

LAB EXERCISE

RPN

Background:

1. Most calculators use infix notation, in which the operator is typed between the operands. You have grown up solving infix math expressions such as $2 + 3$ or $9 - 5$.
2. There are two other types of binary math notation systems that are called prefix and postfix. In prefix the expressions look like $+ 2 3$ or $- 9 5$. Notice that the operator comes before ("pre") the operands.
3. Postfix binary expressions are used on some calculators, as complex math expressions can be entered without the need for parentheses. Postfix math is also called reverse polish notation (RPN). These statements look like $2 3 +$ or $9 5 -$. The advantage of postfix (and prefix) is the elimination of parentheses to solve more complex problems.
4. Here is a comparison of infix versus postfix math expressions:

infix	postfix
$5 + ((7 + 9) * 2)$	$5 \ 7 \ 9 \ + \ 2 \ * \ +$

A postfix expression is solved in this manner. If a value is entered it is placed on a stack. If an operation is entered (+, -, *, /) the stack is popped twice and the operation is applied to those two numbers. The resulting answer is placed back on the stack. The expression

$5 \ 7 \ 9 \ + \ 2 \ * \ +$ is solved in this order:
 $5 \ 16 \ 2 \ * \ +$
 $5 \ 32 \ +$
39

Notice that postfix expressions are simply solved left to right and that parentheses are not needed.

5. The following special cases must be addressed. If the problem to be solved is $7 - 5$ (infix), the problem is entered on an RPN calculator in this order:

7 (enter)
5 (enter)
-

This causes the stack to be popped twice and the correct math to be solved is $7 - 5$ -, which equals 2.

A similar ordering issue surrounds the (/) operator. The infix problem $9 / 2$ is entered on an RPN calculator as:

9
2
/ which returns 4.

Assignment:

1. Write a program to implement a simple RPN calculator that processes only this type of input:
single-digit integers
the four integer math operations: +, -, *, /
2. The user will type in single character input until 'Q' or 'q' is entered. As described above, if a digit is entered, the number is placed on the stack. If an operation is entered, pop two values off the stack, do the math, then return the answer back onto the stack. As keyboard input is entered the program should keep track of all keystrokes entered to be replayed after 'Q' or 'q' is entered.
3. When 'Q' or 'q' is entered, the program will print out the entire problem and the answer. A queue and a stack would be a good thing to use in this lab.

Instructions:

1. Enter the following 5 postfix problems:

```
9 5 -  
7 3 * 6 /  
8 4 7 + -  
7 3 9 4 5 * + - -  
8 4 3 * * 6 4 2 - + +
```

2. Your run output should look something like this:

```
9 5 - = 4  
7 3 * 6 / = 3  
8 4 7 + - = -3  
7 3 9 4 5 * + - - = 33  
8 4 3 * * 6 4 2 - + + = 104
```