83	83	83	83	83	83	83	83	83	
84	84	84	84	84	84	84	84	84	84	84	84	84	84	84	84	84	
85	85	85	85	85	85	85	85	85	85	85	85	85	85	85	85	85	
86	86	86	86	86	86	86	86	86	86	86	86	86	86	86	86	86	
87	87	87	87	87	87	87	87	87	87	87	87	87	87	87	87	87	
88	88	88	88	88	88	88	88	88	88	88	88	88	88	88	88	88	
89	89	89	89	89	89	89	89	89	89	89	89	89	89	89	89	89	
90	90	90	90	90	90	90	90	90	90	90	90	90	90	90	90	90	
91	91	91	91	91	91	91	91	91	91	91	91	91	91	91	91	91	
92	92	92	92	92	92	92	92	92	92	92	92	92	92	92	92	92	
93	93	93	93	93	93	93	93	93	93	93	93	93	93	93	93	93	
94	94	94	94	94	94	94	94	94	94	94	94	94	94	94	94	94	
95	95	95	95	95	95	95	95	95	95	95	95	95	95	95	95	95	
96	96	96	96	96	96	96	96	96	96	96	96	96	96	96	96	96	
97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	
98	98	98	98	98	98	98	98	98	98	98	98	98	98	98	98	98	
99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	
100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	
101	101	101	101	101	101	101	101	101	101	101							

(NOTE: DAPL can be run either in ASCII or the
APL character set.)

<u>APL</u>	<u>SQL</u>	<u>SQL Category</u>
<u>-</u>	<u>Monadic Scalar Operators</u>	
<u>~</u>	<u>Minus Y</u>	
<u> Y</u>	<u>The absolute value of Y</u>	
<u>~Y</u>	<u>Not Y</u>	
<u>Y</u>	<u>Random from 1 to Y</u>	
<u>Y</u>	<u>Monadic Mixed Operators</u>	
<u>Y</u>	<u>Vector of consecutive integers from 1 to Y</u>	
<u>Y</u>	<u>The length of Y</u>	
<u>X Y</u>	<u>Diadic Scalar Operators</u>	
<u>X Y</u>	<u>X plus Y; X or Y</u>	
<u>X Y</u>	<u>X minus Y</u>	
<u>X Y</u>	<u>X times Y; X and Y</u>	
<u>X Y</u>	<u>X divided by Y</u>	
<u>X Y</u>	<u>Minimum of X and Y</u>	
<u>X Y</u>	<u>Maximum of X and Y</u>	
<u>X Y</u>	<u>Remainder of X by Y</u>	
<u>X Y</u>	<u>X less than Y</u>	
<u>X Y</u>	<u>X less than or equal to Y</u>	
<u>X Y</u>	<u>X greater than Y</u>	
<u>X Y</u>	<u>X greater than or equal to Y</u>	
<u>X Y</u>	<u>X equal to Y</u>	
<u>X Y</u>	<u>X not equal to Y</u>	
<u>X Y</u>	<u>Diadic Mixed Operators</u>	
<u>X Y</u>	<u>Y catenated to X</u>	
<u>X Y</u>	<u>Elements of X at locations Y</u>	
<u>X Y</u>	<u>Composite Operators</u>	
<u>X Y</u>	<u>The Operation X performed from right to left on all the elements of Y</u>	
<u>X Y</u>	<u>Special Operators and Characters</u>	
<u>X Y</u>	<u>Assign Y to X</u>	
<u>X</u>	<u>Print X as a string</u>	
<u>"TEXT"</u>	<u>Print literal text</u>	
<u>X</u>	<u>Execute a string</u>	
<u>X</u>	<u>Branch to X</u>	
<u>X</u>	<u>Indexed variable assignment</u>	
<u>X</u>	<u>Numeric Input/Output</u>	
<u>X</u>	<u>String Input/Output</u>	
<u>X</u>	<u>String Vector</u>	
<u>X</u>	<u>Expression in parentheses</u>	
<u>X</u>	<u>Constant negation</u>	

