682 Baseball Game

2018年4月23日 19:15

You're now a baseball game point recorder.
Given a list of strings, each string can be one of the 4 following types:

1. Integer (one round's score): Directly represents the number of points you get in this round.
2. "+" (one round's score): Represents that the points you get in this round are the sum of the last two valid round's points.
3. "D" (one round's score): Represents that the points you get in this round are the doubled data of the last valid round's points.
4. "C" (an operation, which isn't a round's score): Represents the last valid round's points you get were invalid and should be removed.

Each round's operation is permanent and could have an impact on the round before and the round after.
You need to return the sum of the points you could get in all the rounds.

Example 1:
Input: ["5","2","C","D","+"]
Output: 30
Explanation:
Round 1: You could get 5 points. The sum is: 5.
Round 2: You could get 2 points. The sum is: 7.
Operation 1: The round 2's data was invalid. The sum is: 5.
Round 3: You could get 10 points (the round 2's data has been removed). The sum is: 15.
Round 4: You could get 5 + 10 = 15 points. The sum is: 30.

Example 2:
Input: ["5","-2","4","C","D","9","+","+"]
Output: 27
Explanation:
Round 1: You could get 5 points. The sum is: 5.
Round 2: You could get -2 points. The sum is: 3.
Round 3: You could get 4 points. The sum is: 7.
Operation 1: The round 3's data is invalid. The sum is: 3.
Round 4: You could get -4 points (the round 3's data has been removed). The sum is: -1.
Round 5: You could get 9 points. The sum is: 8.
Round 6: You could get -4 + 9 = 5 points. The sum is 13.
Round 7: You could get 9 + 5 = 14 points. The sum is 27.

Note:

- The size of the input list will be between 1 and 1000.
- Every integer represented in the list will be between -30000 and 30000.

来自 <https://leetcode.com/problems/baseball-game/description/>

你现在是棒球比赛记录员。
给定一个字符串列表，每个字符串可以是以下四种类型之一：
1.整数（一轮的得分）：直接表示您在本轮中获得的积分数。
2. "+"（一轮的得分）：表示本轮获得的得分是前两轮有效 回合得分的总和。
3. "D"（一轮的得分）：表示本轮获得的得分是前一轮有效 回合得分的两倍。
4. "C"（一个操作，这不是一个回合的分数）：表示您获得的最后一个有效 回合的分数是无效的，应该被移除。

每一轮的操作都是永久性的，可能会对前一轮和后一轮产生影响。
你需要返回你在所有回合中得分的总和。

示例 1:

输入: ["5","2","C","D","+"]

输出: 30

解释:

第1轮：你可以得到5分。总和是：5。

第2轮：你可以得到2分。总和是：7。

操作1：第2轮的数据无效。总和是：5。

第3轮：你可以得到10分（第2轮的数据已被删除）。总数是：15。

第4轮：你可以得到5 + 10 = 15分。总数是：30。

示例 2:

输入: ["5","-2","4","C","D","9","+","+"]

输出: 27

解释:

第1轮：你可以得到5分。总和是：5。

第2轮：你可以得到-2分。总数是：3。

第3轮：你可以得到4分。总和是：7。

操作1：第3轮的数据无效。总数是：3。

第4轮：你可以得到-4分（第三轮的数据已被删除）。总和是：-1。

第5轮：你可以得到9分。总数是：8。

第6轮：你可以得到-4 + 9 = 5分。总数是13。

第7轮：你可以得到9 + 5 = 14分。总数是27。

注意:

- 输入列表的大小将介于1和1000之间。
- 列表中的每个整数都将介于-30000和30000之间。

Solution for Python3:

```python
class Solution:
    def calPoints(self, ops):
        """
        :type ops: List[str]
        :rtype: int
        """
        res = []
        for c in ops:
            if c == 'C' and len(res) > 0:
                res.pop()
            elif c == 'D':
                res.append(sum(res[-1:]) * 2)
            elif c == '+':
                res.append(sum(res[-2:]))
            else:
                res.append(int(c))
        return sum(res)
```

Solution for C++:

```
class Solution {
public:
    int calPoints(vector<string>& ops) {
        stack<int> sta;
        for (string c : ops) {
            if (c == "+") {
                int top = sta.top();
                sta.pop();
                int newtop = top + sta.top();
                sta.push(top);
                sta.push(newtop);
            } else if (c == "C") {
                sta.pop();
            } else if (c == "D") {
                sta.push(2 * sta.top());
            } else {
                sta.push(stoi(c));
            }
        }
        int ans = 0, p = sta.size();
        while (p-- > 0) {
            ans += sta.top();
            sta.pop();
        }
        return ans;
    }
};

class Solution2 {
public:
    int calPoints(vector<string>& ops) {
        int sum = 0, score = 0;
        vector<int> v;
        for (string op : ops) {
            if (op == "C") {
                sum -= v.back();
                v.pop_back();
                continue;
            } else if (op == "D") {
                sum += score = v.back() * 2;
            } else if (op == "+") {
                sum += score = v[v.size() - 1] + v[v.size() - 2];
            } else {
                sum += score = stoi(op);
            }
            v.push_back(score);
        }
        return sum;
    }
};
```