Try at home
Stack Examples

1. Convert the following expression from infix to post-fix, using a stack. Show the state of the output and the stack after reading each token.

   10 + (8 - 27 / 3) * 7 - 5 * (4 + 2)

   Answers should be: 10 8 27 3 / - 7 * + 5 4 2 + * -

Move	Current token	Stack (grows toward left)	Output
1	10	empty	10
2	+	+	10
3	(	(+	10
4	8	(+	10 8
5	-	-(+	10 8
6	27	-(+	10 8 27
7	/	/-(+	10 8 27
8	3	/-(+	10 8 27 3
9	)	+	10 8 27 3 / -
10	*	*+	10 8 27 3 / -
11	7	*+	10 8 27 3 / - 7 *
12	-	-	10 8 27 3 / - 7 * +
13	5	-	10 8 27 3 / - 7 * + 5
14	*	*-	10 8 27 3 / - 7 * + 5
15	(	(*-	10 8 27 3 / - 7 * + 5
16	4	(*-	10 8 27 3 / - 7 * + 5 4
17	+	+(*-	10 8 27 3 / - 7 * + 5 4
18	2	+(*-	10 8 27 3 / - 7 * + 5 4 2
19	)	*-	10 8 27 3 / - 7 * + 5 4 2 +
20		empty	10 8 27 3 / - 7 * + 5 4 2 + * -

Notes:
In this table, the stack grows toward the left. Thus, the top of the stack is the leftmost symbol.

2. Assuming the above post-fix expression, compute the result using a stack. Show the result of the stack after reading each token.

Answer should be -27

Move	Current Token	Stack (grows toward left)
1	10	10
2	8	8 10
3	27	27 8 10
4	3	3 27 8 10
5	/	9 8 10
6	-	-1 10
7	7	7 -1 10
8	*	-7 10
9	+	3
10	5	5 3
11	4	4 5 3
12	2	2 4 5 3
13	+	6 5 3
14	*	30 3
15	-	-27

3. In pseudo code add two infinitely long integers using stacks

```
 main()
{
        operand1 //string
        operand2// string
        addLong(operand1, operand2)
}

add(operand1, operand2) //non-void function
{
        #instantiate stack
        stack stack1, stack2

        #assuming we have these function implemented already
        for i in operand1
                stack1.push(i) //append to top of stack
        for i in operand2
                stack2.push(i) //append to top of stack

        string display
        int carryOver

        for i in stack1
                temp1 = stack1.top() #assuming we converted back to int
                temp2 = stack2.top() #assuming we converted back to int
                temp3 = temp1 + temp2 + carryOver
                if temp3 > 10
                        carryOver = temp3[0] #take
                        display += temp3[1] #assuming we converted to char
                  else
                        display += temp3//assuming we converted to char
                        carryOver = 0

                stack1.pop()
                stack2.pop()

        int i = 0
        size = display.size
        while size > i #print string in reverse order
                cout << display.at(size)
                size--
}
```