<u>APP</u>	<u>ASCII</u>	<u>HEX</u>	<u>DEFINITION</u>
<u>Q</u>	Q n		Definition Mode Commands
<u>R</u>	R n		List function
<u>S</u>	S n		Number
<u>V</u>	V		Return to execution mode
			Function Definitions
<u>W</u>	W		Wildcard functions with no result
<u>W-R</u>	W-R f b		Wildcard functions with result
<u>W-R-F</u>	W-R-F f b		Dyadic function with result
			Commands
<u>CEAR</u>	CEAR		Clear Workspace
<u>FNS</u>	FNS		Display function names
<u>VARS</u>	VARS		Display variable names
<u>PUR</u>	PUR		Clear flags indicator
<u>ER-OB</u>	ER-OB		Erase function or variable
<u>SCN</u>	SCN		Clear Screen
<u>SI</u>	SI		Display Status Indicator
<u>STOP</u>	STOP		Return to execution mode
<u>QUIT</u>	QUIT		Return to user's monitor
			Special Control characters
<u>B</u>	B		Backspace
<u>C</u>	C		Stop
<u>F</u>	F		Forward space
<u>T</u>	T		
<u>P</u>	P		
<u>V</u>	V		
<u>F</u>	F		
<u>T</u>	T		
<u>V</u>	V		
<u>L</u>	L		
<u>C</u>	C		
<u>X</u>	X		Cancel line
<u>E</u>	E		
<u>T</u>	T		
<u>B</u>	B		

HP-1, Range = 133767

CORRECTED SAMPLE INSTRUCTIONS IN APL CHARACTER SET

(1) GENERATE A PRIME NUMBERS:

```

PR 10
(1) P-1-T-1
(2) TEST:O-1-PRD-OP
(3) ADD:V-1-OP-1-T-2
(4) P-2-T
(5) TEST

```

(2) COMPRESS A VECTOR:

```

(1) ID-1-ANS-1-0
(2) BK:ANS-ANS:(VCTR(102)-1)SLCT(102)
(3) -BK:(ID-102+1)15DVCTR

```

(3) SORT A GROUP OF NUMBERS:

```

(1) ORD-1-0
(2) LB:O-1-P-UNS
(3) WHICH-UNS:1/UNS
(4) ORD-ORD-WHICH COMP UNS
(5) UNS:(-WHICH) COMP UNS
(6) -LB

```

(4) REVERSE A VECTOR:

```

(1) RV-VCL(1+VCL)-10VC
(2) REV-1-0
(3) REV-1-0
(4) REV-1-0
(5) REV-1-0
(6) REV-1-0
(7) REV-1-0
(8) REV-1-0
(9) REV-1-0
(10) REV-1-0
(11) REV-1-0
(12) REV-1-0
(13) REV-1-0
(14) REV-1-0
(15) REV-1-0
(16) REV-1-0
(17) REV-1-0
(18) REV-1-0
(19) REV-1-0
(20) REV-1-0
(21) REV-1-0
(22) REV-1-0
(23) REV-1-0
(24) REV-1-0
(25) REV-1-0
(26) REV-1-0
(27) REV-1-0
(28) REV-1-0
(29) REV-1-0
(30) REV-1-0
(31) REV-1-0
(32) REV-1-0
(33) REV-1-0
(34) REV-1-0
(35) REV-1-0
(36) REV-1-0
(37) REV-1-0
(38) REV-1-0
(39) REV-1-0
(40) REV-1-0
(41) REV-1-0
(42) REV-1-0
(43) REV-1-0
(44) REV-1-0
(45) REV-1-0
(46) REV-1-0
(47) REV-1-0
(48) REV-1-0
(49) REV-1-0
(50) REV-1-0
(51) REV-1-0
(52) REV-1-0
(53) REV-1-0
(54) REV-1-0
(55) REV-1-0
(56) REV-1-0
(57) REV-1-0
(58) REV-1-0
(59) REV-1-0
(60) REV-1-0
(61) REV-1-0
(62) REV-1-0
(63) REV-1-0
(64) REV-1-0
(65) REV-1-0
(66) REV-1-0
(67) REV-1-0
(68) REV-1-0
(69) REV-1-0
(70) REV-1-0
(71) REV-1-0
(72) REV-1-0
(73) REV-1-0
(74) REV-1-0
(75) REV-1-0
(76) REV-1-0
(77) REV-1-0
(78) REV-1-0
(79) REV-1-0
(80) REV-1-0
(81) REV-1-0
(82) REV-1-0
(83) REV-1-0
(84) REV-1-0
(85) REV-1-0
(86) REV-1-0
(87) REV-1-0
(88) REV-1-0
(89) REV-1-0
(90) REV-1-0
(91) REV-1-0
(92) REV-1-0
(93) REV-1-0
(94) REV-1-0
(95) REV-1-0
(96) REV-1-0
(97) REV-1-0
(98) REV-1-0
(99) REV-1-0
(100) REV-1-0

