



C Piscine

C 08

*Summary: This document is the subject for C 08 module of the C Piscine @ 42.*

*Version: 13*

# Contents

<b>I</b>	<b>Instructions</b>	<b>2</b>
<b>II</b>	<b>Foreword</b>	<b>4</b>
<b>III</b>	<b>Exercise 00 : ft.h</b>	<b>5</b>
<b>IV</b>	<b>Exercise 01 : ft_boolean.h</b>	<b>6</b>
<b>V</b>	<b>Exercise 02 : ft_abs.h</b>	<b>8</b>
<b>VI</b>	<b>Exercise 03 : ft_point.h</b>	<b>9</b>
<b>VII</b>	<b>Exercise 04 : ft_strs_to_tab</b>	<b>10</b>
<b>VIII</b>	<b>Exercise 05 : ft_show_tab</b>	<b>12</b>
<b>IX</b>	<b>Submission and peer-evaluation</b>	<b>13</b>

# Chapter I

## Instructions

- Only this page serves as your reference, do not trust rumors.
- Watch out! This document may change before submission.
- Ensure you have the appropriate permissions on your files and directories.
- You must follow the **submission procedures** for all your exercises.
- Your exercises will be checked and graded by your fellow classmates.
- Additionally, your exercises will be evaluated by a program called **Moulinette**.
- **Moulinette** is meticulous and strict in its assessment. It is fully automated, and there is no way to negotiate with it. To avoid unpleasant surprises, be as thorough as possible.
- **Moulinette** is not open-minded. If your code does not adhere to the Norm, it won't attempt to understand it. **Moulinette** relies on a program called **norminette** to check if your files comply with the Norm. TL;DR: Submitting work that doesn't pass **norminette**'s check makes no sense.
- These exercises are arranged in order of difficulty, from easiest to hardest. We **will not** consider a successfully completed harder exercise if an easier one is not fully functional.
- Using a forbidden function is considered cheating. Cheaters receive a grade of **-42**, which is non-negotiable.
- You only need to submit a **main()** function if we specifically ask for a **program**.
- **Moulinette** compiles with the following flags: **-Wall -Wextra -Werror**, using **cc**.
- If your program does not compile, you will receive a grade of **0**.
- You **cannot** leave **any** additional file in your directory beyond those specified in the assignment.
- Have a question? Ask the peer on your right. If not, try the peer on your left.

- Your reference guide is called **Google / man / the Internet / ...**
- Check the "C Piscine" section of the forum on the intranet or the Piscine on Slack.
- Carefully examine the examples. They may contain crucial details that are not explicitly stated in the assignment...
- By Odin, by Thor! Use your brain!!!

# Chapter II

## Foreword

Here's what Wikipedia has to say about Platypus:

The platypus (*Ornithorhynchus anatinus*), also known as the duck-billed platypus, is a semiaquatic egg-laying mammal endemic to eastern Australia, including Tasmania. Together with the four species of echidna, it is one of the five extant species of monotremes, the only mammals that lay eggs instead of giving birth. The animal is the sole living representative of its family (*Ornithorhynchidae*) and genus (*Ornithorhynchus*), though a number of related species have been found in the fossil record.


The unusual appearance of this egg-laying, duck-billed, beaver-tailed, otter-footed mammal baffled European naturalists when they first encountered it, with some considering it an elaborate hoax. It is one of the few venomous mammals, the male platypus having a spur on the hind foot that delivers a venom capable of causing severe pain to humans. The unique features of the platypus make it an important subject in the study of evolutionary biology and a recognisable and iconic symbol of Australia; it has appeared as a mascot at national events and is featured on the reverse of its 20-cent coin. The platypus is the animal emblem of the state of New South Wales.

Until the early 20th century, it was hunted for its fur, but it is now protected throughout its range. Although captive breeding programs have had only limited success and the platypus is vulnerable to the effects of pollution, it is not under any immediate threat.

This subject is absolutely not talking about platypuses.

