150. Evaluate Reverse Polish Notation

Created	@March 2, 2022 9:51 PM
Difficulty	Medium
LC Url	https://leetcode.com/problems/evaluate-reverse-polish-notation/
Importance	
Tag	NEET Stack
Video	https://www.youtube.com/watch?v=iu0082c4HDE

Evaluate the value of an arithmetic expression in Reverse Polish Notation.

Valid operators are `+`, `-`, `*`, and `/`. Each operand may be an integer or another expression.

Note that division between two integers should truncate toward zero.

It is guaranteed that the given RPN expression is always valid. That means the expression would always evaluate to a result, and there will not be any division by zero operation.

Example 1:

```
Input: tokens = ["2","1","+","3","*"]
Output: 9
Explanation: ((2 + 1) * 3) = 9
```

Example 2:

```
Input: tokens = ["4","13","5","/","+"]
Output: 6
Explanation: (4 + (13 / 5)) = 6
```

Example 3:

```
Input: tokens = ["10","6","9","3","+","-11","*","/","*","17","+","5","+"]
Output: 22
Explanation: ((10 * (6 / ((9 + 3) * -11))) + 17) + 5
= ((10 * (6 / (12 * -11))) + 17) + 5
= ((10 * (6 / -132)) + 17) + 5
= ((10 * 0) + 17) + 5
= (0 + 17) + 5
= 17 + 5
= 22
```

Constraints:

- `1 <= tokens.length <= 10 4`
- `tokens[i]` is either an operator: `"+"`, `"-"`, `"*"`, or `"/"`, or an integer in the range `[-200, 200]`.

Solution

```
class Solution:
    def evalRPN(self, tokens: List[str]) -> int:
        stack = []
        for c in tokens:
            if c == '+':
                stack.append(stack.pop() + stack.pop())
            elif c == '-':
                post_c, pre_c = stack.pop(), stack.pop()
                stack.append(pre_c - post_c)
            elif c == '*':
                stack.append(stack.pop() * stack.pop())
            elif c == '/':
                post_c, pre_c = stack.pop(), stack.pop()
                stack.append(int(pre_c / post_c))
            else:
                stack.append(int(c))
        return stack[0]
```