```

(5) HALF THE LEFT ARGUMENT TO THE RIGHT ARGUMENT:

```

(1) ET-1/BS-1-100M
(2) MAIS 3

```

SAMPLE DIALOG IN EAPL RUNNING APL CHARACTER SET:

```

EAPL 1.0
CLEAR US
*/110

```

```

55 9'1'135
BCNCHIDKMOPOQSTUVWIZ
RV-REV VC
>10 RV-VCL(1+VCL)-10VC
>0
RV-REV VC
[10] RV-VCL(1+VCL)-10VC

```

```

>V
5 REV TESTING 1
CHITSET
AVC
TESTING
RV-1-1-1
2025 FREE
VASCAL
>P-1

```

```

ERROR 3027
>10 P-1
>20 -PRINT-MAP*(0,P)-P-0
>15 PRINT:P
>21C
>0
VASCAL
[10] P-1
[20] PRINT:P
[30] -PRINT-MAP*(0,P)-P-0

```

```

>V
PASCAL
1 1
1 1
1 2 1
1 3 3 1
1 6 6 4 1
1 5 10 10 5 1
)PMS
REV.PASCAL,
TEST-13000

```

```

ERROR 5097
TEST
ERROR 2010

```

WISC RULES

- 1) NOTE THAT THERE MUST BE A SPACE BETWEEN A '-' AND A NUMBER WHEN THE MISIS SIGN IS BEING USED AS A DYNAMIC OPERATOR AND NOT A CONSTANT NEGATOR. THE SPACE IS NOT NECESSARY IF A NUMBER IS TO THE LEFT OF THE MISIS SIGN.
 - A - 5
 - ERROR 2650
 - A - 5
 - 0
 - A - 5
 - 0
 - 5 - 5
- 2) WHEN CONTROL X (CANCEL LINE) IS TYPED, THE COMPUTER PRESENTLY CLEANS THE BUFFER. THE USER MIGHT WISH TO CHANGE THIS TO PRINT '\ ' THEN 'CR' FOR A PRINTING A CHANGE MAY BE MADE IN LOCATIONS 015 232 TO 015 237.
 - EXAMPLE: 015 232 016 WAI A. \'
 - 015 233 017
 - 015 234 018
 - 015 235 019
 - 015 236 020
 - 015 237 021
 - 015 238 022
 - 015 239 023
 - 015 240 024
 - 015 241 025
 - 015 242 026
 - 015 243 027
 - 015 244 028
 - 015 245 029
 - 015 246 030
 - 015 247 031
 - 015 248 032
 - 015 249 033
 - 015 250 034
 - 015 251 035
 - 015 252 036
 - 015 253 037
 - 015 254 038
 - 015 255 039
 - 015 256 040
 - 015 257 041
 - 015 258 042
 - 015 259 043
 - 015 260 044
 - 015 261 045
 - 015 262 046
 - 015 263 047
 - 015 264 048
 - 015 265 049
 - 015 266 050
 - 015 267 051
 - 015 268 052
 - 015 269 053
 - 015 270 054
 - 015 271 055
 - 015 272 056
 - 015 273 057
 - 015 274 058
 - 015 275 059
 - 015 276 060
 - 015 277 061
 - 015 278 062
 - 015 279 063
 - 015 280 064
 - 015 281 065
 - 015 282 066
 - 015 283 067
 - 015 284 068
 - 015 285 069
 - 015 286 070
 - 015 287 071
 - 015 288 072
 - 015 289 073
 - 015 290 074
 - 015 291 075
 - 015 292 076
 - 015 293 077
 - 015 294 078
 - 015 295 079
 - 015 296 080
 - 015 297 081
 - 015 298 082
 - 015 299 083
 - 015 300 084
 - 015 301 085
 - 015 302 086
 - 015 303 087
 - 015 304 088
 - 015 305 089
 - 015 306 090
 - 015 307 091
 - 015 308 092
 - 015 309 093
 - 015 310 094
 - 015 311 095
