Project base

```
// include Standard Input and
Output
#include <stdio.h>
// include CS 50 helper
#include <cs50.h>
int main(void)
{
    // Asks for the user name
    string name = get_strin-
g("What is your name? ");
    // Print the message
    printf("hello, %s.\n",
name);
}
```

Declaring variables

```
// Variables declaration
// The structure to declare
variables is:
// type variable_name;
int my_integer;
char my_single_char;
bool true_or_false;
string a_long_string;
float price;
// you can also declare multiple
vars of the same type in one
line
string first, last, full;
int total, a, b;
// assign "=" values to a
variable
my_integer = 10;
my_single_char = 'A';
true_or_false = true;
```

Declaring variables (cont)

```
a_long_string = "A really long
text";
price = 1.25;
// note: When working with
strings the value needs to be
under double quotes "
// When working with chars the
value needs to be under single
quotes '
// intialization
int my_age = 30;
float product_price = 123.45;
string message = "thank you";
char letter = 'A';
```

Conditional statements

```
if (boolean-expression)
{
    // code if true
}
// ------
if (boolean-expression)
{
    // code if true
}
else
{
    //code if false
}
// ------
if (boolean-expression)
{
```

Conditional statements (cont)

```
    // code if true
}
else if (another-boolean-expr-
ession)
{
    //code if first is false but
second is true
}
else
{
    //code if all are false
}
//--
// Example ok ?: use
int x;
if (boolean-expression)
{
    x = 5;
}
else
{
    x = 10;
}
// this is a short form of the
same code above
int x = boolean-expression ? 5 :
10;
//--------
int my_number = 10;
switch(my_number)
{
    case 1:
```

Conditional statements (cont)

```
        printf("Number is 1");
      break;
    case 10:
        printf("Number is 10");
      break;
    default:
        printf("Number is
invalid");
}
```

Loops

```
while (boolean-expression)
{
    // code will be executed
while the boolean-expression is
true
}
do
{
    // do-while will always
execute at least once.
    // code will be executed
while the boolean-expression is
true
}
while (boolean-expression);
for (initilization ; boolean-e‐
xpression ; increment)
{


}
for (int index = 0 ; index < 20
; index ++)
{


}
```

Data Types

int	Integers - 4 bytes (32 bits)
char	Single characters - 1 byte (8 bits)
float	Real numbers - 4 bytes (32 bits)
double	Double precision for floats
void	Is a type but not a Data Type. Used in functions to say that params or returns are not required

CS50 ones

bool	Boolean value (true or false)
string	Serie of characters (words, sentences, etc)

Arithmetic Operators

+	Sum (add)
-	Subtraction
*	Multiplication
/	Division
%	Module, gets the remainder (15 % 7 == 1)
x++	x = x + 1
x--	x = x - 1
x += 5	x = x + 5
x *=5	x = x * 5

Boolean expressions

Boolean expression is used in C to compare values.

There are just two possible values for it **true** or **false**.

On boolean expressions, anything other than **false, 0 or NULL** is considered as **true**.

Logical operators:

&& - **AND** is true only if both operands are true, otherwise false

|| - **OR** is false only if both operands are false, otherwise true

! - **NOT** inverts the value of its operand.

Relational operators

== - **EQUAL** is true only if both operands are equal.

!= - **NOT EQUAL** is true if the operands are not equal.

< - **LESS THAN** is true if the left operand is less than the one in the right.

> - **GREATER THAN** is true if the left operand is greater than the one in the right.

>= - **GREATER THAN OR EQUAL TO** is true if the left operand is greater than or equal to the one in the right.

<= - **LESS THAN OR EQUAL TO** is true if the left operand is less or equal than the one in the right.

By **Ruan**

cheatography.com/ruan/

Published 28th September, 2020.
Last updated 28th September, 2020.
Page 2 of 3.

cs50.h Functions

get_char("MESSAGE")	prompts user for a char
get_double("MESSAGE")	prompts user for a double
get_float("MESSAGE")	prompts user for a float
get_int("MESSAGE")	prompts user for an int
get_long("MESSAGE")	prompts user for an long
get_string("MESSAGE")	prompts user for a string

Truth table

	AND &&	
x	y	(x && y)
true	true	true
true	false	false
false	true	false
false	false	false
	OR ||	
x	y	(x || y)
true	true	true
true	false	true
false	true	true
false	false	false

Linux commands

make file_name	Compile the code
./file_name	Execute a compiled file
cd folder_name	Change directory
cd	Go back to the home folder (~)
cp original_file copied_file	Copy file
ls	List files
mkdir name_of_the_folder	Create a folder
pwd	show the current path
rm file_name	Remove a file
rm -R folder	Remove a folder recursively
touch file_name	Create a file
mv original_file renamed_file	Rename or move a file

String scapes

\n	New line
\r	Return
\t	Tab
\"	Double quote
\\	Backslash

more http://www.lix.polytechnique.fr/~liberti/public/computing/prog/c/C/FUNCTIONS/escape.html

Format String

%c	Char
%s	String
%d || %i	Int
%f	Double or float
%.#f	Float (limit the output to # decimal places)
%%	%