Description Solution Discuss (999+) Submissions

682. Baseball Game

Easy 1168 1431 Add to List Share

You are keeping score for a baseball game with strange rules. The game consists of several rounds, where the scores of past rounds may affect future rounds' scores.

At the beginning of the game, you start with an empty record. You are given a list of strings `ops`, where `ops[i]` is the `i`th operation you must apply to the record and is one of the following:

1. An integer `x` - Record a new score of `x`.
2. `"+"` - Record a new score that is the sum of the previous two scores. It is guaranteed there will always be two previous scores.
3. `"D"` - Record a new score that is double the previous score. It is guaranteed there will always be a previous score.
4. `"C"` - Invalidate the previous score, removing it from the record. It is guaranteed there will always be a previous score.

Return *the sum of all the scores on the record*.

Example 1:

```
Input: ops = ["5","2","C","D","+"]
Output: 30
Explanation:
"5" - Add 5 to the record, record is now [5].
"2" - Add 2 to the record, record is now [5, 2].
"C" - Invalidate and remove the previous score, record is now [5].
"D" - Add 2 * 5 = 10 to the record, record is now [5, 10].
"+" - Add 5 + 10 = 15 to the record, record is now [5, 10, 15].
The total sum is 5 + 10 + 15 = 30.
```

Example 2:

```
Input: ops = ["5","-2","4","C","D","9","+","+"]
Output: 27
Explanation:
"5" - Add 5 to the record, record is now [5].
"-2" - Add -2 to the record, record is now [5, -2].
"4" - Add 4 to the record, record is now [5, -2, 4].
```

Java Autocomplete

```java
class Solution {
    public int calPoints(String[] ops) {

        Stack<Integer> stack = new Stack<>();

        for(String str : ops){

            if(str.equals("+")){
                int first = stack.pop();
                int second = stack.pop();

                stack.push(second);
                stack.push(first);
                stack.push(first + second);

            }else if(str.equals("D")){
                stack.push(2 * stack.peek());
            }else if(str.equals("C")){
                stack.pop();
            }else{
                stack.add(Integer.valueOf(str));
            }
        }

        int result = 0;

        while(!stack.isEmpty()){
            result += stack.pop();
        }

        return result;
    }
}
```