DSA Question

Name : Omkar Kishor Patil
Roll call: BI39

Q1. To reduce the size of this string using mathematical logic given as in the example below :
Input : aabbbbeeeeffggg
Output: a2b4e4f2g3
Input : abbccccc
Output: ab2c5

Ans:
```
 #include <iostream>
#include <string>
using namespace std;

int main() {
    string s;
    cin >> s;

    string ans = "";
    int count = 1;

    for (int i = 1; i < s.length(); i++) {
        if (s[i] == s[i - 1]) {
            count++;
        } else {
            ans += s[i - 1];
            ans += to_string(count);

            count = 1;
        }
    }

    ans += s[s.length() - 1];
    ans += to_string(count);


    cout << ans << endl;

    return 0;
}
```

Q6. Given an integer array, find the maximum product of two integers in it.
E.g. For example, consider array {-10, -3, 5, 6, -2}. The maximum product is the (-10, -3) or (5, 6) pair.

```
#include <iostream>
#include <vector>
using namespace std;

int main() {
    int n;
    cin >> n;
    vector<int> arr(n);


    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int maxProduct = INT_MIN;
    int num1 = 0, num2 = 0;


    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            int prod = arr[i] * arr[j];
            if (prod > maxProduct) {
                maxProduct = prod;
                num1 = arr[i];
                num2 = arr[j];
            }
        }
    }

    cout << "Pair with max product: " << num1 << " " << num2 << endl;


    return 0;
}
```

Q7. Given an array of distinct integers, replace every element with the least greater element on its right or with -1 if there are no greater elements.
For example,
Input: { 10, 100, 93, 32, 35, 65, 80, 90, 94, 6 }
Output: { 32, -1, 94, 35, 65, 80, 90, 94, -1, -1 }

```
#include <iostream>
#include <vector>
#include <set>
using namespace std;

int main() {
    int n;
    cin>>n;
    vector<int> arr(n);
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }

    set<int> bst;
    vector<int> result(n, -1);


    for (int i = n - 1; i >= 0; i--) {

        auto it = bst.upper_bound(arr[i]);
        if (it != bst.end()) {
            result[i] = *it;
        }
        bst.insert(arr[i]);
    }


    for (int i = 0; i < n; i++) {
        cout << result[i] << " ";
    }

    return 0;
}
```

Q8. A lost-and-found counter logs items being placed and returned.
- P item → place an item on the counter
- R → remove last placed item

Find which item remains on top at the end.
Input:
6

P Wallet
P Phone
R
P Bag

R
P Keys
Output:
Top item: Keys

```
#include <iostream>
#include <vector>
#include <set>
#include<stack>
using namespace std;

int main() {
    int n;
    cin>>n;
    char ch;
    stack<string> st;

    for(int i=0;i<n;i++){
        cin>>ch;
        if(ch=='P'){
            string s;
            cin>>s;
            st.push(s);

        }
        if(ch=='R'){
            st.pop();
        }
    }
    cout<<st.top();


}
```

Q10. Topic: Queue / Array
Scenario:
A café receives orders in sequence. Each order has a preparation time. The café can handle only one order at a time. You need to find the total time taken to complete all orders, and which order finishes last.
Input Format:
● n → number of orders
● array of n integers → preparation time of each order
Output:
● Total preparation time
● Index of last order finished

Example:
Input:
5
2 3 1 4 2
Output:
Total time = 12
Last order index = 4

```
#include <iostream>
#include <vector>
#include <set>
#include<stack>
using namespace std;

int main() {
    int n;
    cin>>n;
    vector<int> order(n);
    for(int i=0;i<n;i++){
        cin>>order[i];
    }
    int sum=0;
    for(int i=0;i<n;i++){
        sum=sum+order[i];
    }
    cout<<"Total time ="<<" "<<sum<<endl;
    cout<<"Last order index= "<<" "<<n-1;

    return 0;


}
```