String

09 June 2021 07:07

1. Creation

   I

```
std::string sSource;
std::string sSource{ "my string" };
std::string sOutput{ sSource };
```

2. Converting numbers to string - using std::ostringstream and std::istringstream

   https://stackoverflow.com/questions/5123674/and-in-c

```
template <typename T>
inline std::string ToString(T tX)
{
    std::ostringstream oStream;
    oStream << tX;
    return oStream.str();
}
```

   ostringstream object takes input like cout << from input and obj.str() gives you the string

```
template <typename T>
inline bool FromString(const std::string& sString, T &tX)
{
    std::istringstream iStream(sString);
    return !(iStream >> tX).fail(); // extract value into tX, return success or not
}
```

   Istringstream object is initialized with string and it outputs to destination obj type

3. Const, Non-const Random access iterator, reverse iterator

iterator	a random access iterator to char (convertible to const_iterator)
const_iterator	a random access iterator to const char
reverse_iterator	reverse_iterator<iterator>
const_reverse_iterator	reverse_iterator<const_iterator>

4. string.capacity() to know capacity - string is dynamic in length & string.reserve(), shrink_to_fit()
5. string.length(), size() to know length & resize(), clear to modify
6. string.empty() to know its empty or not

   I

7. Use string[] or .at to access characters..

   I

8. Special operations
   1. string.data() - pointer to first char - no null termin guarantee
   2. Convert std::string to C-style strings

      sSource.c_str()

      - Returns the contents of the string as a const C-style string i.e. character array
      - A null terminator is appended

   3. Copy from std::string to charac array

```
char buffer[20];
std::string str ("Test string...");
std::size_t length = str.copy(buffer,6(length),5(pos));
buffer[length]='\0';
std::cout << "buffer contains: " << buffer << '\n'; // can print char array like this
```

   4. Find a substr within std::string.. Return start position of found string.. Else gives std::string::npos = -1

      std::size_t found = str.find(str2);

   5. Compare - return 0 if match.. else

      Comparedstring.compare(comparing string)

<0	Either the value of the first character that does not match is lower in the *compared string*, or all compared characters match but the *compared string* is shorter.

>0	Either the value of the first character that does not match is greater in the *compared string*, or all compared characters match but the *compared string* is longer.

```
std::string str1 ("green apple");
std::string str2 ("red apple");
if (str1.compare(str2) != 0)
```

6. Kk

9. Modifiers
    1. pop_back - remove from back
    2. erase
    3. insert
    4. push_back
    5. str.assign(input str)
    6. str.append(input string, other params)
    7. str.replace(input string)
10. Overloads
    1. Addition +
    2. +=
    3. << and >> (cout and cin)
    4. Relational operators for comparison - use compare

I
11. F

I
12. F

I
13. F

I
14. F

I
15. F

I
16. D

I
17. F

I
18. F

I
19. F

I
20. F
21. F

I
22. F

I
23. F

I
24. F

I
25. F

I
26. F

I
27. D

I
28. F

I
29. F

I

30. F
    I
31. F
32. F
    I
33. F
    I
34. F
    I
35. F
    I
36. F
    I
37. F
    I
38. D
    I
39. F
    I
40. F
    I
41. F
    I
42. F
43. F
    I
44. F
    I
45. F
    I
46. F
    I
47. F
    I