C/C++ How to / Cheatsheet

More info at:

cplusplus.com

cppreference.com

isocpp.org

You may find interesting:

cppquiz.org

Escape characters

\n	new line
\b	backspace
\t	horizontal tab
\v	vertical tab
\\	backslash
\0	null char (string character terminator)
\?	?
\“	“
\‘	‘

Arrays

get the size of an array

`size(arr)`

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 saying v[i]
    it.doSomething();
}


for(auto & elem : v) {

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

create a matrix using vectors

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

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

Strings

get the size of a string

`.size()`

`.length()`

get the char at the index position from a string

`s[index]`

convert a string to a char array

may require these headers

```
#include <string.h>  C header ( strcpy )
#include <cstring>   C++ header ( strcpy )
```

```
string s = "hello!";
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;
std::cin >> s;      // only takes the string up to the first space
getline(cin, s);    // s owns the input stream
```

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()`.

```
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.

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

std::cout << value << "\n";  // set a new line (not flushing)
```

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));        // 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 these headers

```
#include <algorithm>  C++ header ( sort )
```

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

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