 - 015 312 096
 - 015 313 097
 - 015 314 098
 - 015 315 099
 - 015 316 100
 - 015 317 101
 - 015 318 102
 - 015 319 103
 - 015 320 104
 - 015 321 105
 - 015 322 106
 - 015 323 107
 - 015 324 108
 - 015 325 109
 - 015 326 110
 - 015 327 111
 - 015 328 112
 - 015 329 113
 - 015 330 114
 - 015 331 115
 - 015 332 116
 - 015 333 117
 - 015 334 118
 - 015 335 119
 - 015 336 120
 - 015 337 121
 - 015 338 122
 - 015 339 123
 - 015 340 124
 - 015 341 125
 - 015 342 126
 - 015 343 127
 - 015 344 128
 - 015 345 129
 - 015 346 130
 - 015 347 131
 - 015 348 132
 - 015 349 133
 - 015 350 134
 - 015 351 135
 - 015 352 136
 - 015 353 137
 - 015 354 138
 - 015 355 139
 - 015 356 140
 - 015 357 141
 - 015 358 142
 - 015 359 143
 - 015 360 144
 - 015 361 145
 - 015 362 146
 - 015 363 147
 - 015 364 148
 - 015 365 149
 - 015 366 150
 - 015 367 151
 - 015 368 152
 - 015 369 153
 - 015 370 154
 - 015 371 155
 - 015 372 156
 - 015 373 157
 - 015 374 158
 - 015 375 159
 - 015 376 160
 - 015 377 161
 - 015 378 162
 - 015 379 163
 - 015 380 164
 - 015 381 165
 - 015 382 166
 - 015 383 167
 - 015 384 168
 - 015 385 169
 - 015 386 170
 - 015 387 171
 - 015 388 172
 - 015 389 173
 - 015 390 174
 - 015 391 175
 - 015 392 176
 - 015 393 177
 - 015 394 178
 - 015 395 179
 - 015 396 180
 - 015 397 181
 - 015 398 182
 - 015 399 183
 - 015 400 184
 - 015 401 185
 - 015 402 186
 - 015 403 187
 - 015 404 188
 - 015 405 189
 - 015 406 190
 - 015 407 191
 - 015 408 192
 - 015 409 193
 - 015 410 194
 - 015 411 195
 - 015 412 196
 - 015 413 197
 - 015 414 198
 - 015 415 199
 - 015 416 200
 - 015 417 201
 - 015 418 202
 - 015 419 203
 - 015 420 204
 - 015 421 205
 - 015 422 206
 - 015 423 207
 - 015 424 208
 - 015 425 209
 - 015 426 210
 - 015 427 211
 - 015 428 212
 - 015 429 213
 - 015 430 214
 - 015 431 215
 - 015 432 216
 - 015 433 217
 - 015 434 218
 - 015 435 219
 - 015 436 220
 - 015 437 221
 - 015 438 222
 - 015 439 223
 - 015 440 224
 - 015 441 225
 - 015 442 226
 - 015 443 227
 - 015 444 228
 - 015 445 229
 - 015 446 230
 - 015 447 231
 - 015 448 232
 - 015 449 233
 - 015 450 234
 - 015 451 235
 - 015 452 236
 - 015 453 237
 - 015 454 238
 - 015 455 239
 - 015 456 240
 - 015 457 241
 - 015 458 242
 - 015 459 243
 - 015 460 244
 - 015 461 245
 - 015 462 246
 - 015 463 247
 - 015 464 248
 - 015 465 249
 - 015 466 250
 - 015 467 251
 - 015 468 252
