26

Reverse Polish calculator

Reverse Polish notation is a method of working complex mathematical formulae, invented a long time ago by a Polish mathematician. Of course, that introduction will immediately put off a lot of people - but don't let it! For one thing, Reverse Polish Notation - or RPN for short - is so easy to understand that after a few minutes of working with this calculator, you'll wonder why PRN isn't in wide use. For another, it is enormously useful and cuts out all of those horrendous-looking brackets that you see in formulae. Finally, this program will provide you with a very useful calculator - and one that not only offers calculating ability, but will also offer help if you need it!

On typing RUN, the screen clears and immediately fills with the display shown in the illustration. The calculator is ready for use. PROCDISPLAY and PROCBOUNDS take care of this, and if we turn to the illustration, or the listing between lines 580 and 670, you will observe that a number of functions are offered at the touch of a key. You will know, or be able to guess at, most of the functions. Pressing the letter P, for example, immediately produces the value of pi ready for calculating with. However, there are some commands that may be strange, such as POP or ROTATE, and to understand these we need to look into RPN a little.

Using normal notation, we write 3 x 4, but in RPN we write 3, 4, x - the operator following the two numbers that it works upon - hence the ' reversebf the title. To do this, we require a ' stackupon which is stored the numbers to be worked on. The point about any stack (a stack of plates, for example) is that last item added to the stack is the first off; as distinct from a queue, in which the first in is the first out. In this computer program, we are using an upside-down stack, capable of storing five items, which will be found to be more than enough for most purposes. I have called the five parts (a) Display - the item we see or work upon first, (b) X, (c) Y, (d) Z, and (e) Top. Follow the 3 x 4 example. We type 3 and press RETURN. 3 is entered into the Display register. Now we type 4 and press RETURN. The figure 4 appears in the Display register, but the 3 has moved up to the X register. Both are held separately. Typing * (for multiply) multiplies the bottom two registers together; 12 appears in the Display register, and 3 and 4 disappear.

So what, you ask? Well, we have looked at a pathetically simple example. The real value of RPN comes when we start encountering brackets - the more the merrier. So as not to take too long over it, let' stake another fairly simple example, 8 * ((4 - 2) + (5 - 1)). Working this conventionally, we find ourselves hopping backwards and forwards; we need to calculate 4 - 2 and store the result, then work 5 - 1, recover the sub-result (2) and add it to the other (4), making 6; then go back to the beginning to find the 8 and multiply, making 48.

In RPN, there is no hopping about. Type 8 (Return). Type 4 (Return). We cannot operate on these two yet, so we type the next, which is 2. We can operate on the 4 and 2, so we type ' -'(no need for RETURN). Type 5 (Return). Type 1, -, +, *. The job is done! Note that we worked from left to right on the numbers, ignoring brackets, except to ask ourselves if we could operate on the last two numbers. Put succinctly, the entire command line, with (R) short for RETURN, is: 8 (R) 4 (R) 2 - 5 (R) 1 - + *.

There is really only one basic rule to remember; an operator such as + or * operates on the last two numbers on the stack - the bottom pair, in this program. An operator pair, such as S (sine value) operates only on the bottom (Display) value.

Now I could spend a lot of time extolling the virtues of RPN and teaching the reader how to use it, but this is not a book on mathematics and I want to point out other features of this very useful and fascinating program.

In addition to the five-layer stack there is a Memory to which values can be assigned, added to, etc., but only from the Display register. Secondly, as you type commands - numbers or mnemonics - they appear in a growing ' CommandString' at the bottom of the display, which is useful as a record of what you have done. Finally, there is a fully implemented HELP function. Pressing H at any time will bring you a succinct description of any of the displayed functions.

So to the POP and ROTATE. POP is a method of discarding the contents of the Display register; it is simply thrown away and the rest of the stack is dropped one position. Perhaps you can guess at ROTATE: the stack is rotated one place. That is, the Display value is moved to the TOP position, and all the rest dropped one place.

Finally, it is worth pointing out that apart from its value in teaching and utilising RPN, this program has value in teaching the user what a stack is and how it works. As the computer itself uses stacks, this should be of great interest to us. I can recommend a study of this program before you start thinking about using a language such as FORTH, for example.

