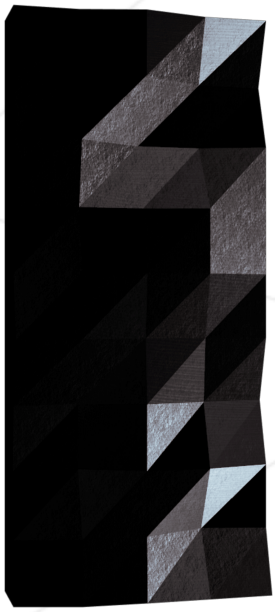

C Piscine

Day 07

Staff 42 pedago@42.fr

Abstract: This document is the subject for Day07 of the C Piscine @ 42.

Contents

Chapter I

Instructions

- Only this page will serve as reference: do not trust rumors.
- Watch out! This document could potentially change up to an hour before submission.
- Make sure you have the appropriate permissions on your files and directories.
- You have to follow the submission procedures for 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. 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 `Norminator` to check if your files respect the norm. TL;DR: it would be idiotic to submit a piece of work that doesn't pass `Norminator`'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.
- 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, 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 your right. Otherwise, try your peer on your left.
- Your reference guide is called `Google / man / the Internet / ...`.
- Check out the "C Piscine" part of the forum on the intranet.
- 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 !!!

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

Chapter II

Foreword

Morty: Rick!

Rick: Uhp-uhp-uhp! Morty, keep your hands off your ding-dong! It's the only way we can speak freely. Look around you, Morty. Do you really think this wuh-world is real? You'd have to be an idiot not to notice all the sloppy details. Look, that guy's putting a bun between two hot dogs.

Morty: I dunno, Rick, I mean, I've seen people do that before.

Rick: Well, look at that old lady. She's-she's walking a cat on a leash.

Morty: Uh, Mrs. Spencer does that all the time, Rick.

Rick: Look, I-I-I don't want to hear about Mrs. Spencer, Morty! She's an idiot! All right, all right, there. Wh-what about that, Morty?

Morty: Okay, okay, you got me on that one.

Rick: Oh, really, Morty? Are you sure you haven't seen that somewhere in real life before?

Morty: No, no, I haven't seen that. I mean, why would a Pop-Tart want to live inside a toaster, Rick? I mean, th-that would be like the scariest place for them to live. Y'know what I mean?

Rick: You're missing the point, Morty. Why would he drive a smaller toaster with wheels? I mean, does your car look like a smaller version of your house? No.

Morty: So, why are they doing this? W-what do they want?

Rick: Well, that would be obvious to you, Morty, if you'd been paying attention.
[an ambulance drives past Rick and Morty and stops; open back doors]
Paramedic: We got the President of the United States in here! We need 10cc of concentrated dark matter, stat, or he'll die!

Morty: Concentrated dark matter? They were asking about that in class.

Rick: Yeah, it’s a special fuel I invented to travel through space faster than anybody else. These Zigerions are always trying to scam me out of my secrets, but they made a big mistake this time, Morty. They dragged you into this. Now they’re gonna pay!

Morty: What do you- w-w-what are we gonna do?

Rick: We’re gonna scam the scammers, Morty. And we’re gonna take ’em for everything they’ve got.

The following exercices will be easier to complete if you are a fan of "Rick and Morty"

Chapter III

Exercise 00 : ft_strdup

	Exercice : 00
ft_strdup	
Turn-in directory : $ex00/$	
Files to turn in : `ft_strdup.c`	
Allowed functions : `malloc`	
Remarks : `n/a`	

- Reproduce the behavior of the function `strdup` (man strdup).
- Here's how it should be prototyped :

```
char    *ft_strdup(char *src);
```

Chapter IV

Exercise 01 : ft_range

	Exercice : 01
ft_range	
Turn-in directory : *ex*01/	
Files to turn in : `ft_range.c`	
Allowed functions : `malloc`	
Remarks : `n/a`	

- Create a function `ft_range` which returns an array of`ints`. This `int` array should contain all values between `min` and `max`.
- `Min` included - `max` excluded.
- Here's how it should be prototyped :

```
int    *ft_range(int min, int max);
```

- If `min`́value is greater or equal to `max`'s value, a null pointer should be returned.

