C Piscine

Day 06

Staff 42 pedago@42.fr

Summary: This document is the subject for Day06 of the C Piscine @ 42.

Contents

Chapter 1

Instructions

- The exercises are carefully laid out in order of difficulty, from easiest to hardest. An exercise is only graded if all previous ones are correct. In other words: the grading for a day stops at the first mistake.
- Be mindful of the submission procedures indicated at the start of every exercise.
- 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. Be as thorough as possible!
- Moulinette relies on a program called **norminette** to check if your files respect the Norm. An exercise containing files that do not respect the Norm will be graded 0.
- Using a forbidden function is considered cheating. Cheaters get -42, and this grade is non-negotiable.
- If **ft_putchar()** is an authorized function, we will compile your code with our **ft_putchar.c**.
- 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, it will be graded 0.
- You should not leave any additional file in your directory than those specified in the subject.

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

The forewords are entirely unrelated to the subjects and can safely be ignored.

Chapter 2

Topics

Today, you will have to learn about:

- Function libraries
- Handling command line arguments

Chapter 3

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 4

Exercise 00 : libft

Turn-in directory : `ex00/`

Files to turn in: `libft_creator.sh, ft_putchar.c, ft_swap.c, ft_putstr.c, ft_strlen.c, ft_strcmp.c`

Allowed functions: `write`

- Create your **ft** library. It'll be called **libft.a**.
- A shell script called **libft_creator.sh** will compile source files appropriately and will create your library.
- This library should contain all of 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);
```

- We'll launch the following command-line :

```
sh libft_creator.sh
```

Don't hesitate to add other useful functions... ;-]

Chapter 5

Exercise 01 : ft_print_program_name

Turn-in directory : `ex01/`
Files to turn in: `ft_print_program_name.c`
Allowed functions: `ft_putchar`

- We're dealing with a program here, you should therefore have a function **main** in your **.c** file.
- Create a program that displays its own name.
- Example :

```
$>./a.out
./a.out
$>
```

Chapter 6

Exercise 02 : ft_print_params

Turn-in directory : `ex02/`

Files to turn in: `ft_print_params.c`

Allowed functions: `ft_putchar`

- We're dealing with a program here, you should therefore have a function **main** in your **.c** file.
- Create a program that displays its given arguments.
- Example :

```
$>./a.out test1 test2 test3
test1
test2
test3
$>
```

Chapter 7

Exercise 03 : ft_rev_params

Turn-in directory : `ex03/`
Files to turn in: `ft_rev_params.c`
Allowed functions: `ft_putchar`

- We're dealing with a program here, you should therefore have a function **main** in your **.c** file.
- Create a program that displays its given arguments in reverse order.
- It should display all arguments, except for **argv[0]**.
- All arguments have to have their own line.

Chapter 8

Exercise 04 : ft_sort_params

Turn-in directory : `ex04/`
Files to turn in: `ft_sort_params.c`
Allowed functions: `ft_putchar`

- We're dealing with a program here, you should therefore have a function **main** in your **.c** file.
- Create a program that displays its given arguments sorted by ascii order.
- It should display all arguments, except for **argv[0]**.
- All arguments have to have their own line.