Decode String

Given an encoded string, return it's 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]`.

Examples:

```
s = "3[a]2[bc]", return "aaabcbc".
s = "3[a2[c]]", return "accaccacc".
s = "2[abc]3[cd]ef", return "abcabccdcdcdef".
```

Subscribe to see which companies asked this question

Solution 1

```
class Solution(object):
    def decodeString(self, s):
        stack = []
        stack.append(["", 1])
        num = ""
        for ch in s:
            if ch.isdigit():
              num += ch
            elif ch == '[':
                stack.append(["", int(num)])
                num = ""
            elif ch == ']':
                st, k = stack.pop()
                stack[-1][0] += st*k
            else:
                stack[-1][0] += ch
        return stack[0][0]
```

written by ayushpatwari original link here

Solution 2

```
class Solution {
public:
    string decodeString(string s, int& i) {
        string res;

        while (i < s.length() && s[i] != ']') {
            if (!isdigit(s[i]))
                res += s[i++];
            else {
                int n = 0;
                while (i < s.length() && isdigit(s[i]))
                    n = n * 10 + s[i++] - '0';

                i++; // '['
                string t = decodeString(s, i);
                i++; // ']'

                while (n-- > 0)
                    res += t;
            }
        }

        return res;
    }

    string decodeString(string s) {
        int i = 0;
        return decodeString(s, i);
    }
};
```

written by bluedawnstar original link here

Solution 3

```java
public class Solution {
    public String decodeString(String s) {
        String res = "";
        Stack<Integer> countStack = new Stack<>();
        Stack<String> resStack = new Stack<>();
        int idx = 0;
        while (idx < s.length()) {
            if (Character.isDigit(s.charAt(idx))) {
                int count = 0;
                while (Character.isDigit(s.charAt(idx))) {
                    count = 10 * count + (s.charAt(idx) - '0');
                    idx++;
                }
                countStack.push(count);
            }
            else if (s.charAt(idx) == '[') {
                resStack.push(res);
                res = "";
                idx++;
            }
            else if (s.charAt(idx) == ']') {
                StringBuilder temp = new StringBuilder (resStack.pop());
                int repeatTimes = countStack.pop();
                for (int i = 0; i < repeatTimes; i++) {
                    temp.append(res);
                }
                res = temp.toString();
                idx++;
            }
            else {
                res += s.charAt(idx++);
            }
        }
        return res;
    }
}
```

written by sampsonchan original link here