

# String Challenges

## Challenge 1

### UpperCase-LowerCase interconversion

Given a string *s* with both uppercase and lowercase latin characters ('a' - 'z'). Your task is convert whole string into

1. Lower Case
2. Upper Case

Base idea: 'a' - 'A' = 32

1. Lowercase to UpperCase

Approach

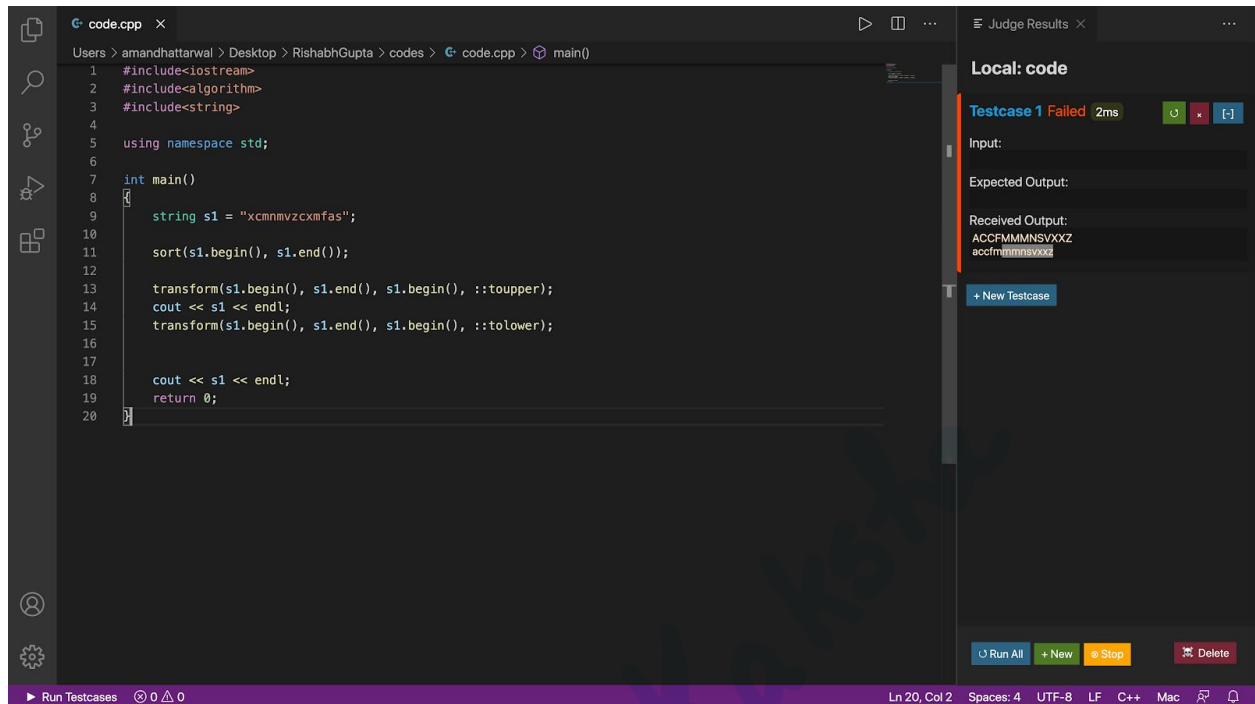
1. Iterate over the string *s* and if *s*[*i*] is a lower case character, then update  
 $s[i] -= 32$