 - 015 469 253
 - 015 470 254
 - 015 471 255
 - 015 472 256
 - 015 473 257
 - 015 474 258
 - 015 475 259
 - 015 476 260
 - 015 477 261
 - 015 478 262
 - 015 479 263
 - 015 480 264
 - 015 481 265
 - 015 482 266
 - 015 483 267
 - 015 484 268
 - 015 485 269
 - 015 486 270
 - 015 487 271
 - 015 488 272
 - 015 489 273
 - 015 490 274
 - 015 491 275
 - 015 492 276
 - 015 493 277
 - 015 494 278
 - 015 495 279
 - 015 496 280
 - 015 497 281
 - 015 498 282
 - 015 499 283
 - 015 500 284
 - 015 501 285
 - 015 502 286
 - 015 503 287
 - 015 504 288
 - 015 505 289
 - 015 506 290
 - 015 507 291
 - 015 508 292
 - 015 509 293
 - 015 510 294
 - 015 511 295
 - 015 512 296
 - 015 513 297
 - 015 514 298
 - 015 515 299
 - 015 516 300
 - 015 517 301
 - 015 518 302
 - 015 519 303
 - 015 520 304
 - 015 521 305
 - 015 522 306
 - 015 523 307
 - 015 524 308
 - 015 525 309
 - 015 526 310
 - 015 527 311
 - 015 528 312
 - 015 529 313
 - 015 530 314
 - 015 531 315
 - 015 532 316
 - 015 533 317
 - 015 534 318
 - 015 535 319
 - 015 536 320
 - 015 537 321
 - 015 538 322
 - 015 539 323
 - 015 540 324
 - 015 541 325
 - 015 542 326
 - 015 543 327
 - 015 544 328
 - 015 545 329
 - 015 546 330
 - 015 547 331
 - 015 548 332
 - 015 549 333
 - 015 550 334
 - 015 551 335
 - 015 552 336
 - 015 553 337
 - 015 554 338
 - 015 555 339
 - 015 556 340
 - 015 557 341
 - 015 558 342
 - 015 559 343
 - 015 560 344
 - 015 561 345
 - 015 562 346
 - 015 563 347
 - 015 564 348
 - 015 565 349
 - 015 566 350
 - 015 567 351
 - 015 568 352
 - 015 569 353
 - 015 570 354
 - 015 571 355
 - 015 572 356
 - 015 573 357
 - 015 574 358
 - 015 575 359
 - 015 576 360
 - 015 577 361
 - 015 578 362
 - 015 579 363
 - 015 580 364
 - 015 581 365
 - 015 582 366
 - 015 583 367
 - 015 584 368
 - 015 585 369
 - 015 586 370
 - 015 587 371
 - 015 588 372
 - 015 589 373
 - 015 590 374
 - 015 591 375
 - 015 592 376
 - 015 593 377
 - 015 594 378
 - 015 595 379
 - 015 596 380
 - 015 597 381
 - 015 598 382
 - 015 599 383
 - 015 600 384
 - 015 601 385
 - 015 602 386
 - 015 603 387
 - 015 604 388
 - 015 605 389
 - 015 606 390
 - 015 607 391
 - 015 608 392
 - 015 609 393
 - 015 610 394
 - 015 611 395
 - 015 612 396
 - 015 613 397
 - 015 614 398
 - 015 615 399
 - 015 616 400
 - 015 617 401
 - 015 618 402
 - 015 619 403
 - 015 620 404
 - 015 621 405
 - 015 622 406
 - 015 623 407
 - 015 624 408
 - 015 625 409
