C/C++ How to / Cheatsheet

More info at:

cplusplus.com
cppreference.com
isocpp.org
learncpp.com

by NeptuneLuke

Escape characters

`\n`	`new line`
`\b`	`backspace`
`\t`	`horizontal tab`
`\v`	`vertical tab`
`\\`	`backslash`
`\0`	`null char (string character terminator)`
`\?`	`?`
`\"`	`"`
`\'`	`'`

Arrays and Vectors

get the size of an array

`size(arr)`

get the size of a vector

`v.size()`

get the element at the index position from a vector

`v[index]`

`v.at(index)`

directly print arrays of char

```
char char_array [] = "string";
cout << char_array;
```

using iterators to iterate through a vector

```
vector<T> v;

for(auto it = v.begin(); it != v.end(); ++it) {

    // it is the same as writing v[i]
    it.doSomething();
}

for(auto & elem : v) {

    // elem is the same as writing v[i]
    elem.doSomething();
}
```

create a matrix using vectors

```
vector< vector<T> > matrix;

vector< vector<T> > matrix(rows, vector<T>(columns, init_value));
```

reverse a vector

```
#include <algorithm>
vector<T> v;
```

```
reverse(v.begin(), v.end());
```

---

sort a vector

```
// may require this header
#include <algorithm>   C++ header ( sort )

vector<int> v = {5,4,3,2,1};
sort(v.begin(), v.end());
```

---

initialize a vector from another vector

```
vector<int> v1 = {1,2,3,4,5};
vector<int> v2 = (v1.begin(), v1.end());
```

---

resize a vector

```
vector<int> v1 = {1,2,3,4,5};
v1.resize(3);    // now v1 is {1, 2, 3}

vector<int> v1 = {1,2,3,4,5};
v1.resize(10,0);    // now v1 is {1, 2, 3, 4, 5, 0, 0, 0, 0, 0}
```

---

compare vectors

```
vector<int> v1 = {1,2,3,4,5};
vector<int> v2 = {1,2,3,4,5};
vector<int> v3 = {1,2,3,4,6};

v1 == v2 -> true
v1 == v3 -> false
```

Strings

ASCII numbers are chars from 48 to 57

ASCII uppercase letters are chars from 65 to 90

ASCII lowercase letters are chars from 97 to 122

---

get the size of a string

```
s.size()
```

```
s.length()
```

---

get the char at the index position from a string

```
s[index]
```

```
s.at(index)
```

---

convert from char to int

```
int x = (int)character - 48;
```

```
int x = character - '0';
```

---

convert from int to string and from string to int

```
// may require this header
#include <string>  C++ header ( to_string() / sto*() )

string s = to_string(42);  // for all numerical types
int i = stoi(s);
long l = stol(s);
double d = stod(s);
```

---

convert a string to a char array

```
// may require these headers
#include <string.h>  C header ( strcpy )
#include <cstring>   C++ header ( strcpy )

string s = "string";
char char_str [s.length()];

// converts a string to a char array
strcpy(char_str, s.c_str());
```

initialize a string with a char array

```
char char_str [];

// constructor of a string with a char array as argument
string s(char_str);
```

take a string in input the correct way

```
string s;
cin >> s;      // only takes the string up to the first space

// cin.ignore()
// if, before taking the string in input other data are
// taken in input

getline(cin, s);    // takes the entire string
```

remove characters from a string

```
// may require these headers
#include <algorithm>  C++ header ( remove_if() )
#include <string>
C++ header( erase() / find_first_not_of() / find_last_not_of())
#include <regex>      C++ header ( regex_replace() )

// remove all spaces
s.erase(remove_if(s.begin(), s.end(), isspace), s.end());

// remove leading/trailing spaces with custom method
string trim_string(string s) {

   const string white_spaces = " \t\n\r\f\v";

   // Remove leading whitespace
   size_t first_non_space = s.find_first_not_of(white_spaces);
   s.erase(0, first_non_space);

   // Remove trailing whitespace
   size_t last_non_space = s.find_last_not_of(white_spaces);
   s.erase(last_non_space + 1);

   return s;
}
```

```
// removing leading, trailing and extra spaces
s = regex_replace(s, regex("^ +| +$|( ) +"), "$1");

// remove only extra spaces
s = regex_replace(s, regex(" +"), " ");

// remove chars with custom method
s = "my data";
s.erase(remove_if(s.begin(), s.end(), my_predicate), s.end());
bool my_predicate(char c) {
    // return true if i want to remove c
    // return false in any other case
}
```

STD Stream

---

cout <<

`std::cout <<`

This instruction does not directly display data.

It first sends data to be displayed to a buffer and only after the buffer is full (all the data of **`std::cout`** are sended) the data is displayed to the output.

If we want to send the data directly to the output we can use **`std::flush`**.

---

cin.ignore()

`std::cin.ignore()`

Is used to reset the stream buffer.

If the buffer of the stream is containing some data not taken from the previous **`std::cin`** (like taking in input an int before a string) we can use **`std::cin.ignore()`**.

```
// may require these headers
#include <iostream>  C++ header ( cin / cin.ignore() )

int n;
string s;

cin >> n;           // n owns the input stream
cin.ignore();       // reset the input stream
getline(cin, s);    // s owns the input stream
```

---

endl - “\n”

`std::endl`

`“\n”`

Because **`std::endl`** terminates the current line but also flushes the stream, we should use **`\n`** instead.

```
// may require these headers
#include <iostream>  C++ header ( cout / endl)

cout << value;    // print value
cout << endl;     // set a new line (flushing the stream)

cout << value << “\n”;  // set a new line (not flushing the stream)
```

Miscellaneous

generate pseudo-random numbers

```
// may require these headers
#include <stdlib.h>  C header ( rand )
#include <time.h>    C header ( srand )
#include <cstdlib>   C++ header ( rand )
#include <ctime>     C++ header ( srand )

srand(time(0));        // seeds the srand to get real random numbers
int x = rand()%n       // [0,n-1]
int x = rand()%n-m     // [-m, m-1]
int x = rand()%n*2     // [0, n-2]
```

sort a variety of objects

```
// may require this header
#include <algorithm>   C++ header ( sort )

int array[5];
sort(array, array + size(array));

string s = "edcba";
sort(s.begin(), s.end());

vector<int> v = {5,4,3,2,1};
sort(v.begin(), v.end());
```