2. UpperCase to LowerCase

Approach

1. Iterate over the string *s* and if *s*[*i*] is a upper case character, then update  
 $s[i] += 32$

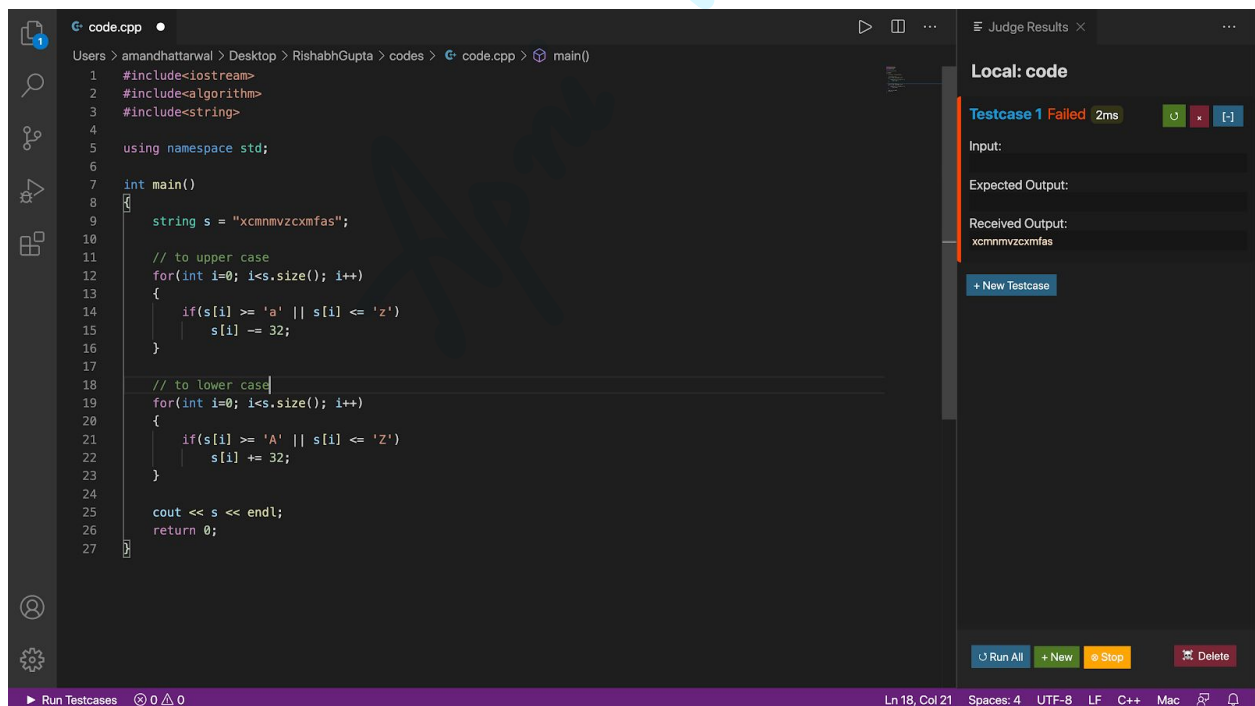
Code:



The screenshot shows a C++ IDE with a file named `code.cpp`. The code in the editor is as follows:

```
1 #include<iostream>
2 #include<algorithm>
3 #include<string>
4
5 using namespace std;
6
7 int main()
8 {
9     string s1 = "xcmmmvzcxmfas";
10
11     sort(s1.begin(), s1.end());
12
13     transform(s1.begin(), s1.end(), s1.begin(), ::toupper);
14     cout << s1 << endl;
15     transform(s1.begin(), s1.end(), s1.begin(), ::tolower);
16
17     cout << s1 << endl;
18     return 0;
19 }
```

The right sidebar shows the 'Judge Results' for 'Local: code'. It indicates 'Testcase 1 Failed' with a time of '2ms'. The 'Input' is empty. The 'Expected Output' is empty. The 'Received Output' is 'ACCFMMMNSVXXZ' followed by 'accfmmmvzcxmfas' on a new line. Below the output, there is a '+ New Testcase' button. At the bottom of the sidebar, there are buttons for 'Run All', '+ New', 'Stop', and 'Delete'.



The screenshot shows a C++ IDE with a file named `code.cpp`. The code in the editor is as follows:

```
1 #include<iostream>
2 #include<algorithm>
3 #include<string>
4
5 using namespace std;
6
7 int main()
8 {
9     string s = "xcmmmvzcxmfas";
10
11     // to upper case
12     for(int i=0; i<s.size(); i++)
13     {
14         if(s[i] >= 'a' || s[i] <= 'z')
15             s[i] -= 32;
16     }
17
18     // to lower case
19     for(int i=0; i<s.size(); i++)
20     {
21         if(s[i] >= 'A' || s[i] <= 'Z')
22             s[i] += 32;
23     }
24
25     cout << s << endl;
26     return 0;
27 }
```

The right sidebar shows the 'Judge Results' for 'Local: code'. It indicates 'Testcase 1 Failed' with a time of '2ms'. The 'Input' is empty. The 'Expected Output' is empty. The 'Received Output' is 'xcmmmvzcxmfas'. Below the output, there is a '+ New Testcase' button. At the bottom of the sidebar, there are buttons for 'Run All', '+ New', 'Stop', and 'Delete'.

