C++ string class - Quick Reference Guide

#include <string>; strings are **objects** in C++. The C language used char arrays/pointers and c-string literals.

C++ strings may be assigned, modified, etc. using operators or member functions.

Simplest way to create a string object: string name; / string name = "Jack"; / string name("Jack");

C++14 supports a string class literal: "C++ string literal"s (append 's' to a c-string literal)

string constructors

string class constructor prototypes:

1. string str1;
2. string str1 (const char* str);
3. string str1 (const string& s);
4. string str1 (input_iterator start, input_iterator stop);

The string constructors create a new string containing:

1. create an empty string – ""
2. create a string object using a c-string literal
3. create a new string object from an existing string object using the copy constructor
4. create a string object using iterators to specify a collection of characters for the string

Operator	Meaning
`>>`	extracts characters from input stream up to whitespace, insert into string
`<<`	inserts string into output stream
`=`	assigns one string object to another
`+=`	appends a string or a cstring to an existing string object
`+`	concatenates two string objects or a string and a c-string
`[]`	dereferences a single character in a string. No bounds checking!
`>, >=, <, <=, ==, !=`	relational operators for string comparison

C++ string class Methods	Method Descriptions - These methods use zero-based indexing
The following string class methods use string object s declaration: `string s;`	
`s.at(index)`	dereferences a single character in string s at position index. The at( ) method does bounds checking and throws an exception if an invalid index is used
`s.begin()`	returns an iterator pointing to the first character in the string s
`s.capacity()`	returns the amount of storage (in bytes) allocated for string s
`s.clear()`	clears string s by deleting all the characters stored in it.
`s.compare(s2)`	performs a comparison like the strcmp function with the same return values: (negative integer, 0, positive integer) s2 can be a string object or a character array

C++ string class Methods	Method Descriptions - These methods use zero-based indexing
The following string class methods use string object s declaration: **string s;**	
s.c_str()	returns a non-modifiable standard C character array version of string s
s.data()	returns a pointer to the first character in string s
s.empty()	returns true if string s is an empty string (“”)
s.end()	returns an iterator pointing just past the last character in string s
s.erase(pos, n)	erases n characters from string s, beginning at position pos
s.Insert(pos, str)	Inserts a copy of str into string s beginning at position pos str can be a string object or a character array
s.insert(pos, n, 'z')	Inserts 'z' n times into string s at position pos
s.length()	returns the length of string s. Note: you can also use the size( ) method
s.replace(pos, n, str)	replaces n characters in string s beginning at position pos with the characters in string object str
s.resize(n) **s.resize(n, 'z')**	changes the space allocatied for string s to n characters If n is less than the current size of the string: s is truncated to n characters. If n is greater than the current size of the string: s is expanded and 'z' is appended at the end enough times to fill the new positions. If there is no second argument s is padded with spaces.
s.size()	returns the length of string s. Note: you can also use the length( ) method
s.substr(pos, n)	returns a copy of a substring. The substring is n characters long and begins at position pos in string s
s.swap(str)	swaps the contents of string s with string str
to_string(<num exp>)	converts a number to a string. string number = to_string(123); the function returns a string. **Note: this is NOT a method it is a function (like getline) declared in the string header file.**

string function getline

The getline string function should be used when you want input a string from the keyboard using the cin object or from a file using a stream object that may contain embedded whitespace characters (spaces/tabs)

Note: This is not a method of the string class and should not be confused with the c-string function getline!

Syntax: `getline( <stream object>, <string variable> );`

- `<stream object>` is **cin** or the name of some other input stream object
- `<string variable>` is the name of a string variable to receive the input string

Examples: `getline ( cin, studentName );` `getline ( inFile, customerFullName );`