

C Piscine C 08

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

Contents

Ι	Instructions	2
II	Foreword	4
III	Exercise 00 : ft.h	5
IV	Exercise 01 : ft_boolean.h	6
V	Exercise 02 : ft_abs.h	8
VI	Exercise 03 : ft_point.h	9
VII	Exercise 04 : ft_strs_to_tab	10
VIII	Exercise 05 : ft_show_tab	12

Chapter I

Instructions

- Only this page will serve as reference: do not trust rumors.
- Watch out! This document could potentially change before submission.
- Make sure you have the appropriate permissions on your files and directories.
- You have to follow the submission procedures for all your exercises.
- Your exercises will be checked and graded by your fellow classmates.
- On top of that, your exercises will be checked and graded by a program called Moulinette.
- Moulinette is very meticulous and strict in its evaluation of your work. It is entirely automated and there is no way to negotiate with it. So if you want to avoid bad surprises, be as thorough as possible.
- Moulinette is not very open-minded. It won't try and understand your code if it doesn't respect the Norm. Moulinette relies on a program called norminette to check if your files respect the norm. TL;DR: it would be idiotic to submit a piece of work that doesn't pass norminette's check.
- These exercises are carefully laid out by order of difficulty from easiest to hardest. We will not take into account a successfully completed harder exercise if an easier one is not perfectly functional.
- Using a forbidden function is considered cheating. Cheaters get -42, and this grade is non-negotiable.
- You'll only have to submit a main() function if we ask for a program.
- Moulinette compiles with these flags: -Wall -Wextra -Werror, and uses gcc.
- If your program doesn't compile, you'll get 0.
- You <u>cannot</u> leave <u>any</u> additional file in your directory than those specified in the subject.
- Got a question? Ask your peer on the right. Otherwise, try your peer on the left.

- Your reference guide is called Google / man / the Internet /
- Check out the "C Piscine" part of the forum on the intranet, or the slack Piscine.
- Examine the examples thoroughly. They could very well call for details that are not explicitly mentioned in the subject...
- By Odin, by Thor! Use your brain!!!

Chapter II

Foreword

Here's what Wikipedia have 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.

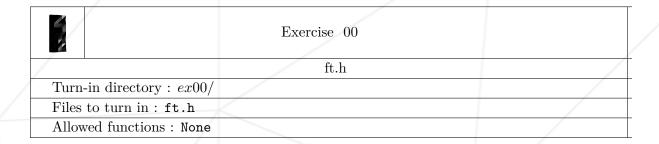
The unusual appearance of this egg-laying, duck-billed, beaver-tailed, otter-footed mammal baffled European naturalists when they first encountered it, with some considering it an elaborate hoax. It is one of the few venomous mammals, the male platypus having a spur on the hind foot that delivers a venom 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 absolutly not talking about platypus.

Chapter III

Exercise 00: ft.h

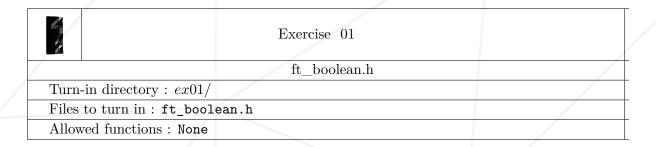


- Create your ft.h file.
- It should countain the prototypes of all the following functions:

```
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



• Create a ft_boolean.h file. It'll compile and run the following main appropriately:

• This program should display

I have an even number of arguments.

• or

I have an odd number of arguments.

 \bullet followed by a line break when adequate.



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

Chapter V

Exercise 02 : ft_abs.h

3	Exercise 02	
	${ m ft_abs.h}$	
Turn-in directory : $ex0$		
Files to turn in : ft_abs.h		
Allowed functions : Nor	ne	

• Create a macro ABS which replaces its argument by it absolute value:

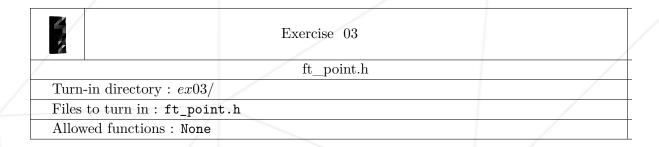
#define ABS(Value)



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

Chapter VI

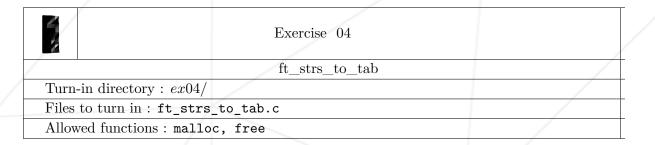
Exercise 03: ft_point.h



• Create a file ft_point.h that'll compile the following main:

Chapter VII

Exercise 04: ft_strs_to_tab



- Create a function that takes an array of string as 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 provided, like this:

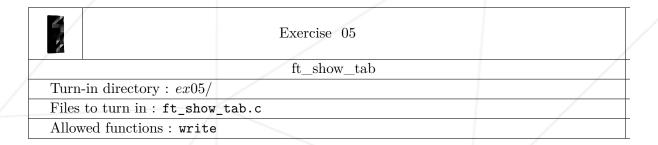
```
typedef struct s_stock_str
{
  int size;
  char *str;
  char *copy;
}
  t_stock_str;
```

- 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 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



- 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:
 - \circ the string followed by a '\n'
 - \circ 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!