## Challenge 2

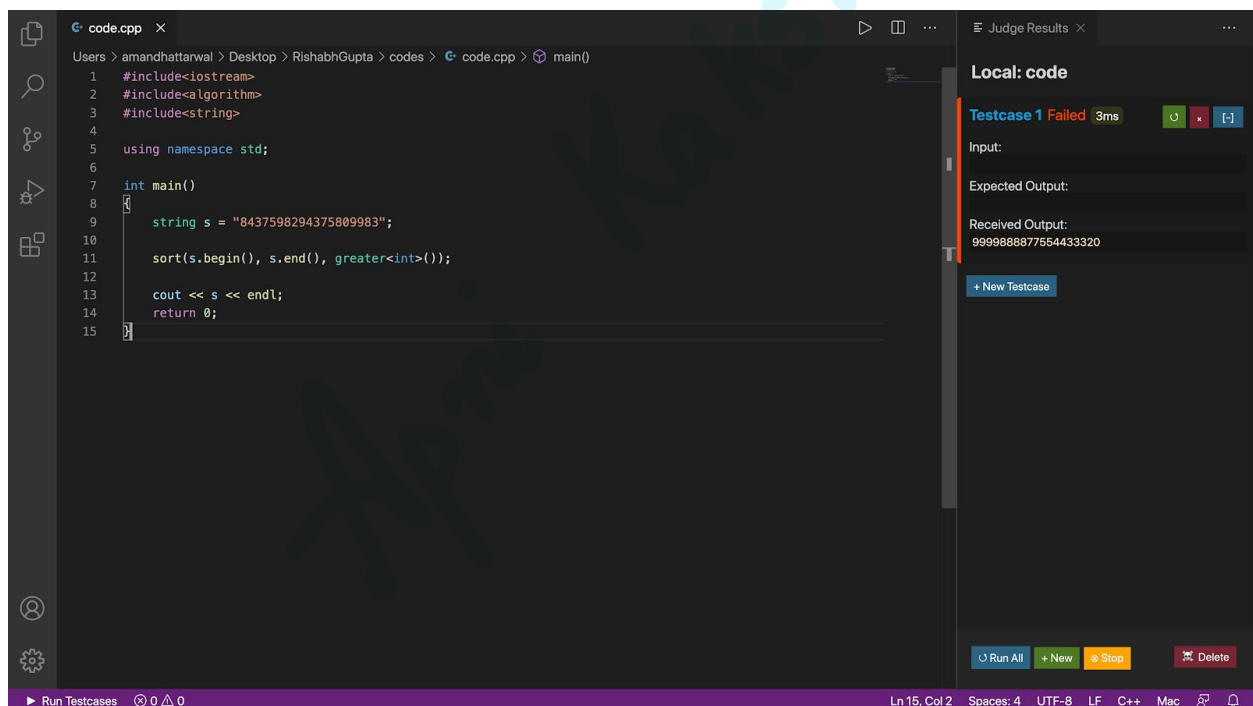
### Form the biggest number

Given a string of integers, our task is to form the biggest number out of those numbers in the string.

Approach:

Sort the string in descending order using inbuilt sort function.