# Chapter III

## Exercise 00 : ft.h


	Exercise 00
ft.h	
Turn-in directory: <i>ex00/</i>	
Files to turn in: <b>ft.h</b>	
Allowed functions: <b>None</b>	

- Create your **ft.h** file.
- It should contain the prototypes of all the following functions:

```
void    ft_putchar(char c);
void    ft_swap(int *a, int *b);
void    ft_putstr(char *str);
int     ft_strlen(char *str);
int     ft_strcmp(char *s1, char *s2);
```

# Chapter IV

## Exercise 01 : ft\_boolean.h

	Exercise 01
	ft_boolean.h
	Turn-in directory: ex01/
	Files to turn in: ft_boolean.h
	Allowed functions: None

- Create a ft\_boolean.h file. It will compile and run the following main appropriately:

```
#include "ft_boolean.h"

void      ft_putstr(char *str)
{
    while (*str)
        write(1, str++, 1);
}

t_bool    ft_is_even(int nbr)
{
    return ((EVEN(nbr)) ? TRUE : FALSE);
}

int       main(int argc, char **argv)
{
    (void)argv;
    if (ft_is_even(argc - 1) == TRUE)
        ft_putstr(EVEN_MSG);
    else
        ft_putstr(ODD_MSG);
    return (SUCCESS);
}
```

- This program should display:

```
I have an even number of arguments.
```

- OR:

```
I have an odd number of arguments.
```

- followed by a line break when appropriate.




Norminette must be launched with the `-R CheckDefine` flag.  
Moulinette will use it too.



# Chapter V

## Exercise 02 : ft\_abs.h

	Exercise 02
	ft_abs.h
	Turn-in directory: <i>ex02/</i>
	Files to turn in: <b>ft_abs.h</b>
	Allowed functions: <b>None</b>

- Create a macro ABS which replaces its argument with its absolute value:


```
#define ABS(Value)
```



Norminette must be launched with the `-R CheckDefine` flag.  
Moulinette will use it too.

# Chapter VI

## Exercise 03 : ft\_point.h

	Exercise 03
ft_point.h	
Turn-in directory: <i>ex03/</i>	
Files to turn in: <b>ft_point.h</b>	
Allowed functions: None	

- Create a file **ft\_point.h** that will compile the following main:

```
#include "ft_point.h"


void      set_point(t_point *point)
{
    point->x = 42;
    point->y = 21;
}

int       main(void)
{
    t_point      point;

    set_point(&point);
    return (0);
}
```

# Chapter VII

## Exercise 04 : ft\_strs\_to\_tab

	Exercise 04
ft_strs_to_tab	
Turn-in directory: <i>ex04/</i>	
Files to turn in: <code>ft_strs_to_tab.c</code>	
Allowed functions: <code>malloc</code> , <code>free</code>	

- Create a function that takes an array of strings as an argument and the size of this array.
- Here's how it should be prototyped:

```
struct s_stock_str *ft_strs_to_tab(int ac, char **av);
```

- It will transform each element of `av` into a structure.
- The structure will be defined in the `ft_stock_str.h` file that we will provide, like this:


```
typedef struct s_stock_str
{
    int size;
    char *str;
    char *copy;
} t_stock_str;
```

- `size` being the length of the string;
  - `str` being the string;
  - `copy` being a copy of the string;
- It should keep the order of `av`.

- The returned array should be allocated in memory and its last element's `str` set to 0, this will mark the end of the array.
- It should return a NULL pointer if an error occurs.
- We'll test your function with our `ft_show_tab` (next exercise). Make it work according to this!

# Chapter VIII

## Exercise 05 : ft\_show\_tab

	Exercise 05
	ft_show_tab
	Turn-in directory: <i>ex05/</i>
	Files to turn in: <b>ft_show_tab.c</b>
	Allowed functions: <b>write</b>

- Create a function that displays the content of the array created by the previous function.
- Here's how it should be prototyped:

```
void ft_show_tab(struct s_stock_str *par);
```

- The structure will be the same as the previous exercise and will be defined in the `ft_stock_str.h` file
- For each element, we'll display:
  - the string followed by a `'\n'`
  - the size followed by a `'\n'`
  - the copy of the string (that could have been modified) followed by a `'\n'`
- We'll test your function with our `ft_strs_to_tab` (previous exercise). Make it work according to this !

# Chapter IX

## Submission and peer-evaluation

Submit your assignment to your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Make sure to double-check the filenames to ensure they are correct.



You must submit only the files required by the project instructions.