ZBasic™ 3.0 QUICK REFERENCE CARD

Standard Commands

APPEND line# filename	Append a no line# program at line#
AUTO [start [,incr]]	Auto line# for inserting lines
DEL, DELETE lines	Delete line(s) from program in mem
DIR [drivespec]	Displays DISK [drive] Directory
EDIT,E or ,[line]	Edits current or [line#]. See Line Editor
FIND [#] string	Find string or line# in pgm. ';' finds next
HELP	Displays a list of CMD's
LIST , L [+ *] line or label	Lists to Display [*]=Highlight cmd's
LLIST [+ *] lines	Lists to Printer [+]=no line#
LOAD [*] filename	Load Program. [*]=Condense Pgm
MEM [ORY]	Displays memory utilization
MERGE [*] filename	Merge ASCII ZBasic program
NEW	Clears Program from memory
QUIT	Exit ZBasic to System
RENUM [new ,old ,incr]	Renumbers program in memory
RUN [filename]	Execute from Memory or [Disk file]
RUN * [filename]	Compile & Save object code
RUN + [filename]	Compile & Save Chain object code
SAVE filename	Save program ZBasic Tokenized
SAVE * filename	Save program in ASCII with line #'s
SAVE + filename	Save program in ASCII no line#'s

Standard Line Editor

<Break> or <CTRL C> Abort all changes and exit Line Edit mode.

[n]	<BKSPACE>	Move cursor backwards [n] characters
	<ENTER>	Accept & insert EDITED line
	<ESC>ape	Abort C,H,I,K,S,X commands
[n]	<SPACE>	Move cursor forward [n] characters
	A	Abort and re-load line to EDIT
[n]	C <Key[s]>	Change [n] characters to <Key['s]>
[n]	D	Delete [n] char(s) from cursor to the right
	H	Hack to end of line and enter insert mode.
	I	Insert characters at current cursor position
[n]	K <Key>	Delete to [n]th occurence of <Key>
	L	List remainder of line and start EDIT again
[n]	S <Key>	Move cursor to [n]th occurence of <Key>
	X	Go to end of line and enter insert mode.

To edit a line: EDIT (or E) Line Number or Label n defaults to 1
'.'=EDIT current line, '.'=LIST current line, '/'=LIST 10 lines

String Functions

Returns the following String or Number

ASC (str$)	ASCII value of the 1st char in a string.
BIN$ (expr)	16 char Binary "bbbbbbbbbbbbbbbb", b=0/1
CHR$ (expr)	String = ASCII of expr, CHR$(65)="A"
DATE$	Date in MM/DD/YY format "08/11/85"
HEX$ (expr)	4 character HEX string " 0000" to "FFFF"
INDEX$ (expr)	String at INDEX$(expr)
INSTR (pos,v1$,v2$)	Finds v2$ in v1$ Start=pos ,0=Not found
LEN (string$)	Length from 0 to 255 of string
LEFT$ (str$,len)	String of len from left side of str$
MID$ (str$,pos [,len])	String starting at pos for len
MKB$ (expr)	BCD binary string of double precision
MKI$ (expr)	2 chr Integer binary string
OCT$ (expr)	6 chr string of octal "00000" to "377777"
PSTR$ (expr)	String pointed to by address expr
RIGHT$ (str$,len)	Len characters from right side
SPACE$ (expr)	String of expr spaces
STR$ (expr)	Numeric String for expr like "-123.456"
STRING$ (expr,char)	String of length expr all char's "********"
TIME$	Time in HH:MM:SS format "03:01:45"
UCASE$ (str$)	Str$ all lower case returned as upper
UNS$ (expr)	Unsigned 5 digit int. UNS$(-1)="65535"

