# C++ for beginners Cheat Sheet

by jofre pla via cheatography.com/26595/cs/7478/

## Include Headers

| | |
|---|---|
| #include | <headerfile> |

**Common Headers / Libraries**

| | |
|---|---|
| #include <stdio.h> | I / O functions |
| #include <string.h> | string functions |
| #include <time.h> | time functions |
| #include <stdlib.h> | memory, rand, ... |
| #include <math.h> | math functions |
| #include <iostream.h> | |
| #include <fstream.h> | I / O file functions |
| #include "myfile.h" | Insert file in current directory |

## Namespaces

using namespace std;

## Comments

// One line comment text

/* multiple line block comment text */

## Basic Variable Types

NUMBER

```
int a; float a;
```

CHARACTER

```
char car; string s;
char car = 'c'; string s = "hola mon";
```

BOOL

```
bool b = false/true;
```

## Basic input / Output Operators

| | |
|---|---|
| cin | cin >> var |
| cout | cout<<"The variable has"<<var |

## Basic Operators / Math Operators

| | | | |
|---|---|---|---|
| + | Add | - | Less |
| * | Mult | / | Div |
| % | Mod | | |
| ++var / --var | | var++ / var-- | |

## Conditionals

| | |
|---|---|
| A == B | if A is equal to B, this is true; otherwise, it's false |
| A != B | if A is NOT equal to B, this is true; otherwise, it's false |
| A < B | if A is less than B, this is true; otherwise, it's false |
| A > B | if A is greater B, this is true; otherwise, it's false |
| A <= B | if A is less than or equal to B, this is true; otherwise, it's false |
| A >= B | if A is greater or equal to B, this is true; otherwise, it's false |
| A ! B | if A |
| A && B | if condition A and condition B are true, this is true; otherwise, it's false. |
| A \|\| B | if condition A or condition B is true, this is true; otherwise, it's false. |

Boolean expressions in C++ are evaluated left to right!

## Arrays

type array_name [ # of elements ];

```
int price [10];
```

type array_name [# elements] [# elements];

```
int price [5] [10];
```

· Array index starts at 0.

· Ex: Access 3rd element : cout<<price [2];

## Control Flow

**if sentence**

```
if ( conditional ) {
  // do something
}
else if ( another_conditional ) {
  // do something else
}
else {
  // do something as default
}
```

**while sentence**

```
while ( conditional ) {
// do something
}
```

placing **"break;"** breaks out of the loop.

placing **"continue;"** jumps to next loop.

**for sentence**

```
for ( init; test; command ) {
  // do something
}
```

**"break;"** and **"continue;"** identical effects.

**do while sentence**

```
do {
  //do something
} while (bool expression);
```

**switch case sentence**

```
switch ( variable )
{
case value1:
  // do something;
  break;
case value2:
  // do something else;
  break;
[default:
  // do something by default:
  break; ]
}
```

By **jofre pla**

cheatography.com/jofre-pla/

Published 4th April, 2016.

Last updated 19th May, 2016.

## File Input / Output

```
#include <fstream.h>
ifstream file; //read buffer
ofstream file; //write buffer
file.open ("fi len ame ", [file
mode consta nt]);
//Test if the file was created
if(fs.i s_ ope n() )     if(fs)
//Read s/W rites like cin and
cout
file >> var; //Read
file << ''Text: " << var <<
endl; //Write
//Read Entire line
getline (file, Str ing);
//Read until it arrives at the
end of file
while( fil e.e of())
//Detect if the read/write fail
if(fil e.f ail())
//Close File
file.c lose();
```

**File Mode Constants**

ios::in //Opens file for reading
ios::out //Opens file for writing
ios::app //Causes output to be appended at EOF
ios::trunc //Destroys the previous contents
ios::nocreate //Causes open() to fail if file doesn't already exist
ios::noreplace //Causes open() to fail if file already exists

## Procedures

```
//Declaration
void Proced ure Name()
{
     // do something
}
//Call to procedure
Proced ure Name();
```

In the procedures we don't receive variables and don't return other variable.

## Functions

```
//Declaration
[returnType] functi onName (
[input1Type input1 Name,
input2Type input2 Name, ....] )
{
     // do something
      return value; // value
must be of type returnType
}
//Call to function
[retur ntype var =] functi -
onName ([input1Type input1 Name,
input2Type input2 Name, ....])
```

We have two methods to create and call functions:
passed with values and passed for reference.
**Pass by reference** : we put *&* before variable in the declaration.

## Structures

Structure declaration :

```
struct <structure_name>
{
   <type> <name>, <name>, ... ;
   <type> <name>, <name>, ... ;
}
```

Var declaration with structure type :

```
<structure_name> var_name;
```

Acces to structure :

```
var_name.name;
```

By **jofre pla**
cheatography.com/jofre-pla/

Published 4th April, 2016.
Last updated 19th May, 2016.