String in C++

`string` is a dynamic sequence of characters provided by the C++ Standard Library. It can resize automatically when characters are added or removed, supports fast random access to individual characters, and provides many useful functions for manipulating text.

---

1. Include Header

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

---

2. Creating a String

```
string s1;            // empty string: ""
string s2("hello");   // initialized with C-style string
string s3 = "world"; // direct initialization with C-style string
string s4(5, 'x');    // string of 5 'x' characters: "xxxxx"
string s5 = s2;       // copy constructor
```

---

3. Common Functions with Examples

3.1. Add Characters / Append

```
string s = "Hello";
s.push_back('!'); // Adds '!' at the end → "Hello!"
s += " World";    // Append C-style string → "Hello! World"
s.append("!!!");  // Append using member function → "Hello!
World!!!"
```

Amortized Time Complexity: `O(1)` for `push_back`, `O(n)` for `append`

3.1.2. Access Characters

```
cout << s[0] << endl;       // Direct access (no bounds check)
cout << s.at(0) << endl;    // Safe access (throws exception if out
of bounds)
cout << s.front() << endl; // First character
cout << s.back() << endl;  // Last character
```

3.1.3. Size and Capacity

```
cout << s.size() << endl;      // Number of characters
cout << s.length() << endl;    // Same as size()
cout << s.capacity() << endl; // Allocated storage capacity
```

3.1.4. Shrink to Fit

```
s.shrink_to_fit(); // Removes unused capacity
(non-binding request)
```

3.1.5. Iterate Over String

```
for (char c : s)
    cout << c << " "; // range-based loop

for (int i = 0; i < s.size(); i++)
    cout << s[i] << " "; // index-based loop
```

3.1.6. Insert, Erase and Replace

```
string s = "hello";
s.insert(1, "!!!");     // Insert "!!!" at index 5
s.erase(5, 3);          // Remove 3 characters from index 5
s.replace(0, 5, "Hi"); // Replace first 5 chars with "Hi"
s.replace(s.begin(), s.begin() + 5, "Hi"); // also supports
iterators
```

3.1.7. Remove Last Character

```
s.pop_back(); // Remove last character
```

3.1.8. Clear Entire String

```
s.clear(); // string becomes empty
```

3.1.9. Check If Empty

```
if (s.empty())
    cout << "Empty";
```

---

4. String Iterator

```
string s = "hello";
string::iterator it;
for (it = s.begin(); it != s.end(); it++)
```

```
    cout << *it << endl;
```

---

5. Sorting a String

```
string s = "hello";
sort(s.begin(), s.end());
sort(s.begin(), s.end(), greater<char>());
```

---

6. Reverse a String

```
string s = "hello";
reverse(s.begin(), s.end());
```

---

7. Find Character

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    string s = "hello world";
    int idx = s.find('l'); // returns index
    cout << idx << endl;
    int idx2 = s.find_first_of('o');
    cout << idx2 << endl;
    int idx3 = s.find_last_of('o');
    cout << idx3 << endl;
    int idx4 = s.find_first_not_of('h');
    cout << idx4 << endl;
    int idx5 = s.find_last_not_of('d');
    cout << idx5 << endl;
    return 0;
}
```

---

8. Get Unique Characters

```
string s = "eeeedbcccdbbbbdccaaabbbbaaa";
```

```
sort(s.begin(), s.end());
auto last = unique(s.begin(), s.end());
s.erase(last, s.end());
cout << s << endl; // Output: abcde
```

---

9. Get Min and Max Character

```
string s = "bdacfehg";
auto it_min = min_element(s.begin(), s.end()); // smallest
character
auto it_max = max_element(s.begin(), s.end()); // largest
character

cout << *it_min << " " << *it_max << endl;
```

---

10. Get Count of a Character

```
string s = "abracadabra";
int count_a = count(s.begin(), s.end(), 'a');
cout << count_a << endl; // Output: 5
```

---

11. Fill all Characters

```
string s = "abracadabra";
fill(s.begin(), s.end(), '*'); // Replace all characters with '*'
cout << s << endl;
```

---

12. Rotate elements

```
string s = "abcde";
rotate(s.begin(), s.begin() + 3, s.end()); // Left rotation by 3 →
"deabc"
cout << s << endl;
```