Chapter V

Exercise 02 : ft_ultimate_range

	Exercice : 02
ft_ultimate_range	
Turn-in directory : *ex*02/	
Files to turn in : `ft_ultimate_range.c`	
Allowed functions : `malloc`	
Remarks : `n/a`	

- Create a function `ft_ultimate_range` which allocates and assigns an array of `ints`. This `int` array should contain all values between `min` and `max`.
- `Min` included - `max` excluded.
- Here's how it should be prototyped :

```
int    ft_ultimate_range(int **range, int min, int max);
```

- If the value of `min` is greater or equal to `max`'s value, `range` will point on NULL.
- The size of `range` should be returned (or 0 on error).

Chapter VI

Exercise 03 : ft_concat_params

	Exercice : 03
ft_concat_params	
Turn-in directory : $ex03/$	
Files to turn in : `ft_concat_params.c`	
Allowed functions : `malloc`	
Remarks : `n/a`	

- Create a function that transforms arguments given as command-line into a single string of characters. Those arguments should be separated by a "\n".
- Here's how it should be prototyped :

```
char *ft_concat_params(int argc, char **argv);
```

Chapter VII

Exercise 04 : ft_split_whitespaces

	Exercice : 04
ft_split_whitespaces	
Turn-in directory : $ex04/$	
Files to turn in : `ft_split_whitespaces.c`	
Allowed functions : `malloc`	
Remarks : `n/a`	

- Create a function that splits a string of characters into words.
- Separators are spaces, tabs and line breaks.
- This function returns an array where each box contains a character-string's address represented by a word. The last element of this array should be equal to 0 to emphasise the end of the array.
- There can't be any empty strings in your array. Draw the necessary conclusions.
- The given string can't be modified.
- Here's how it should be prototyped :

```
char **ft_split_whitespaces(char *str);
```

Chapter VIII

Exercise 05 : ft_print_words_tables

	Exercice : 05
ft_print_words_tables	
Turn-in directory : $ex05/$	
Files to turn in : `ft_print_words_tables.c`	
Allowed functions : `ft_putchar`	
Remarks : `n/a`	

- Create a function that displays the content of the array you created in the last excercise's function.
- One word per line.
- Each word will be followed by a "\n", including the last one.
- This exercise will be compiled with your ft_split_whitespaces.c
- Watch out not to have `multiple define`.
- Here's how it should be prototyped :

```
void ft_print_words_tables(char **tab);
```

Chapter IX

Exercise 06 : ft_convert_base

	Exercice : 06
ft_convert_base	
Turn-in directory : *ex*06/	
Files to turn in : `ft_convert_base.c`	
Allowed functions : `malloc, free`	
Remarks : `n/a`	

- Create a function that returns the result of the conversion of the string `nbr` from a base `base_from` to a base `base_to`. The string must have enough allocated memory. The number represented by `nbr` must fit inside an `int`.

- Here's how it should be prototyped :

```
char *ft_convert_base(char *nbr, char *base_from, char *base_to);
```

Chapter X

Exercise 07 : ft_split

	Exercice : 07
ft_split	
Turn-in directory : $ex07/$	
Files to turn in : `ft_split.c`	
Allowed functions : `malloc`	
Remarks : `n/a`	

- Create a function that slits a string of character depending on another string of characters.
- You'll have to use each character from the string `charset` as a separator.
- The function returns an array where each box contains the address of a string wrapped between two separators. The last element of that array should equal to 0 to indicate the end of the array.
- There cannot be any empty strings in your array. Draw your conclusions accordingly.
- The string given as argument won't be modifiable.
- Here's how it should be prototyped :

```
char **ft_split(char *str, char *charset);
```