



Challenge 6: Evaluate Postfix Expression Using a Stack

Let's try to compute postfix mathematical expressions using stacks!

We'll cover the following



- Problem Statement
 - Input
 - Output
 - Sample Input
 - Sample Output
- Coding Exercise

Problem Statement#

The usual convention followed in mathematics is the **infix expression**. Operators like **+** and ***** appear between the two numbers involved in the calculation:

```
6 + 3 * 8 - 4
```

Another convention is the **postfix expression** where the operators appear after the two numbers involved in the expression. In postfix, the expression written above will be presented as:

```
6 3 8 * + 4 -
```





The two digits preceding an operator will be used with that operator.

1. From the first block of digits **6 3 8**, we pick the last two which are **3** and **8**.
2. Reading the operators from left to right, the first one is *****. The expression now becomes **3 * 8**
3. The next number is **6** while the next operator is **+**, so we have **6 + 3 * 8**.
4. The value of this expression is followed by **4**, which is right before **-**. Hence we have **6 + 3 * 8 - 4**.

Implement a function called `evaluatePostFix()` that will compute a postfix expression given to it as a string.

Input#

A string containing a postfix mathematic expression. Each digit is considered to be a separate number, i.e., there are no double digit numbers.

Output#

A result of the given postfix expression.

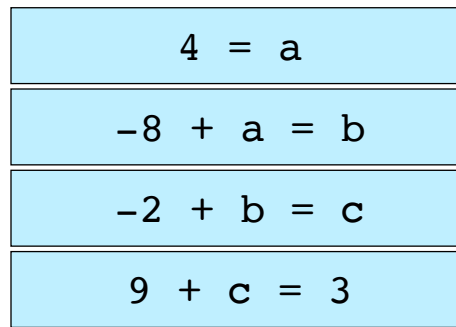
Sample Input#

```
exp = "921 * - 8 - 4 +" # 9 - 2 * 1 - 8 + 4
```

Sample Output#

```
3
```



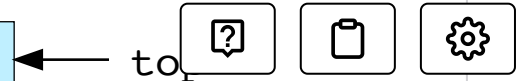


4 = a

-8 + a = b

-2 + b = c

9 + c = 3



Coding Exercise

Take a close look and design a step-by-step algorithm first before jumping on to the implementation. This problem is designed for your practice, so try to solve it on your own first.

If you get stuck, you can always refer to the solution review in the next lesson.

Good luck!

main.py

Stack.py

```
from Stack import MyStack

def evaluate_post_fix(exp):
    # Write your code here
    my_stack = MyStack()

    for char in exp:
        if char.isnumeric():
            my_stack.push(int(char))
        else:
            second = my_stack.pop()
            first = my_stack.pop()
            if char == "+":
                my_stack.push(first + second)
            elif char == "-":
                my_stack.push(first - second)
```

```
elif char == "/":  
    my_stack.push(first / second)  
elif char == "*":  
    my_stack.push(first * second)  
  
return my_stack.pop()
```



Interviewing soon? We've partnered with Hired so that companies apply to you instead of you applying to them. [See how](#) ⓘ

[← Back](#)[Next →](#)[Solution Review: Sort Values in Stack](#)[Solution Review: Evaluate Postfix Expr...](#)

Completed

[Report an Issue](#)