Variables

S$	String of spaces for register clearance purposes
M	Value held in Memory
T	Value help in TOP register
X, Y, Z	Values in these registers
D	Value held in DISPLAY register
X$	Command string already executed
C$	Current command
A%	ASCII value of C$
Q	Temporary holding variable
K$	Start of line for blue on yellow
L$	Start of line for red text

```
   10 ON ERROR GOTO 420
   20 S$=STRING$(12," ")
   30 Y$=CHR$131+CHR$157+CHR$132
   40 B$=CHR$132+CHR$157+CHR$131
   50 MODE7:M=0:T=0:Z=0:Y=0:X=0:D=0:X$="":PROCBOUNDS
   60 PROCDISPLAY
   70 REPEAT:PRINTTAB(0,20);B$;"YOUR COMMAND ";:INPUT C$:UNT
IL C$<>""
   80 IF X$="" X$=C$ ELSE X$=X$+" "+C$
   90 PRINTTAB(0,21);B$;"STRING: ";X$
  100 IF LEFT$(C$6,1)="." C$="0"+C$
  110 A%=ASC(LEFT$(C$,1))
  120 IF A%>57 OR A%<48 GOTO140
  130 C=VAL(C$):PROCUP:D=C:A%=ASC(RIGHT$(C$,1))
  140 IF A%=33 D=X:PROCDOWN
  150 IF A%=35 M=M+D
  160 IF A%=42 D=D*X:PROCDOWN
  170 IF A%=43 D=D+X:PROCDOWN
  180 IF A%=45 D=X-D:PROCDOWN
  190 IF A%=47 D=X/D:PROCDOWN
  200 IF A%=60 D=SQR(D)
  210 IF A%=62 D=D*D
  220 IF A%=64 D=10^D
  230 IF A%=65 D=ATN(D)
  240 IF A%=67 D=COS(D)
  250 IF A%=68 D=DEG(D)
  260 IF A%=69 PROCUP:D=2.71828183
  270 IF A%=70 PROCUP:D=M
  280 IF A%=72 PROCHELP:GOTO50
  290 IF A%=73 D=1/D
  300 IF A%=76 D=LOG(D)
  310 IF A%=77 M=D
  320 IF A%=79 Q=D:D=X:PROCDOWN:T=Q
  330 IF A%=80 PROCUP:D=PI
  340 IF A%=82 D=RAD(D)
  350 IF A%=83 D=SIN(D)
  360 IF A%=84 D=TAN(D)
  370 IF A%=88 Q=M:M=D:D=Q
  380 IF A%=94 D=X^D:PROCDOWN
  390 IF A%=124 D=-D
  400 GOTO60
  410
  420 IF ERR=17 GOTO50
  430 REPORT:PRINT"... PRESS RETURN":INPUT Q$:GOTO50
  440
  450 DEFPROCDISPLAY
  460 PRINTTAB(0,12);"MEMORY:";TAB(10);Y$;M;S$'"TOP:";TAB(1
      0);Y$;T;S$'"Z:";TAB(10);Y$;Z;S$
  470 PRINT"Y:";TAB(10);Y$;Y;S$'"X:";TAB(10);Y$;X;S$'"DISPLA
      Y:";TAB(10);Y$;D;S$
  480 PRINTTAB(0,20),STRING$(39," ")
  490 ENDPROC
  500
```

```
  510 DEFPROCBOUNDS
  520 PRINTY$;"     REVERSE-POLISH CALCULATOR"'
  530 PRINTCHR$129;"P - Pi       E - Const.e   S - Sine"'CHR
$129;"C - Cosine   T - Tangent   A - Arctan"'CHR$129;"D - De
g[Rad  R - Rad[Deg   L - N'l Log"
  540 PRINTCHR$129;"< - Root     > - Square    @ - 10^D"'CHR
$129;"O - Rotate   ! - Pop       M - To Mem"'CHR$129;"# - Ad
d to Mem      F - From Mem"'CHR$129;"X - Exchange Mem    | -
 Change sign"
  550 PRINTCHR$129;"I - Inverse         Esc - Master Clear"'
CHR$129;"H<cmd> - Help & Clear"
  560 ENDPROC
  570
  580 DEFPROCUP
  590 T=Z:Z=Y:Y=X:X=D:ENDPROC
  600
  610 DEFPROCDOWN
  620 X=Y:Y=Z:Z=T:T=0:ENDPROC
  630
  640 DEFPROCHELP
  650 CLS:PRINT'':IF LEN(C$)>1 S%=ASC(RIGHT$(C$,1)):GOTO680
  660 PRINT"HELP PAGE"'"_________"''
  670 INPUT "WHICH SYMBOL REQUIRES EXPLANATION",X$:S%=ASC(X$
):PRINT
  680 IF S%=33 PRINT"! = POP: The display value is discarded
 and the stack dropped one position.":GOTO930
  690 IF S%=35 PRINT"# = ADD  TO MEMORY: The display value i
s added to the memory value.":GOTO930
  700 IF S%=42 PRINT"* = MULTIPLY: The display value is"'"re