 - 015 626 410
 - 015 627 411
 - 015 628 412
 - 015 629 413
 - 015 630 414
 - 015 631 415
 - 015 632 416
 - 015 633 417
 - 015 634 418
 - 015 635 419
 - 015 636 420
 - 015 637 421
 - 015 638 422
 - 015 639 423
 - 015 640 424
 - 015 641 425
 - 015 642 426
 - 015 643 427
 - 015 644 428
 - 015 645 429
 - 015 646 430
 - 015 647 431
 - 015 648 432
 - 015 649 433
 - 015 650 434
 - 015 651 435
 - 015 652 436
 - 015 653 437
 - 015 654 438
 - 015 655 439
 - 015 656 440
 - 015 657 441
 - 015 658 442
 - 015 659 443
 - 015 660 444
 - 015 661 445
 - 015 662 446
 - 015 663 447
 - 015 664 448
 - 015 665 449
 - 015 666 450
 - 015 667 451
 - 015 668 452
 - 015 669 453
 - 015 670 454
 - 015 671 455
 - 015 672 456
 - 015 673 457
 - 015 674 458
 - 015 675 459
 - 015 676 460
 - 015 677 461
 - 015 678 462
 - 015 679 463
 - 015 680 464
 - 015 681 465
 - 015 682 466
 - 015 683 467
 - 015 684 468
 - 015 685 469
 - 015 686 470
 - 015 687 471
 - 015 688 472
 - 015 689 473
 - 015 690 474
 - 015 691 475
 - 015 692 476
 - 015 693 477
 - 015 694 478
 - 015 695 479
 - 015 696 480
 - 015 697 481
 - 015 698 482
 - 015 699 483
 - 015 700 484
 - 015 701 485
 - 015 702 486
 - 015 703 487
 - 015 704 488
 - 015 705 489
 - 015 706 490
 - 015 707 491
 - 015 708 492
 - 015 709 493
 - 015 710 494
 - 015 711 495
 - 015 712 496
 - 015 713 497
 - 015 714 498
 - 015 715 499
 - 015 716 500
 - 015 717 501
 - 015 718 502
 - 015 719 503
 - 015 720 504
 - 015 721 505
 - 015 722 506
 - 015 723 507
 - 015 724 508
 - 015 725 509
 - 015 726 510
 - 015 727 511
 - 015 728 512
 - 015 729 513
 - 015 730 514
 - 015 731 515
 - 015 732 516
 - 015 733 517
 - 015 734 518
 - 015 735 519
 - 015 736 520
 - 015 737 521
 - 015 738 522
 - 015 739 523
 - 015 740 524
 - 015 741 525
 - 015 742 526
 - 015 743 527
 - 015 744 528
 - 015 745 529
 - 015 746 530
 - 015 747 531
 - 015 748 532
 - 015 749 533
 - 015 750 534
 - 015 751 535
 - 015 752 536
 - 015 753 537
 - 015 754 538
 - 015 755 539
 - 015 756 540
 - 015 757 541
 - 015 758 542
 - 015 759 543
 - 015 760 544
 - 015 761 545
 - 015 762 546
 - 015 763 547
 - 015 764 548
 - 015 765 549
 - 015 766 550
 - 015 767 551
 - 015 768 552
 - 015 769 553
 - 015 770 554
 - 015 771 555
 - 015 772 556
 - 015 773 557
 - 015 774 558
 - 015 775 559
 - 015 776 560
 - 015 777 561
 - 015 778 562
 - 015 779 563
 - 015 780 564
 - 015 781 565
 - 015 782 566
 - 015 783 567