Program Statements

CLEAR	Clears all var's to 0's and 0 len var$
CLEAR number	Allocates INDEX$ memory
CLEAR END	Clears for variables not common'ed
CLEAR INDEX$	Clears INDEX$ to all nulls
DATA item[,item.]	Store in data table for READ
DEFINT var, var-var	Set variable default type Integer %
DEFSNG var,var-var	Set variable default type Single !
DEFDBL var,var-var	Set variable default type Double #
DEFSTR var,var-var	Set variable default type String $
DEF FN name [(var[s]...)]	Defines a function by name
DEF LEN [=] expr	Sets string var's Max length
DELAY [=] expr	Delay expr milliseconds 1/1000 sec
DIM [len] var[(num..)][,...]	Assign arrays & variables
DO... UNTIL cond	Loops UNTIL cond TRUE<>0
ELSE	Else executed if IF cond is FALSE
END	Ends program execution
END IF	Ends 'LONG IF'
END FN [= expr]	Ends 'LONG FN' [Returns expr]
FN name [(expr's...)]	Executes function by name
FOR var = x TO y [STEP z]	Loops at least once from NEXT
GOSUB line or label	Subroutine line# or "label"
GOTO line or label	Continue exec at line# or "label"
IF cond THEN...ELSE	True if cond<>0 else false
INDEX$ (expr)=string	Replace INDEX$(expr)
INDEX$I(expr)=string	Insert at INDEX$(expr)
INDEX$D(expr)	Deletes INDEX$(expr)
[LET] var = expression	Sets var to expression
LONG FN name [(v.)]	Multi line Funtion Definition
LONG IF cond	Multi line 'IF' construct start
MID$ (v$,pos,len) = str$	Sets middle pos of V$ for len = str$
NEXT [v [,v...]]	Terminates FOR loop[s...]
ON x GOSUB L1,L2	IF x=1 THEN GOSUB L1 line# or label
ON x GOTO L1,L2	IF x=2 THEN GOTO L2 line# or label
PSTR$(V%)="string"	Sets V% to point to "string"
RANDOM [expr]	Sets random # seed.
READ var,PSTR$(v%)	Reads DATA into variables
REM Remark stuff....	Program remarks Ignored...
RESTORE [item#]	Point to 1st or [expr] DATA item
RETURN [line]	Return from GOSUB [Skip/GOTO]
SOUND freq,duration	Sound of freq hz,ms duration
STEP expr	Changes Default STEP+1 FOR/NEXT
STOP	Stop and print Break in nnnnnn
SWAP var,var	Swap contents of 2 Variables
TRON[,X,B,S]	Debug,<B>reak & <S>ingle step
UNTIL cond	End of DO loop if TRUE else Loop.
WEND	Marks End of WHILE loop
WHILE cond	Loop WHILE expr is TRUE
XELSE	ELSE for 'LONG IF'

Machine Specific Commands

CALL nnnnn or LINE line	CALL's addr nnnnn or MACHLG line
DEF USR digit = address	Sets USR CALL address to expr
INP (port)	Reads Byte data from I/O port
LINE line	Returns Start address Compiled Line.
OUT port,data	Outputs DATA to I/O PORT
PEEK [WORD] (addr)	Byte or WORD memory read addr.
POKE [WORD] addr,d	Byte or WORD memory Write addr.
MACHLG byte,word,var	Creates Inline CODE to execute
USR digit (expr)	USR Statement&Function.0 to 9

8086 versions replace SEG with addr[,segment] for PEEK & POKE
Special note for 32 bit versions of ZBasic 3.0:
POKE LONG & PEEK LONG are 32 bit Memory READ & WRITE
32bit Double INTEGER Variable type using var& or DEFDBLINT

Numbers and Ranges

Type	Minumum		Maximum	Memory Required in Bytes
Integer	-32768	to	+32767	2
Real(BCD)	-9.999E+63	to	+9.999E+63	2 to 28
Hex	&H0000	to	&HFFFF	2
Octal	&O000000	to	&O377777	2
Binary	&Xbbbbbbbbbbbbbbbb b=0 or 1			2

OVERFLOW ERRORS: BCD: 9.99999E+63, INTEGER: NONE
Strings of CHR$(0 to 255), LEN from 0 to 255 char, (See DEF LEN)
BCD Accuracy for Single, Double & Scientific: 6 to 54 digits

Math Operators

^ or [	Raise to the power (REAL)
+ - * / \	Add,Sub,Mult,Divide,Divide (REAL)
AND OR XOR	Binary & Conditional operators
MOD	Remainder of a division
NOT	Binary & Conditional operator
<,=,>,<=,=<,>=,=>,<>	Conditional operators 0=false,-1=TRUE
<<,>>	Shift Left *2^int., Shift Right /2^int.

Variable types: %=Integer !=Single #=Double $=String

Numeric Functions

Returns the following Numeric Result:

ABS (expr)	Absolute Value of expr always positive.
ASC (str$)	ASCII value of 1st character in string
ATN (expr)	Arctangent in radians of expr (REAL)
COS (expr)	COSINE of expr in radians (REAL)
CVI (str$)	INTEGER value of 1st 2 bytes in string
EXP (expr)	Returns e^expr (REAL)
FIX (expr)	Truncates digits right of d.p. (REAL)
FRAC (expr)	Truncates digits left of d.p. (REAL)
INDEXF(str[,s])	Finds leading str in INDEX$ starting at s
INT (expr)	Truncates digits right of d.p.
LOG (expr)	Natural log of the expr (REAL)
MAYBE	0 or -1, random 50% of the time
MEM	Room in bytes available in INDEX$ area
PAGE	Present line on printer page starting at 0
POS (0,1,2)	Character Position of screen,printer,Disk
RND (expr)	Random # from 1 to expr
SGN (expr)	Sign of expr -1 if neg,0 if zero,+1 if positive
SIN (expr)	SINE of expr in radians (REAL)
SQR (expr)	SQUARE ROOT of expr (REAL)
TAN (expr)	TANGENT of expr in radians (REAL)
VAL (str$)	Converts Numeric String to number
VARPTR (var)	ADDRESS of the variable in memory.