```
rotate(s.begin(), s.end() - 3, s.end()); // Right rotation by 3 → back to "abcde"
cout << s << endl;
```

---

13. Get Sum

```
string s = "abcde";
int sum = accumulate(s.begin(), s.end(), 0);
cout << sum << endl; // Output: 97 + 98 + 99 + 100 + 101 = 495
```

---

14. Substring

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    string s = "hello_world";
    cout << s.substr(4, 5) << endl; // output: o_wor
    cout << s.substr(6) << endl; // output: world
    return 0;
}
```

---

15. Get Words using stringstream and getline

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    string s;
    getline(cin, s);

    stringstream ss(s);
    string word;
    while (ss >> word)
    {
        cout << word << endl;
    }
    return 0;
}
```

16. Practice Problems

1. **Extract Filename from Path** – Get the file name from a full path
   Input: `/user/code/hello.cpp`
   Output: `hello.cpp`

```cpp
#include <bits/stdc++.h>

using namespace std;

int main()

{

    string path = "/user/code/hello.cpp";

    int pos = path.find_last_of('/');

    cout << path.substr(pos + 1) << endl;

    return 0;

}
```

2. **Check Palindrome** – Check if a string reads the same forward and backward
   Input: `madam`
   Output: `Yes`

```cpp
#include <bits/stdc++.h>

using namespace std;

int main()

{

    string s = "madam";

    string rev = s;

    reverse(rev.begin(), rev.end());

    cout << (s == rev ? "Yes" : "No") << endl;

    return 0;

}
```

3. **Check Anagram** – Check if two strings have the same characters in any order
   Input: `listen`, `silent`
   Output: `Yes`

```cpp
#include <bits/stdc++.h>

using namespace std;

int main()

{

    string a = "listen", b = "silent";

    sort(a.begin(), a.end());

    sort(b.begin(), b.end());

    cout << (a == b ? "Yes" : "No") << endl;

    return 0;

}
```

4. **Count Frequency of Each Character** – Count how many times each character appears
   Input: `aabbc`
   Output: `a:2 b:2 c:1`

```cpp
#include <bits/stdc++.h>

using namespace std;

int main()

{

    string s = "aabfegsahhxbczz";

    vector<int> freq(26, 0);

    for (char c : s)

        freq[c - 'a']++;

    for (int i = 0; i < 26; i++)

        cout << char(i + 'a') << ": " << freq[i] << endl;

    return 0;
```

```
}
```

5. **Convert Case** – Swap uppercase letters to lowercase and vice versa
   Input: `HeLLo`
   Output: `hEllO`

```
#include <bits/stdc++.h>

using namespace std;

int main()

{

    string s = "HeLLo";

    for (char &c : s) {

        if (islower(c)) c = toupper(c);

        else if (isupper(c)) c = tolower(c);

    }

    cout << s << endl;

    return 0;

}
```

6. **Remove All Vowels** – Remove vowels (a, e, i, o, u) from a string
   Input: `education`
   Output: `dctn`

```
#include <bits/stdc++.h>

using namespace std;

int main()

{

    string s = "education", result = "";

    for (char c : s) {

        if (string("aeiouAEIOU").find(c) == -1)
```

```
            result += c;

    }

    cout << result << endl;

    return 0;

}
```

7. **Replace All Spaces with Hyphens** – Replace spaces in a string with `-`
   Input: `this is a test`
   Output: `this-is-a-test`

```
#include <bits/stdc++.h>

using namespace std;

int main()

{

    string s = "this is a test";

    for (char &c : s) {

        if (c == ' ') c = '-';

    }

    cout << s << endl;

    return 0;

}
```

8. **Capitalize First Letter of Each Word** – Make the first letter of each word uppercase
   Input: `hello world from c++`
   Output: `Hello World From C++`

```
#include <bits/stdc++.h>

using namespace std;

int main()

{
```

```
    string s;
    getline(cin, s);
    stringstream ss(s);
    string word;
    string ans;
    while (ss >> word)
    {
        word[0] = toupper(word[0]);
        ans += word;
        ans += " ";
    }
    ans.pop_back();
    cout << ans << endl;
    return 0;
}
```