 - 015 784 568
 - 015 785 569
 - 015 786 570
 - 015 787 571
 - 015 788 572
 - 015 789 573
 - 015 790 574
 - 015 791 575
 - 015 792 576
 - 015 793 577
 - 015 794 578
 - 015 795 579
 - 015 796 580
 - 015 797 581
 - 015 798 582
 - 015 799 583
 - 015 800 584
 - 015 801 585
 - 015 802 586
 - 015 803 587
 - 015 804 588
 - 015 805 589
 - 015 806 590
 - 015 807 591
 - 015 808 592
 - 015 809 593
 - 015 810 594
 - 015 811 595
 - 015 812 596
 - 015 813 597
 - 015 814 598
 - 015 815 599
 - 015 816 600
 - 015 817 601
 - 015 818 602
 - 015 819 603
 - 015 820 604
 - 015 821 605
 - 015 822 606
 - 015 823 607
 - 015 824 608
 - 015 825 609
 - 015 826 610
 - 015 827 611
 - 015 828 612
 - 015 829 613
 - 015 830 614
 - 015 831 615
 - 015 832 616
 - 015 833 617
 - 015 834 618
 - 015 835 619
 - 015 836 620
 - 015 837 621
 - 015 838 622
 - 015 839 623
 - 015 840 624
 - 015 841 625
 - 015 842 626
 - 015 843 627
 - 015 844 628
 - 015 845 629
 - 015 846 630
 - 015 847 631
 - 015 848 632
 - 015 849 633
 - 015 850 634
 - 015 851 635
 - 015 852 636
 - 015 853 637
 - 015 854 638
 - 015 855 639
 - 015 856 640
 - 015 857 641
 - 015 858 642
 - 015 859 643
 - 015 860 644
 - 015 861 645
 - 015 862 646
 - 015 863 647
 - 015 864 648
 - 015 865 649
 - 015 866 650
 - 015 867 651
 - 015 868 652
 - 015 869 653
 - 015 870 654
 - 015 871 655
 - 015 872 656
 - 015 873 657
 - 015 874 658
 - 015 875 659
 - 015 876 660
 - 015 877 661
 - 015 878 662
 - 015 879 663
 - 015 880 664
 - 015 881 665
 - 015 882 666
 - 015 883 667
 - 015 884 668
 - 015 885 669
 - 015 886 670
 - 015 887 671
 - 015 888 672
 - 015 889 673
 - 015 890 674
 - 015 891 675
 - 015 892 676
 - 015 893 677
 - 015 894 678
 - 015 895 679
 - 015 896 680
 - 015 897 681
 - 015 898 682
 - 015 899 683
 - 015 900 684
 - 015 901 685
 - 015 902 686
 - 015 903 687
 - 015 904 688
 - 015 905 689
 - 015 906 690
 - 015 907 691
 - 015 908 692
 - 015 909 693
 - 015 910 694
 - 015 911 695
 - 015 912 696
 - 015 913 697
 - 015 914 698
 - 015 915 699
 - 015 916 700
 - 015 917 701
 - 015 918 702
 - 015 919 703
 - 015 920 704
 - 015 921 705
 - 015 922 706
 - 015 923 707
 - 015 924 708
 - 015 925 709
 - 015 926 710
 - 015 927 711
 - 015 928 712
 - 015 929 713
 - 015 930 714
 - 015 931 715
 - 015 932 716
 - 015 933 717
 - 015 934 718
 - 015 935 719
 - 015 936 720
 - 015 937 721
 - 015 938 722
 - 015 939 723
 - 015 940 724

