***#JustBasic manual***

*Inspired by Sinclair BASIC.*

***Code formatting***

*You can freely format code: many commands in single line, tabs, etc. PICO tested w/ Putty. Copy & Paste supported.*

*Example*

*print “Hi” for k=1 to 10 print k next k*

*end*

***Program structure***

*Command ‘end’ is required following the last line of code. subroutines shall follow the ‘end’ command.*

*code*

*end*

*subroutines*

*Example*

*print “Hi!”*

*gosub doitagain*

*end*

*doitagain:*

*print “Hi!”*

*return*

***System modes***

*Two modes using different prompts: user (’>’ prompt) and enhanced (‘#’ prompt)*

***-U-MODE*** *(aka single-line mode) can be entered by issuing: ‘U’ command; useful to run single line programs, for example:*

*m=3 data 2, 3, 4 sum=0 for i=1 to m read a sum=sum+a next i print “s=", sum, ", av= ", sum/m end*

*E – enter E-mode*

***-E-MODE*** *(aka enhanced mode); can be entered by issuing: ‘E’ command; this mode supports many system commands:*

*? - shows quick info about the VM (version, list of available commands)*

*H – help – shows some program examples*

*U – enter U-mode*

*C – code – list current program code*

*R – run program*

*N – new – clear VM memory and code*

*B – bye – reboots the PICO in USB disk mode, in Windows version exists the VM*

*L - load program for persistent memory (auto.bas) - TBD*

*S – save program to persistent memory (auto.bas) - TBD*

*T- tech data (after ‘****R’*** *- VM state incl. the runtime results, after ‘****N’->’L’*** *- the state after initial tokenization)*

*Built-in editor*

*Every command is appended to the existing code. ‘C’ shows code and internal line numbers used for @N commands.*

*@N (e.g. @3) – removes Nth line of code (use C to see the line numbers)*

*@N cmd (e.g. @4 PRINT 5 – inserts ‘PRINT 5’ before line 4) inserts the new code line following the @N before the Nth line*

*Built-in debugger*

*T0 – disables tracing*

*T1 – enables stepping mode*

*T2 – normal run with tracing*

***The programming language***

*-ESC - break program when loading or running;*

*-DEBUG**(enables/disables internal debug messages) – do we need it?*

*-REM - comment*

*-CLS – clear screen*

*-SYSMODE options: MATH(INT+INT/FLOAT); GRAPHIC(NONE, EXPLORER, OLED); CONSOLE(1st/2nd-core) - TBD*

***VARIABLES, EXPRESSIONS***

*-var types (suffix matters): INT (no suffix, name=expr), FLOAT (suffix ‘#’, name#=expr), STRING (suffix ‘$’, name$=expr)*

*-variable name: up to 8chars letter&digits starting w/ a letter(digits, ‘#”, ‘$’, ‘\_’, ‘:’ accepted),*

*-var initialization: 1st-reference creates var(value=0); any variable can be assigned an expressions: var=expr;*

*-expressions: INT/FLOAT, +, -, \*, /, %,(, ), vars;*

*-array:1-dimension; INT/FLOAT supported; STRING NOT supported; DIM name(size); name(item)=expr*

*-strings supports only ‘+’ in expressions + string functions and variables*

*Example*

*hi5=2*

*w#=2.5*

*name$=”John”*

*Example*

*DIM a(3)*

*a(0)=3*

*DIM(b#(3)*

*b#(0)=2.5*

*Example*

*sy=2\*abs(-15) + a\*20*

*v#=fexpr; b#=a#-2\*(2+3)+abs(-1.0)+aa#(2);*

*Example*

*v$=“a”+a$+left$(str$(13),1);*

*PRINT sexpr; VAL(“-1”)->-1*

***LOOPS & PROGRAM FLOW***

*-label name: up to 8chars letter & digits starting w/ a letter( ‘\_’ accepted),*

*-loop/if nesting supported*

*-FOR var=expr TO expr [STEP expr] [] NEXT var; if STEP[default=1] is negative var decreases; FLOAT supported*

*Example*

*FOR i = 5 TO 1 STEP -1 NEXT I END*

*-WHILE expr1 op expr2 [code] ENDWHILE; FLOAT supported;*

*Example:*

*a=0 while a<5 print a a=a+1 endwhile end*

*-GOTO – label (aka name with colon); can be located everywhere (before and after the GOTO): Label: [code] GOTO label*

*Example*

*k=1*

*again:*

*print k k=k+1*

*if k<5 then goto again: endif*

*end*

*-GOSUB – label can be located everywhere (before and after the GOTO): Label: [code] RETURN GOSUB label*

*Example*

*gosub task0*

*end*

*task0: print “done” return*

*-IF conditions THEN [code] [ELSE] [code] ENDIF; FLOAT supported; AND/OR/NOT) supported*

*Example*

*if a>1 and b#>3.4 then print “good” else print “bad” endif*

*-END last instruction (GOSUB labels can be located behind the END)*

***INPUT, OUTPUT, DATA***

*-PRINT expr, sexpr,…( separate items by ‘,’); ‘;’ at the end to skip NEW LINE;*

*Example*

*PRINT "How:", 6/3 ;*

*PRINT 1 (prints 1 w/ NEW LINE); PRINT 1; (prints 1 w/o NEW LINE)*

*-INPUT – assignes int/float/string values to var or array element: INPUT var, array\_element, a$, …;*

*Example*

*INPUT a(2), d#, name$*

*print a(2), d#, name$*

*-DATA expr, fexpr, str; READ a, b#, d$; RESTORE clears data pointer; INT/FLOAT/STR supported*

*Example*

*DATA 1.5, 2\*a; READ v, v#, v(), v#();*

***BUILT-IN FUNCTIONS***

*LEFT$/RIGHT$/MID$, LEN/VAL(sexpr);*

*HEX$/STR$/CHR$(expr);*

*SIN/COS/SQR/EXP/LOG;*

*RND/SGN/ABS*

*GETTICK/PAUSE*

*INKEY*

***HARDWARE SUPPORT***

*PEEK, POKE,*

*AREAD, AWRITE - TBD*

*PMODE, DREAD, DWRITE*