Code



```
1 #include<iostream>
2 #include<algorithm>
3 #include<string>
4
5 using namespace std;
6
7 int main()
8 {
9     string s = "8437598294375809983";
10
11     sort(s.begin(), s.end(), greater<int>());
12
13     cout << s << endl;
14     return 0;
15 }
```

Local: code

Testcase 1 Failed 3ms

Input:

Expected Output:

Received Output:  
999988887755443320

+ New Testcase

Run All + New Stop Delete

Run Testcases 0 0 0 Ln 15, Col 2 Spaces: 4 UTF-8 LF C++ Mac

### Challenge 3

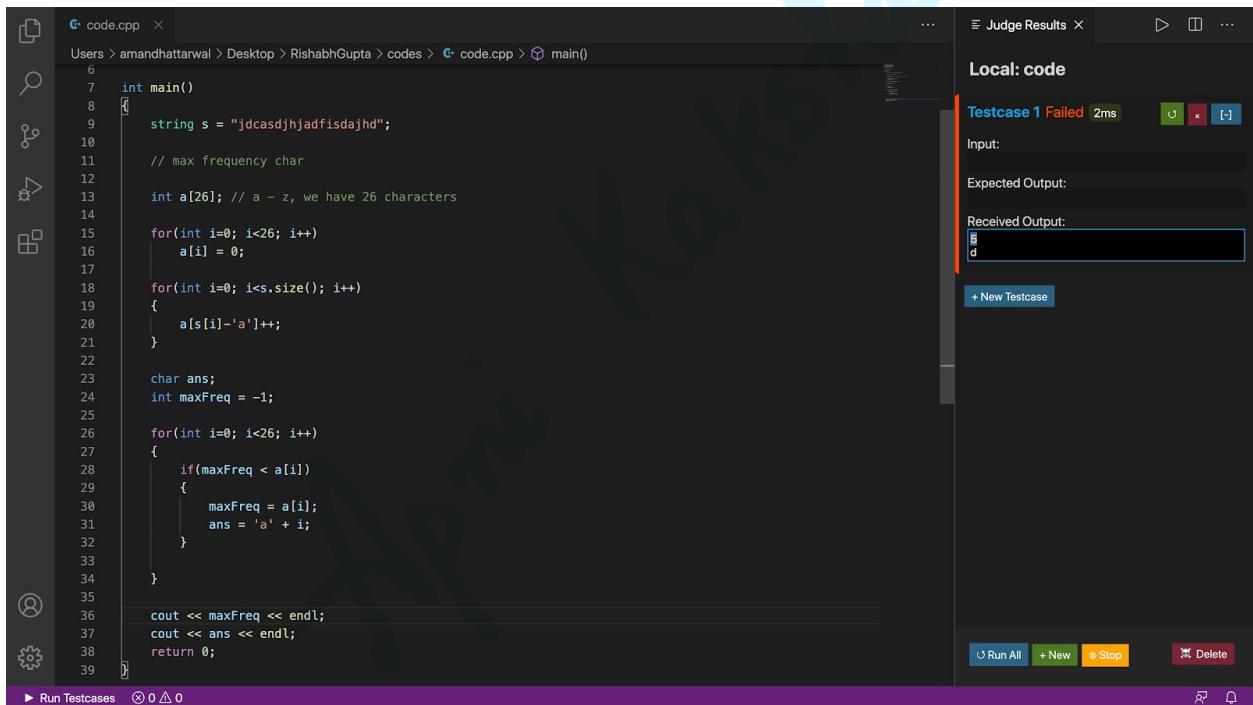
#### Max Frequency

Given a string *s* of latin characters, your task is to output the character which has maximum frequency.

Approach:

Maintain frequency of elements in a separate array and iterate over the array and find the maximum frequency character.

Code



```
6
7 int main()
8 {
9     string s = "jdcasdjhjadfisdaajhd";
10
11     // max frequency char
12
13     int a[26]; // a - z, we have 26 characters
14
15     for(int i=0; i<26; i++)
16         a[i] = 0;
17
18     for(int i=0; i<s.size(); i++)
19     {
20         a[s[i]-'a']++;
21     }
22
23     char ans;
24     int maxFreq = -1;
25
26     for(int i=0; i<26; i++)
27     {
28         if(maxFreq < a[i])
29         {
30             maxFreq = a[i];
31             ans = 'a' + i;
32         }
33     }
34
35     cout << maxFreq << endl;
36     cout << ans << endl;
37     return 0;
38 }
39
```

**Local: code**

Testcase 1 Failed 2ms

Input:

Expected Output:

