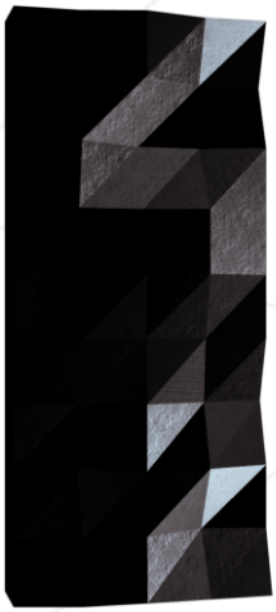

Piscine

04

Summ ry: this document is the subject for the unit C 04 of the C Piscine @ 42.

Version: 3

Contents

Ch pter I

Instructions

- Only this page will serve as reference: do not trust rumors.
- Watch out! This document could potentially change 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 cannot leave any additional file in your directory than those specified in the subject.
- Got a question? sk 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 !!!

`Norminette must be l unched with the -R CheckForbiddenSourceHe der fl g. Moulinette will use it too.`

Ch pter II

Foreword

Here are the lyrics for City Hunter's theme song "Moonlight Shadow":

```
The last time ever she saw him
Carried away by a moonlight shadow
He passed on worried and warning
Carried away by a moonlight shadow.
Lost in a riddle that Saturday night
Far away on the other side.
He was caught in the middle of a desperate fight
 nd she couldn't find how to push through

The trees that whisper in the evening
Carried away by a moonlight shadow
Sing a song of sorrow and grieving
Carried away by a moonlight shadow
 ll she saw was a silhouette of a gun
Far away on the other side.
He was shot six times by a man on the run
 nd she couldn't find how to push through

[Chorus]
I stay, I pray
See you in Heaven far away...
I stay, I pray
See you in Heaven one day.

Four  .M. in the morning
Carried away by a moonlight shadow
I watched your vision forming
Carried away by a moonlight shadow
  star was glowing in the silvery night
Far away on the other side
Will you come to talk to me this night
But she couldn't find how to push through

[Chorus]
```

```
Far away on the other side.
Caught in the middle of a hundred and five
The night was heavy and the air was alive
But she couldn't find how to push through
Carried away by a moonlight shadow
Carried away by a moonlight shadow
Far away on the other side.
```

Unfortunately, this topic has nothing to do with `City Hunter`.

Ch pter III

Exercise 00 : ft_strlen

	Exercise 00
ft_strlen	
Turn-in directory : $ex00$	
Files to turn in : `ft_strlen.c`	
llowed functions : `None`	

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

```
int    ft_strlen(ch r *str);
```

Ch pter IV

Exercise 01 : ft_putstr

	Exercise 01
ft_putstr	
Turn-in directory : $ex01$	
Files to turn in : `ft_putstr.c`	
llowed functions : `write`	

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

```
void    ft_putstr(ch r *str);
```

Ch pter V

Exercise 02 : ft_putnbr

	Exercise 02
ft_putnbr	
Turn-in directory : *ex*02	
Files to turn in : `ft_putnbr.c`	
llowed 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`".

Ch pter VI

Exercise 03 : ft_ toi

	Exercise 03
ft_atoi	
Turn-in directory : *ex*03	
Files to turn in : `ft_ toi.c`	
llowed functions : `None`	

- Write a function that converts the initial portion of the string pointed by str to its int representation
- The string can start with an arbitray amount of white space (as determined by `issp ce(3)`)
- The string can be followed by an arbitrary amount of + and - signs, - sign will change the sign of the int returned based on the number of - is odd or even.
- Finally the string can be followed by any numbers of the base 10.
- Your function should read the string until the string stop following the rules and return the number found until now.
- You should not take care of overflow or underflow. result can be undefined in that case.
- Here's an example of a program that prints the atoi return value:

```
$>./ .out "   ---+--+1234 b567"
-1234
```

- Here's how it should be prototyped :

```
int     ft_ toi(ch r *str);
```

Ch pter VII

Exercise 04 : ft_putnbr_b se

	Exercise 04
ft_putnbr_base	
Turn-in directory : $ex04$	
Files to turn in : `ft_putnbr_b se.c`	
llowed functions : `write`	

- Create a function that displays a number in a base system in the terminal.
- This number is given in the shape of an `int`, and the radix in the shape of a `string of characters`.
- The base-system contains all useable symbols to display that number :

  `0123456789` is the commonly used base system to represent decimal numbers

  `01` is a binary base system ;

  `0123456789 BCDEF` an hexadecimal base system ;

  `poneyvif` is an octal base system.

- The function must handle negative numbers.
- If there's an invalid argument, nothing should be displayed. Examples of invalid arguments :

  base is empty or size of 1;

  base contains the same character twice ;

  base contains + or - ;

- Here's how it should be prototyped :

```
void        ft_putnbr_b se(int nbr, ch r *b se);
```

Ch pter VIII

Exercise 05 : ft_ toi_b se

	Exercise 05
ft_atoi_base	
Turn-in directory : $ex05$	
Files to turn in : `ft_ toi_b se.c`	
llowed functions : `None`	

- Write a function that converts the initial portion of the string pointed by str to int representation.
- str is in a specific base given as a second parameter.
- excepted the base rule, the function should work exactly like ft_atoi.
- If there's an invalid argument, the function should return 0. Examples of invalid arguments :

  base is empty or size of 1;

  base contains the same character twice ;

  base contains + or - or whitespaces;

- Here's how it should be prototyped :

```
int        ft_ toi_b se(ch r *str, ch r *b se);
```