

## C

# Reminder

## First program

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    printf("Lucky world");
    return 0;
}
```

- **#include**  
The files we need to run the software. (stdio is required to print text on the screen)
- **int argc, char \*argv[]**  
(These arguments are optional)  
When we run the software in a terminal (`./test`) we can give it arguments.  
These arguments are stocked in the `argv` array, and the number of arguments is stocked in `argc`.  
(The first argument is the name of the file we have run)

For example : `./test lucky 777`

`argc = 3`

`argv[] = {"test", "lucky", "777"}`

- **printf("Lucky world");**  
The function to print text on the standard output (screen).  
(This function requires the library `<stdio.h>`)
- **return 0;**  
(this line is optional)  
In C, the main method will return **0** if no problem occurred during the execution.

## Comma truncation

```
double ko;
ko = 5/2;
//ko = 2

double ok;
ok = 5.0/2.0
//ok = 2.5
```

## Global and static

```
#include <stdio.h>

int w;
static int x;

static void test();

int main() {
    int y;
    static int z;
    return 0;
}

static void test() {
    printf("Lucky world!");
}
```

- **int w;**
  - **w** is accessible from anywhere.
- **static int x;**
  - **x** is accessible from the current file.
- **int y;**
  - **y** is only accessible inside the function.
- **static int z;**
  - **z** is only accessible inside the function, but will keep it's value.
- **static void test()**
  - **test()** is only callable by a function inside the current file.

## Switch case

```
unsigned langNumber;
printf("How many languages do you know?");
scanf("%d", &langNumber)

switch (langNumber)
{
case 0:
    printf("Wait what?");
    break;
case 1:
    printf("I guess you speak english");
    break;
case 2:
    printf("You are bilingual");
    break;
case 3: //Will
case 4: //Execute
case 5: //This line:
```

```

    printf("Wow! Nice");
    break;
default:
    printf("Amazing!!");
    break;
}

```

## Ternary condition

```

if (isOk) {
    result = "ok";
} else {
    result = "error";
}

// Same as:
result = (isOk) ? "ok" : "error";

```

## Prototypes

---

```

#include <stdio.h>

test();

int main() {
    test();
    return 0;
}

void test() {
    printf("ok");
}

```

- As the `main()` is before the function `test()`, the compiler will think that `test()` doesn't exist.  
So we can add the prototype `test()`; at the beginning of the code, to tell gcc that we will use a function called `test()` after.  
*(If we put the `main()` at the end of the code, prototypes are not required)*

## Headers

---

`main.c`

```
#include "test.h"

int main() {
    testFunction();
}
```

### test.c

```
#include "test.h"

void testFunction() {
    printf("Lucky world!");
}
```

### test.h

```
#ifndef TEST

#define TEST
#include <stdio.h>
void testFunction();

#endif
```

- **main.c**

We include `test.h`

- **test.h**

If **TEST** is not defined yet:

- We define it
- We include `<stdio.h>`
- We call the prototype of `testFunction()`.

*(The `#ifndef` stops at `#endif`)*

- **test.c**

We include `test.h` and write the codes of `testFunction()`.

*(`<stdio.h>` has already been included in `test.h`)*

Note:

- `< >` is for standard libraries in `/usr/include/` directory *(on linux)*
- `" "` is for custom headers inside the current directory

## Pointers