INTEGER is assumed unless a REAL number,var or function used

Input/Output Instructions

CLS	Clears Screen
CLS [char.]	Clears Screen with Char's
CLS LINE	Clears from cursor to end of line
CLS PAGE	Clears from cursor to end of page
DEF TAB [=] expr	Sets comma TAB stops. Default=16
INPUT [vpos] [&maxlen,] [!] [;] ["string";] v [,v...]	See Notes at Bottom
LINE INPUT [vpos] [&maxlen,] [!] [;] ["string";] v$	
INKEY$	KEY$ if pressed else "" null.
LPRINT stuff	Print to PRINTER device
PAGE len,page,top margin	LPRINT & LLIST Page formatting
PRINT [vpos] stuff...	Prints to DISPLAY device
SPC(expr)	Prints expr spaces
TAB(expr)	Move cursor to character position
USING str$,expr	Formats Numeric Quantitys
WIDTH [=] width	Sets PRINT width
WIDTH LPRINT [=] width	Sets LPRINT width

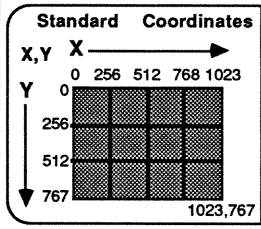
vpos=See cursor positioning. &maxlen=Maximum # of key's
!=accept line after maxlen char's entered ; =Surpress <cr><lf>

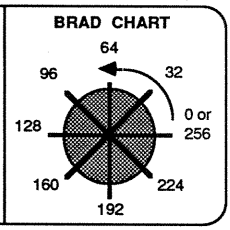
Device Independent Graphics

CIRCLE h,v,r — BOX
CIRCLE FILL — BOX FILL
circle TO — PLOT TO
circle PLOT — PLOT

RATIO: 0,0 | -64,0 | 64,0 | 0,64

COLOR Black=Ø, -1=White MODE Even=Char, Odd=Graphics





Graphics Instructions

BOX [x,y] [TO x,y..]	Empty BOX x,y opposite corners
BOX FILL [x,y] [TO x,y...]	Solid BOX x,y opposite corners
CIRCLE x,y,r	Empty Circle at x,y center of radius r
CIRCLE FILL x,y,r	Solid Circle at x,y of radius r
CIRCLE x,y,r TO s,n	Pie segment s=start,n=# of BRAD's
CIRCLE x,y,r PLOT s,n	Arc Segment s=start,n=# of BRAD's
COLOR [=] color	Color to be used 0=Black,-1=White
FILL x,y	Fills graphic area around x,y
MODE expr 0-15	Sets Screen, MODE 7=High Res
PLOT [TO] x,y [TOx,y...]	Plots point or draw line point TO point
POINT (x,y)	Returns COLOR at x,y
RATIO xAspect,yAspect	CIRCLE aspect ratio ±127
SOUND freq,duration	Output sound hertz,milliseconds

Screen Coordinates: 0,0=upper left, 1023,767=lower right
512,384=Screen center. BRAD's from 0 to 256 = 0 to 360 degrees
Also see Cursor Positioning and Standard Screen Coordinates

Cursor Positioning (vpos)

@(x,y)	Puts cursor at position x char's from left on line y
%(x,y)	Puts cursor at X horiz, y verti GRAPHIC coordinate

@ and % may be used with PRINT, INPUT and LINEINPUT

LOCATE x,y [,cur] Set x,y cursor position [cursor -1=on/0=off]

Disk File Commands

ERROR = expr	Sets disk error number
ERROR	Returns DISK error Number
CLOSE [Fnum [,Fnum]]	Closes all files or [Fnum(s)]
INPUT # Fnum,var[,var..]	Reads ASCII file into var[s]
KILL filename	Variable$ or "string"
LINE INPUT#fnum,v$	Gets ASCII string into var$
LOC (fnum)	Returns location in Disk record.
LOF (fnum)	Returns File size in records
ON ERROR GOSUB line	Error vector [RETURN or 65535]
OPEN "I,O,R",Fnum,"file" [,rl]	Opens "file" as Fnum with rec len=rl
PRINT # Fnum,stuff	Prints stuff in ASCII to DISK
READ # Fnum,v,v$;length	Reads binary info from DISK
REC (fnum)	Returns Current DISK Record #
RECORD fnum,rec[,loc]	Position for next DISK operation
RENAME old TO new	Renames old to new Filenames
ROUTE expr or Fnum	Routes PRINT, 0=Video,128=Printer
RUN [Fnum]	RUN or RUN fnum chain files
WRITE#fnum,v,v$;len	Writes Binary info to DISK

Notes: