Decode the string

An encoded string (s) is given, the task is to decode it. The pattern in which the strings were encoded were as follows

Input : 1[b] **Output** : b

Input : 3[b2[ca]] **Output** : bcacabcacabcaca

ALGORITHM :

1. Create two stacks , one to store integral values , and the other one to store the characters / alphabets.
2. Traverse the string from left to right.
3. Whenever we encounter any number, push it into the integer stack and in case of any alphabet (a to z) or open bracket ' [', push it onto the character stack.
4. Whenever any close bracket '] ' is encountered , pop the character from the character stack until an open bracket ' [' is not found in the character stack. Also, pop the top element from the integer stack , say n. Now make a string repeating the popped character n times. Now, push all characters of the string in the stack.

CODE :

```
1. #include<bits/stdc++.h>
2. #include <iostream>
3. using namespace std;
4.
5. string myfun(string s)
6. {
```

```
7.    stack<int>it;
8.    stack<char>ch;
9.    string temp = "", result = "";
10.      for(int i = 0; i < s.length() ; i++)
11.      {
12.          int count = 0 ;
13.          if(s[i] >= '0' && s[i] <='9')
14.          {
15.              while(s[i] >= '0' && s[i] <= '9')
16.              {
17.                  count = count*10 + s[i] - '0';
18.                  i++;
19.              }
20.              i--;
21.              it.push(count);
22.          }
23.          else if(s[i] == '[')
24.          {
25.              if(s[i-1] >= '0' && s[i-1] <='9')
26.                  ch.push(s[i]);
27.          }
28.          else if(s[i] == ']')
29.          {
30.              temp = "";
31.              count = 0;
32.
33.              if (!it.empty())
34.              {
35.                  count = it.top();
36.                  it.pop();
37.              }
38.
39.              while (!ch.empty() && ch.top()!='[' )
40.              {
41.                  temp = ch.top() + temp;
42.                  ch.pop();
43.              }
```

```
44.
45.                 if (!ch.empty() && ch.top() == '[')
46.                     ch.pop();
47.
48.                 for (int j = 0; j < count; j++)
49.                     result = result + temp;
50.                 for (int j = 0; j < result.length(); j++)
51.                     ch.push(result[j]);
52.
53.                 result = "";
54.             }
55.             else
56.                 ch.push(s[i]);
57.         }
58.          while (!ch.empty())
59.         {
60.             result =ch.top() + result;
61.             ch.pop();
62.         }
63.         return result;
64.     }
65.     int main()
66.     {
67.         int t;
68.         cin >> t;
69.         while(t--)
70.         {
71.             string st;
72.             string s;
73.             cin >> s;
74.             st = myfun(s);
75.             cout << st<< endl;
76.         }
77.        return 0;
78.     }
79.
```

1. https://practice.geeksforgeeks.org/problems/string-manipulation/0
2. https://practice.geeksforgeeks.org/problems/evaluation-of-postfix-expression/0
3. https://practice.geeksforgeeks.org/problems/count-the-reversals/0
4. https://practice.geeksforgeeks.org/problems/easy-string/0
5. https://practice.geeksforgeeks.org/problems/maximum-difference-1587115620/1