C Piscine

C 08

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

Version: 14

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!!!

Chapter II

AI Instructions

● Context

The C Piscine is intense. It's your first big challenge at 42 — a deep dive into problem-solving, autonomy, and community.

During this phase, your main objective is to build your foundation — through struggle, repetition, and especially **peer-learning** exchange.

In the AI era, shortcuts are easy to find. However, it's important to consider whether your AI usage is truly helping you grow — or simply getting in the way of developing real skills.

The Piscine is also a human experience — and for now, nothing can replace that. Not even AI.

For a more complete overview of our stance on AI — as a learning tool, as part of the ICT curriculum, and as a growing expectation in the job market — please refer to the dedicated FAQ available on the intranet.

● Main message

☛ Build strong foundations without shortcuts.

☛ Really develop tech & power skills.

☛ Experience real peer-learning, start learning how to learn and solve new problems.

☛ The learning journey is more important than the result.

☛ Learn about the risks associated with AI, and develop effective control practices and countermeasures to avoid common pitfalls.

● **Learner rules:**

- You should apply reasoning to your assigned tasks, especially before turning to AI.

- You should not ask for direct answers to the AI.

- You should learn about 42 global approach on AI.

● **Phase outcomes:**

Within this foundational phase, you will get the following outcomes:

- Get proper tech and coding foundations.

- Know why and how AI can be dangerous during this phase.

● **Comments and example:**

- Yes, we know AI exists — and yes, it can solve your projects. But you're here to learn, not to prove that AI has learned. Don't waste your time (or ours) just to demonstrate that AI can solve the given problem.

- Learning at 42 isn't about knowing the answer — it's about developing the ability to find one. AI gives you the answer directly, but that prevents you from building your own reasoning. And reasoning takes time, effort, and involves failure. The path to success is not supposed to be easy.

- Keep in mind that during exams, AI is not available — no internet, no smartphones, etc. You'll quickly realise if you've relied too heavily on AI in your learning process.

- Peer learning exposes you to different ideas and approaches, improving your inter-personal skills and your ability to think divergently. That's far more valuable than just chatting with a bot. So don't be shy — talk, ask questions, and learn together!

- Yes, AI will be part of the curriculum — both as a learning tool and as a topic in itself. You'll even have the chance to build your own AI software. In order to learn more about our crescendo approach you'll go through in the documentation available on the intranet.

✓ **Good practice:**

I'm stuck on a new concept. I ask someone nearby how they approached it. We talk for 10 minutes — and suddenly it clicks. I get it.

✗ **Bad practice:**

I secretly use AI, copy some code that looks right. During peer evaluation, I can't explain anything. I fail. During the exam — no AI — I'm stuck again. I fail.

Chapter III

Foreword

Here's what `Wikipedia` has 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 absolutely not talking about platypuses.

Chapter IV

Exercise 00 : ft.h

	Exercise 00
	ft.h
Turn-in directory: *ex00/*	
Files to turn in: `ft.h`	
Allowed functions: `None`	

- Create your `ft.h` file.

- It should contain 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 V

Exercise 01 : ft__boolean.h

	Exercise 01
	ft__boolean.h
Turn-in directory: *ex01/*	
Files to turn in: `ft_boolean.h`	
Allowed functions: `None`	

- Create a `ft_boolean.h` file. It will compile and run the following main appropriately:

```
#include "ft_boolean.h"

void        ft_putstr(char *str)
{
        while (*str)
                write(1, str++, 1);
}

t_bool      ft_is_even(int nbr)
{
        return ((EVEN(nbr)) ? TRUE : FALSE);
}

int             main(int argc, char **argv)
{
        (void)argv;
        if (ft_is_even(argc - 1) == TRUE)
                ft_putstr(EVEN_MSG);
        else
                ft_putstr(ODD_MSG);
        return (SUCCESS);
}
```

- This program should display:

```
    I have an even number of arguments.
```

- or:

```
    I have an odd number of arguments.
```

- followed by a line break when appropriate.

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

Chapter VI

Exercise 02 : ft__abs.h

	Exercise 02
	ft__abs.h
Turn-in directory: *ex02/*	
Files to turn in: `ft_abs.h`	
Allowed functions: `None`	

- Create a macro `ABS` which replaces its argument with its absolute value:

```
#define ABS(Value)
```

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

Chapter VII

Exercise 03 : ft_point.h

	Exercise 03
	ft_point.h
Turn-in directory: *ex03/*	
Files to turn in: `ft_point.h`	
Allowed functions: `None`	

- Create a file `ft_point.h` that will compile the following main:

```
#include "ft_point.h"

void        set_point(t_point *point)
{
        point->x = 42;
        point->y = 21;
}

int             main(void)
{
        t_point             point;

        set_point(&point);
        return (0);
}
```

Chapter VIII

Exercise 04 : ft__strs__to__tab

	Exercise 04
	ft__strs__to__tab
Turn-in directory: *ex04/*	
Files to turn in: `ft_strs_to_tab.c`	
Allowed functions: `malloc, free`	

- Create a function that takes an array of strings as an 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 provide, 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 be 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 IX

Exercise 05 : ft__show__tab

Exercise 05	
ft__show__tab	
Turn-in directory: *ex05/*	
Files to turn in: `ft_show_tab.c`	
Allowed functions: `write`	

- 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:

  ○ the string followed by a '\n'

  ○ 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 !

Chapter X

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. Make sure to double-check the filenames to ensure they are correct.

You must submit only the files required by the project instructions.