EMC08 ADAPTED ASCII CHARACTER SET 5/77 A

DELIMITER	EMC08	ASCII	DELIMITER	EMC08	ASCII
0	300	0	1	333	1
1	301	1	2	334	2
2	302	2	3	335	3
3	303	3	4	336	4
4	304	4	5	337	5
5	305	5	6	338	6
6	306	6	7	339	7
7	307	7	8	340	8
8	308	8	9	341	9
9	309	9	10	342	10
10	310	10	11	343	11
11	311	11	12	344	12
12	312	12	13	345	13
13	313	13	14	346	14
14	314	14	15	347	15
15	315	15	16	348	16
16	316	16	17	349	17
17	317	17	18	350	18
18	318	18	19	351	19
19	319	19	20	352	20
20	320	20	21	353	21
21	321	21	22	354	22
22	322	22	23	355	23
23	323	23	24	356	24
24	324	24	25	357	25
25	325	25	26	358	26
26	326	26	27	359	27
27	327	27	28	360	28
28	328	28	29	361	29
29	329	29	30	362	30
30	330	30	31	363	31
31	331	31	32	364	32
32	332	32	33	365	33
33	333	33	34	366	34
34	334	34	35	367	35
35	335	35	36	368	36
36	336	36	37	369	37
37	337	37	38	370	38
38	338	38	39	371	39
39	339	39	40	372	40
40	340	40	41	373	41
41	341	41	42	374	42
42	342	42	43	375	43
43	343	43	44	376	44
44	344	44	45	377	45
45	345	45	46	378	46
46	346	46	47	379	47
47	347	47	48	380	48
48	348	48	49	381	49
49	349	49	50	382	50
50	350	50	51	383	51
51	351	51	52	384	52
52	352	52	53	385	53
53	353	53	54	386	54
54	354	54	55	387	55
55	355	55	56	388	56
56	356	56	57	389	57
57	357	57	58	390	58
58	358	58	59	391	59
59	359	59	60	392	60
60	360	60	61	393	61
61	361	61	62	394	62
62	362	62	63	395	63
63	363	63	64	396	64
64	364	64	65	397	65
65	365	65	66	398	66
66	366	66	67	399	67
67	367	67	68	400	68
68	368	68	69	401	69
69	369	69	70	402	70
70	370	70	71	403	71
71	371	71	72	404	72
72	372	72	73	405	73
73	373	73	74	406	74
74	374	74	75	407	75
75	375	75	76	408	76
76	376	76	77	409	77
77	377	77	78	410	78
78	378	78	79	411	79
79	379	79	80	412	80
80	380	80	81	413	81
81	381	81	82	414	82
82	382	82	83	415	83
83	383	83	84	416	84
84	384	84	85	417	85
85	385	85	86	418	86
86	386	86	87	419	87
87	387	87	88	420	88
88	388	88	89	421	89
89	389	89	90	422	90
90	390	90	91	423	91
91	391	91	92	424	92
92	392	92	93	425	93
93	393	93	94	426	94
94	394	94	95	427	95
95	395	95	96	428	96
96	396	96	97	429	97
97	397	97	98	430	98
98	398	98	99	431	99
99	399	99	100	432	100

HEX/Octal/BINARY CONVERSION TABLE

HEX	OCTAL	BINARY
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	10	1000
9	11	1001
A	12	1010
B	13	1011
C	14	1100
D	15	1101
E	16	1110
F	17	1111
10	20	10000
11	21	10001
12	22	10010
13	23	10011
14	24	10100
15	25	10101
16	26	10110
17	27	10111
18	30	11000
19	31	11001
1A	32	11010
1B	33	11011
1C	34	11100
1D	35	11101
1E	36	11110
1F	37	11111
20	40	100000
21	41	100001
22	42	100010
23	43	100011
24	44	100100
25	45	100101
26	46	100110
27	47	100111
28	50	101000
29	51	101001
2A	52	101010
2B	53	101011
2C	54	101100
2D	55	101101
2E	56	101110
2F	57	101111
30	60	110000
31	61	110001
32	62	110010
33	63	110011
34	64	110100
35	65	110101
36	66	110110
37	67	110111
38	70	111000
39	71	111001
3A	72	111010
3B	73	111011
3C	74	111100
3D	75	111101
3E	76	111110
3F	77	111111