Received Output:

s  
d

+ New Testcase

Run All + New Stop Delete

Run Testcases 0 0 0

## Challenge 4 (Additional Question)

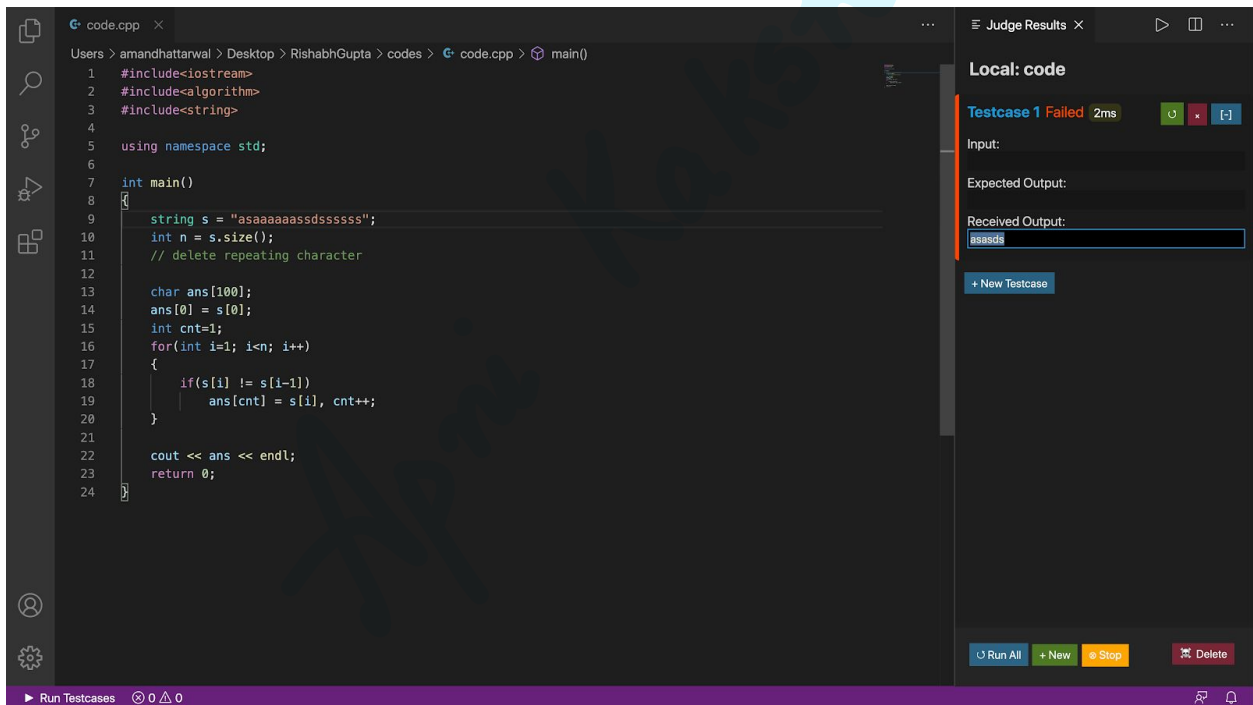
### Compression of strings

Given a string  $s$ , your task is to remove the repeating consecutive characters.

Approach: Create an answer string and iterate in the string from  $i=1$  and check if it is equal to the previous character. Two cases arise

1.  $s[i] = s[i-1]$  - then do not push\_back the  $i^{\text{th}}$  character to the answer string.
2.  $s[i] \neq s[i-1]$  - then push\_back the  $i^{\text{th}}$  character to the answer string.

Code



```
1 #include<iostream>
2 #include<algorithm>
3 #include<string>
4
5 using namespace std;
6
7 int main()
8 {
9     string s = "asaaaaaassdssssss";
10    int n = s.size();
11    // delete repeating character
12
13    char ans[100];
14    ans[0] = s[0];
15    int cnt=1;
16    for(int i=1; i<n; i++)
17    {
18        if(s[i] != s[i-1])
19        {
20            ans[cnt] = s[i], cnt++;
21        }
22    }
23    cout << ans << endl;
24    return 0;
25 }
```

Local: code

Testcase 1 Failed 2ms

Input:

Expected Output:

Received Output:

asads

+ New Testcase

Run All + New Stop Delete

Apni Kaksha