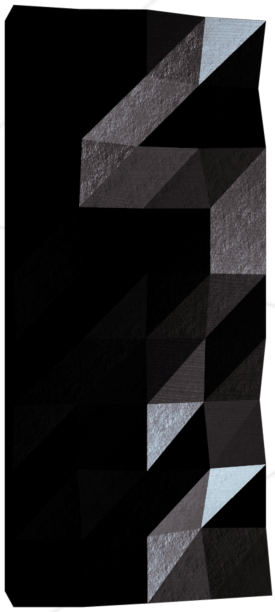

C Piscine

C 00

Summary: THIS document is the subject for the C 00 module of the C Piscine @ 42.

Version: 7.7

Contents

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 `cc`.
- If your program doesn't compile, you'll get `0`.
- You cannot leave any 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 !!!

Do not forget to add the *standard 42 header* in each of your .c/.h files. The norminette check its existence anyway!

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

Chapter II

Foreword

Cod liver oil is a nutritional supplement derived from liver of cod fish (Gadidae).

As with most fish oils, it has high levels of the omega-3 fatty acids, eicosapentaenoic acid (EPA) and docosahexaenoic acid (DHA).
Cod liver oil also contains vitamin A and vitamin D.

It has historically been taken because of its vitamin A and vitamin D content.

It was once commonly given to children, because vitamin D has been shown to prevent rickets and other symptoms of vitamin D deficiency.

Contrary to Cod liver oil, C is good, eat some!

Chapter III

Exercice 00: ft_putchar

	Exercise 00
ft_putchar	
Turn-in directory : *ex*00/	
Files to turn in : `ft_putchar.c`	
Allowed functions : `write`	

- Write a function that displays the character passed as a parameter.
- It will be prototyped as follows :

```
void ft_putchar(char c);
```

To display the character, you must use the `write` function as follows.

```
write(1, &c, 1);
```

The first retry delay is short, do not hesitate to trigger an intermediate evaluation to measure your progress.

Chapter IV

Exercise 01: ft_print_alphabet

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

- Create a function that displays the alphabet in lowercase, on a single line, by ascending order, starting from the letter 'a'.
- Here's how it should be prototyped :

```
void ft_print_alphabet(void);
```

Do not hesitate to pickup randomly someone in your cluster to ask a question.

Chapter V

Exercise 02: ft_print_reverse_alphabet

	Exercise 02
ft_print_reverse_alphabet	
Turn-in directory : *ex*02/	
Files to turn in : `ft_print_reverse_alphabet.c`	
Allowed functions : `write`	

- Create a function that displays the alphabet in lowercase, on a single line, by descending order, starting from the letter 'z'.
- Here's how it should be prototyped :

```
void ft_print_reverse_alphabet(void);
```

`Git push regularly.`

Chapter VI

Exercise 03: ft_print_numbers

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

- Create a function that displays all digits, on a single line, by ascending order.
- Here's how it should be prototyped :

```
void ft_print_numbers(void);
```

Collaboration is a key to success.

Chapter VII

Exercise 04: ft_is_negative

	Exercise 04
ft_is_negative	
Turn-in directory : *ex*04/	
Files to turn in : `ft_is_negative.c`	
Allowed functions : `write`	

- Create a function that displays 'N' or 'P' depending on the integer's sign entered as a parameter. If `n` is negative, display 'N'. If `n` is positive or null, display 'P'.
- Here's how it should be prototyped :

```
void ft_is_negative(int n);
```

`Failure is part of your learning journey.`

Chapter VIII

Exercise 05: ft_print_comb

	Exercise 05
ft_print_comb	
Turn-in directory : $ex05/$	
Files to turn in : `ft_print_comb.c`	
Allowed functions : `write`	

- Create a function that displays all different combinations of three different digits in ascending order, listed by ascending order - yes, repetition is voluntary.
- Here's the intended output :

```
$>./a.out | cat -e
012, 013, 014, 015, 016, 017, 018, 019, 023, ..., 789$>
```

- 987 isn't there because 789 already is.
- 999 isn't there because the digit 9 is present more than once.
- Here's how it should be prototyped :

```
void ft_print_comb(void);
```

Did you check with your right neighbor ?

Chapter IX

Exercise 06: ft_print_comb2

	Exercise 06
	ft_print_comb2
Turn-in directory : *ex*06/	
Files to turn in : `ft_print_comb2.c`	
Allowed functions : `write`	

- Create a function that displays all different combination of two two digits numbers (XX XX) between 00 and 99, listed by ascending order.
- Here's the expected output :

```
$>./a.out | cat -e
00 01, 00 02, 00 03, 00 04, 00 05, ..., 00 99, 01 02, ..., 97 99, 98 99$>
```

- Here's how it should be prototyped :

```
void ft_print_comb2(void);
```

Get inspired by others, do not let them do your job.

Chapter X

Exercise 07: ft_putnbr

	Exercise 07
ft_putnbr	
Turn-in directory : *ex*07/	
Files to turn in : `ft_putnbr.c`	
Allowed functions : `write`	

- Create a function that displays the number entered as a parameter. The function has to be able to display all possible values within an `int` type variable.
- Here's how it should be prototyped :

```
void ft_putnbr(int nb);
```

- For example:
  - `ft_putnbr(42)` displays "42".

Do not believe any source of information: always make your own tests, controls and verifications.

Chapter XI

Exercise 08: ft_print_combn

	Exercise 08
	ft_print_combn
Turn-in directory : *ex*08/	
Files to turn in : `ft_print_combn.c`	
Allowed functions : `write`	

- Create a function that displays all different combinations of `n` numbers by ascending order.
- n will be so that : $0 < n < 10$.
- If `n` = 2, here's the expected output :

```
$>./a.out | cat -e
01, 02, 03, ..., 09, 12, ..., 79, 89$>
```

- Here's how it should be prototyped :

```
void ft_print_combn(int n);
```

Did you check with your left neighbor ?

Chapter XII

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate 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.`