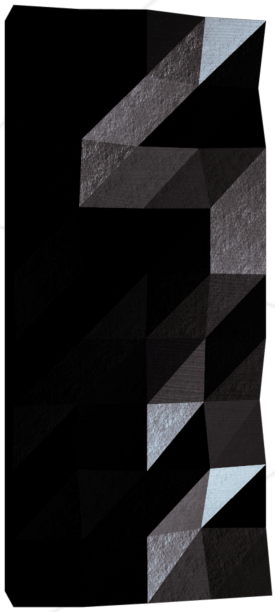

C Piscine

C 06

Staff 42 pedago@42.fr

Summary: This document contains the subject for the C 06 module of the C Piscine at 42.

Version: 7.4

Contents

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!!!

`Norminette must be run with the -R CheckForbiddenSourceHeader flag.`
`Moulinette will use it as well.`

Chapter II

Foreword

Dialog from the movie `The Big Lebowski`:

```
The Dude: Walter, ya know, it's Smokey, so his toe slipped over the line a little,
  big deal. It's just a game, man.
Walter Sobchak: Dude, this is a league game, this determines who enters the next
  round robin. Am I wrong? Am I wrong?
Smokey: Yeah, but I wasn't over. Gimme the marker Dude, I'm marking it 8.
Walter Sobchak: [pulls out a gun] Smokey, my friend, you are entering a world
  of pain.
The Dude: Walter...
Walter Sobchak: You mark that frame an 8, and you're entering a world of pain.
Smokey: I'm not...
Walter Sobchak: A world of pain.
Smokey: Dude, he's your partner...
Walter Sobchak: [shouting] Has the whole world gone crazy? Am I the only one
  around here who gives a shit about the rules? Mark it zero!
The Dude: They're calling the cops, put the piece away.
Walter Sobchak: Mark it zero!
[points gun in Smokey's face]
The Dude: Walter...
Walter Sobchak: [shouting] You think I'm fucking around here? Mark it zero!
Smokey: All right, it's fucking zero. Are you happy, you crazy fuck?
Walter Sobchak: ...It's a league game, Smokey.
```

Chapter III

Exercise 00 : ft_print_program_name

	Exercise 00
ft_print_program_name	
Turn-in directory : $ex00/$	
Files to turn in : `ft_print_program_name.c`	
Allowed functions : `write`	

- Since this is a program, your `.c` file must contain a `main` function.
- Write a program that displays its own name, followed by a new line.
- Example:

```
$>./a.out | cat -e
./a.out$
$>
```

Chapter IV

Exercise 01 : ft_print_params

	Exercise 01
	ft_print_params
Turn-in directory : $ex01/$	
Files to turn in : `ft_print_params.c`	
Allowed functions : `write`	

- Since this is a program, your `.c` file must contain a `main` function.
- Write a program that displays its given arguments.
- Each argument should be printed on a new line, in the same order as in the command line.
- The program should display all arguments except `argv[0]`.
- Example:

```
$>./a.out test1 test2 test3 | cat -e
test1$
test2$
test3$
$>
```

Chapter V

Exercise 02 : ft_rev_params

	Exercise 02
ft_rev_params	
Turn-in directory : $ex02/$	
Files to turn in : `ft_rev_params.c`	
Allowed functions : `write`	

- Since this is a program, your `.c` file must contain a `main` function.
- Write a program that displays its given arguments.
- Each argument should be printed on a new line, in reverse order from the command line.
- The program should display all arguments except `argv[0]`.

Chapter VI

Exercise 03 : ft_sort_params

	Exercise 03
ft_sort_params	
Turn-in directory : $ex03/$	
Files to turn in : `ft_sort_params.c`	
Allowed functions : `write`	

- Since this is a program, your `.c` file must contain a `main` function.
- Write a program that displays its given arguments, sorted in ASCII order.
- The program should display all arguments except `argv[0]`.
- Each argument should be printed on a new line.

Chapter VII

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. Be sure to double-check the names of your files to ensure they are correct.

`You need to return only the files requested by the subject of this project.`