

# Modifying Operations

C++ gives us a variety of tools to modify and manipulate strings.

Strings have many operations that can modify them. `str.assign` assigns a new string to the string `str`. With `str.swap` we can swap two strings. To remove a character from a string use `str.pop_back` or `str.erase`. On the contrary, `str.clear` or `str.erase` deletes the whole string. To append new characters to a string, use `+=`, `std.append` or `str.push_back`. We can use `str.insert` to insert new characters or `str.replace` to replace them.

Methods	Description
<code>str= str2</code>	Assigns <code>str2</code> to <code>str</code> .
<code>str.assign(...)</code>	Assigns a new string to <code>str</code> .
<code>str.swap(str2)</code>	Swaps <code>str</code> and <code>str2</code> .
<code>str.pop_back()</code>	Removes the last character from <code>str</code> .
<code>str.erase(...)</code>	Removes characters from <code>str</code> .
<code>str.clear()</code>	Clears the <code>str</code> .
<code>str.append(...)</code>	Appends characters to <code>str</code> .
<code>str.push_back(s)</code>	Appends the character <code>s</code> to <code>str</code> .
<code>str.insert(pos, ...)</code>	Inserts characters in <code>str</code> starting at <code>pos</code> .

```
str.replace(pos, len, ...)
```

Replaces the `len` characters from  
`str` starting at `pos`

## Methods for modifying a string

The operations have many overloaded versions. The methods `str.assign`, `str.append`, `str.insert`, and `str.replace` are very similar. All four can not only be invoked with C++ strings and substrings but also characters, C strings, C string arrays, ranges, and initializer lists. `str.erase` can not only erase a single character and whole ranges, but also many characters starting at a given position.

The following code snippet shows many of the variations. For the sake of simplicity, only the effects of the strings modifications are displayed:

```
#include <iostream>
#include <string>

int main(){

    std::cout << std::endl;

    std::cout << "ASSIGN: " << std::endl;

    std::string str{"New String"};
    std::string str2{"Other String"};
    std::cout << "str: " << str << std::endl;

    str.assign(str2, 4, std::string::npos);
    std::cout << str << std::endl;

    str.assign(5, '-');
    std::cout << str << std::endl;
    std::cout << std::endl;

    std::cout << "DELETE" << std::endl;

    str={"0123456789"};
    std::cout << "str: " << str << std::endl;
    str.erase(7, 2);
    std::cout << str << std::endl;

    str.erase(str.begin()+2, str.end()-2);
    std::cout << str << std::endl;

    str.erase(str.begin()+2, str.end());
    std::cout << str << std::endl;

    str.pop_back();
    std::cout << str << std::endl;
```

```
str.erase();
std::cout << str << std::endl;

std::cout << "APPEND" << std::endl;

str="01234";
std::cout << "str: " << str << std::endl;

str+="56";
std::cout << str << std::endl;

str+='7';
std::cout << str << std::endl;

str+={'8', '9'};
std::cout << str << std::endl;

str.append(str);
std::cout << str << std::endl;

str.append(str, 2, 4);
std::cout << str << std::endl;

str.append(10, '0');
std::cout << str << std::endl;

str.append(str, 10, 10);
std::cout << str << std::endl;

str.push_back('9');
std::cout << str << std::endl;

std::cout << std::endl;

std::cout << "INSERT" << std::endl;
str={"345"};
std::cout << "str: " << str << std::endl;

str.insert(3, "6789");
std::cout << str << std::endl;

str.insert(0, "012");
std::cout << str << std::endl;

std::cout << std::endl;

std::cout << "REPLACE" << std::endl;
str={"only for testing purpose."};
std::cout << "str: " << str << std::endl;

str.replace(0, 0, "0");
std::cout << str << std::endl;

str.replace(0, 5, "Only", 0, 4);
std::cout << str << std::endl;

str.replace(16, 8, "");
std::cout << str << std::endl;

str.replace(4, 0, 5, 'y');
std::cout << str << std::endl;
```

```
str.replace(str.begin(), str.end(), "Only for testing purpose.");  
std::cout << str << std::endl;  
  
str.replace(str.begin()+4, str.end()-8, 10, '#');  
std::cout << str << std::endl;  
  
std::cout << std::endl;  
  
}
```



Modifying strings

---

In the next lesson, we'll discuss the numeric conversion of strings.