

C Piscine C 01

Staff 42 pedago@42.fr

Summary: This document is the subject for the module C 01 of the C Piscine @ 42.

Contents

1	Instructions	2
II	Foreword	4
III	Exercise 00 : ft_ft	5
IV	Exercise 01 : ft_ultimate_ft	6
V	Exercise 02 : ft_swap	7
VI	Exercise 03 : ft_div_mod	8
VII	Exercise 04 : ft_ultimate_div_mod	9
VIII	Exercise 05 : ft_putstr	10
IX	Exercise 06 : ft_strlen	11
\mathbf{X}	Exercise 07 : ft_rev_int_tab	12
XI	Exercise 08 : ft_sort_int_tab	13

Chapter I

Instructions

- Only this page will serve as reference: do not trust rumors.
- Watch out! This document could potentially change up 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.

C Piscine

C 01

- 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!!!



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

Chapter II

Foreword

Vincent: And you know what they call a... a... a Quarter Pounder with Cheese in Paris?

Jules: They don't call it a Quarter Pounder with cheese?

Vincent: No man, they got the metric system. They wouldn't know what the fuck a Quarter Pounder is.

Jules: Then what do they call it?

Vincent: They call it a Royale with cheese.

Jules: A Royale with cheese. What do they call a Big Mac?

Vincent: Well, a Big Mac's a Big Mac, but they call it le Big-Mac.

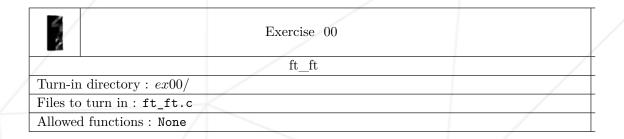
Jules: Le Big-Mac. Ha ha ha ha. What do they call a Whopper?

Vincent: I dunno, I didn't go into Burger King.

At least one of the following exercices has nothing to do you with a Royale with cheese.

Chapter III

Exercise 00: ft_ft

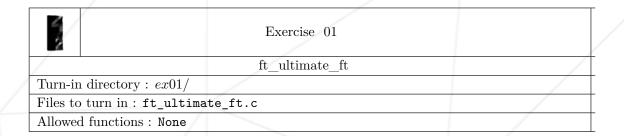


- Create a function that takes a pointer to int as a parameter, and sets the value "42" to that int.
- Here's how it should be prototyped :

void ft_ft(int *nbr);

Chapter IV

Exercise 01: ft_ultimate_ft



- Create a function that takes a pointer to int as a parameter and sets the value "42" to that int.
- Here's how it should be prototyped :

void ft_ultimate_ft(int *******nbr);

Chapter V

Exercise 02: ft_swap

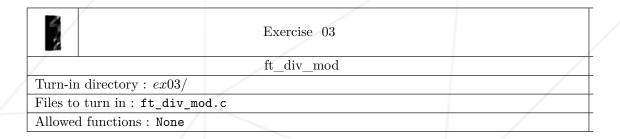
	Exercise 02	
/	ft_swap	
Turn-in directory : $ex02/$		
Files to turn in : ft_swap.c		
Allowed functions: None		

- Create a function that swaps the value of two integers whose addresses are entered as parameters.
- Here's how it should be prototyped :

void ft_swap(int *a, int *b);

Chapter VI

Exercise 03: ft_div_mod



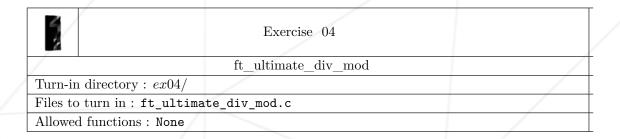
 \bullet Create a function ${\tt ft_div_mod}$ prototyped like this :

void ft_div_mod(int a, int b, int *div, int *mod);

• This function divides parameters a by b and stores the result in the int pointed by div. It also stores the remainder of the division of a by b in the int pointed by mod.

Chapter VII

Exercise 04: ft_ultimate_div_mod



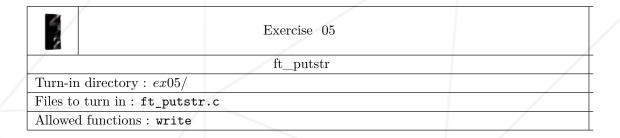
• Create a function ft_ultimate_div_mod with the following prototype :

void ft_ultimate_div_mod(int *a, int *b);

• This function divides parameters a by b. The result of this division is stored in the int pointed by a. The remainder of the division is stored in the int pointed by b.

Chapter VIII

Exercise 05: ft_putstr

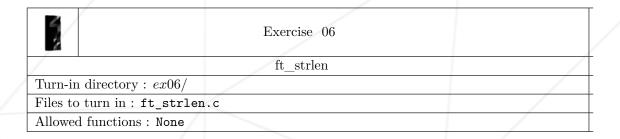


- Create a function that displays a string of characters on the standard output.
- Here's how it should be prototyped :

void ft_putstr(char *str);

Chapter IX

Exercise 06 : ft_strlen

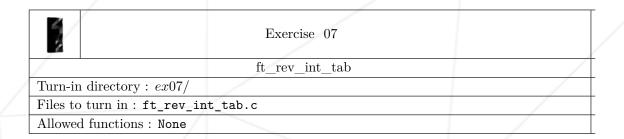


- Create a function that counts and returns the number of characters in a string.
- $\bullet\,$ Here's how it should be prototyped :

int ft_strlen(char *str);

Chapter X

Exercise 07: ft_rev_int_tab

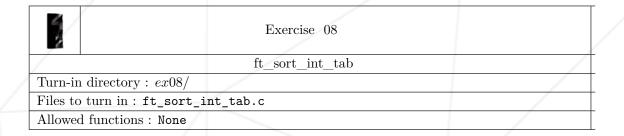


- Create a function which reverses a given array of integer (first goes last, etc).
- The arguments are a pointer to int and the number of ints in the array.
- Here's how it should be prototyped :

void ft_rev_int_tab(int *tab, int size);

Chapter XI

Exercise 08: ft_sort_int_tab



- Create a function which sorts an array of integers by ascending order.
- The arguments are a pointer to int and the number of ints in the array.
- Here's how it should be prototyped :

void ft_sort_int_tab(int *tab, int size);