placed by the product of X and D."'"D and X are lost  and th
e rest of the    stack is dropped.":GOTO930
  710 IF S%=43 PRINT"+ = ADD: The display value is"'"replace
d by the sum of X and D."'"D and X are lost and the rest of
the    stack is dropped.":GOTO930
  720 IF S%=45 PRINT"- = SUBTRACT: The display value is"'"re
placed by the result of subtracting   D from X."'"D and X ar
e lost and the rest of the    stack is dropped.":GOTO930
  730 IF S%=47 PRINT"/ = DIVIDE: The display value is"'"repl
aced by the result of dividing X and D and the rest of the s
tack is dropped.":GOTO930
  740 IF S%=60 PRINT"< = ROOT: The display value is"'"replac
ed by its square root.":GOTO930
  750 IF S%=62 PRINT"> = SQUARE: The display value is"'"repl
aced by its square.":GOTO930
  760 IF S%=64 PRINT"@ = 10^D: The display value is"'"replac
ed by its square.":GOTO930
  770 IF S%=65 PRINT"A = ARCTAN: The display value is"'"repl
aced by its arctan vaue in"'"radians.  Used for finding an a
ngle"'"whose tangent is known.":GOTO930
  780 IF S%=67 PRINT"C = COSINE: The display value in"'"radi
ans is replaced by its cosine.":GOTO930
  790 IF S%=68 PRINT"D = RADIANS TO DEGREES: The radian"'"va
lue of the display is converted to    equivalent degrees.":G
OTO930
```

```
  800 IF S%<>69 GOTO820 ELSE PRINT"E = CONSTANT 'e': The sta
ck is moved"'"up and the constant 'e' (2.71828183)"'"is inse
rted at the bottom."'"TOP value is lost."''
  810 PRINT"NOTE: When included in a numeric"'"value, E is r
ead as scientific notation as in 3.4E6, whic gives 34000000"
:GOTO930
  820 IF S%=70 PRINT"F = FROM MEMORY: The stack is moved up
 and the contents of memory are copied   into the display po
sition."'"TOP value is lost.":GOTO930
  830 IF S%=73 PRINT"I = INVERSE: The display value is"'"rep
laced by its inverse, ie, 1/D":GOTO930
  840 IF S%=76 PRINT"L = NATURAL LOGARITHM (BASE e): The"'"d
isplay value is replaced by its"'"natural logarithm.":GOTO93
0
  850 IF S%=77 PRINT"M = TO MEMORY: The display value is"'"c
opied into the memory.  Original"'"memory is lost.":GOTO930
  860 IF S%=79 PRINT"O = ROTATE STACK: The displaay value"'"
is transferred to the top and the rest  of the stack is move
d down. Nothing is  lost.":GOTO930
  870 IF S%=80 PRINT"P = PI: The stack is moved up and the
 value Pi (3.14159265) is placed in the  display register."'
"TOP value is converted to radians.":GOTO930
  880 IF S%=82 PRINT"R = DEGREES TO RADIANS: The display"'"v
alue is converted to radians.":GOTO930
  890 IF S%=83 PRINT"S = SINE: The radian display value is
 converted to sines.":GOTO930
  900 IF S%=84 PRINT"T = TANGENT: The radian display value i
s converted to tangent.":GOTO930
  910 IF S%=88 PRINT"X = EXCHANGE MEMORY: The displa value
 and the memory value are exchanged.":GOTO930
  920 IF S%=124 PRINT"| = CHANGE SIGN: Positive values in th
e display register are made negative and  vice-versa."
  930 INPUT'"Press RETURN....",A$:ENDPROC
```