Stack – LIFO/FIFO

-

1. Problem Statement

https://leetcode.com/problems/min-stack/

Design a stack that supports push, pop, top, and retrieving the minimum element in constant time.

- push(x) -- Push element x onto stack.
- pop() -- Removes the element on top of the stack.
- top() -- Get the top element.
- getMin() -- Retrieve the minimum element in the stack.

```
Example 1:

Input

["MinStack","push","push","push","getMin","pop","top","getMin"]

[[],[-2],[0],[-3],[],[],[],[]]

Output

[null,null,null,null,-3,null,0,-2]

Explanation

MinStack minStack = new MinStack();

minStack.push(-2);

minStack.push(0);

minStack.push(-3);

minStack.getMin(); // return -3
```

2. Problem Statement

https://leetcode.com/problems/baseball-game/

You're now a baseball game point recorder.

Given a list of strings, each string can be one of the 4 following types:

Integer (one round's score): Directly represents the number of points you get in this round.

"+" (one round's score): Represents that the points you get in this round are the sum of the last two valid round's points.

"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.

"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.

3. Problem Statement

https://leetcode.com/problems/decode-string/

Given an encoded string, return its decoded string.

The encoding rule is: k[encoded_string], where the encoded_string inside the square brackets is being repeated exactly k times. Note that k is guaranteed to be a positive integer.

You may assume that the input string is always valid; No extra white spaces, square brackets are well-formed, etc.

Furthermore, you may assume that the original data does not contain any digits and that digits are only for those repeat numbers, k. For example, there won't be input like 3a or 2[4].

```
Example 1:

Input: s = "3[a]2[bc]"

Output: "aaabcbc"



Example 2:

Input: s = "3[a2[c]]"

Output: "accaccacc"
```

4. Problem Statement

https://leetcode.com/problems/next-greater-element-i/

You are given two arrays (without duplicates) nums1 and nums2 where nums1’s elements are subset of nums2. Find all the next greater numbers for nums1's elements in the corresponding places of nums2.

The Next Greater Number of a number x in nums1 is the first greater number to its right in nums2. If it does not exist, output -1 for this number.

Example 1:

```
Input: nums1 = [4,1,2], nums2 = [1,3,4,2].

Output: [-1,3,-1]

Explanation:
```

```
For number 4 in the first array, you cannot find the next greater number for it in the second array, so output -1.

For number 1 in the first array, the next greater number for it in the second array is 3.

For number 2 in the first array, there is no next greater number for it in the second array, so output -1.
```

5. Problem Statement

Implement N stack using single array.

https://www.geeksforgeeks.org/efficiently-implement-k-stacks-single-array/