CodeInActionLab

Python-01

Summary: *This document is the subject for the Python-01 module of the CodeInActionLab @ aba2020.*

Version: 2.1

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.

- 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 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 / ...`.

- 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 !!!

Chapter II

Foreword

Here is a discuss extract from the `Silicon Valley` serie:

```
- I mean, why not just use Vim over Emacs? (CHUCKLES)
- I do use Vim over Emac.
- Oh, God, help us! Okay, uh you know what? I just don't think this is
  going to work. I'm so sorry. Uh, I mean like, what, we're going to bring
  kids into this world with that over their heads? That's not really fair
  to them, don't you think?
- Kids? We haven't even slept together.
- And guess what, it's never going to happen now, because there is no
  way I'm going to be with someone who uses spaces over tabs.
- Richard! (PRESS SPACE BAR MANY TIMES)
- Wow. Okay. Goodbye.
- One tab saves you eight spaces! - (DOOR SLAMS) - (BANGING)


. . .


(RICHARD MOANS)

- Oh, my God! Richard, what happened?
- I just tried to go down the stairs eight steps at a time. I'm okay, though.
- See you around, Richard.
- Just making a point.
```

Hopefully, you are not forced to use emacs and your space bar to complete the following exercices.

Chapter III

Exercise 00 : args

	Exercise 00

Turn-in directory : *ex00/*
Files to turn in : `ft_min.py`
Allowed functions : `exit, len, print`

- Write a program that contains a function that prints the relation between number x and number y. You have to take the numbers as parameters. You will also need to manage your program with functions

Function name:

```
def ft_min():
```

```
42~ > python3 ft_min.py 24 42
24 is less than 42
42~ >
```

```
42~ > python3 ft_min.py 17 12
17 is greater than 12
42~ >
```

```
42~ > python3 ft_min.py 26 26
26 is equal to 26
42~ >
```

```
42~ > python3 ft_min.py 26
Error! Usage: python3 ft_min.py x y
42~ >
```

Chapter IV

Exercise 01 : ft__abs

	Exercise 01
	ft__abs
Turn-in directory : *ex*01/	
Files to turn in : `ft_abs.py`	
Allowed functions : `abs, eval, exit, len, print`	

- Write a program that take in input a valid mathematical expression and prints the absolute value of the result.

Function name:

```
def ft_abs():
```

```
42~ > python3 ft_abs.py
Insert your expression: ((10*9)/15)+((17*12)/6)+2
The result is: 42
42~ >
```

Chapter V

Exercise 02 : ft_max

	Exercise 02
	ft_max
Turn-in directory : *ex02/*	
Files to turn in : `ft_max.py`	
Allowed functions : `exit, len, max, print`	

- Write a program that takes 3 numbers in input and prints the maximum

Function name:

```
def ft_max():
```

```
42~ > python3 ft_max.py 17 12 42
The maximum is: 42
42~ >
```

```
42~ > python3 ft_max.py 17 12
Error! Usage: python3 ft_max.py <x> <y> <z>
42~ >
```

Chapter VI

Exercise 03 : ft_str_max

	Exercise 03
ft_str_max	
Turn-in directory : *ex03/*	
Files to turn in : `ft_str_max.py`	
Allowed functions : `input, len, max, print`	

- Write a program that takes two optional arguments and ask the user to enter a string. Print the maximum of the three values. If the user does not enter any parameters you will have to manage it.

Function name:

```
def ft_str_max():
```

```
42~ > python3 ft_str_max.py hello world
Insert a string: input
The maximum is: world
42~ >
```

```
42~ > python3 ft_str_max.py
Insert a string: input
The maximum is: input
42~ >
```

Chapter VII

Exercise 04 : ft_str_man

![icon]	Exercise 04

ft_str_man
Turn-in directory : *ex*04/
Files to turn in : **ft_str_man.py**
Allowed functions : **exit, input, len, print**

- Write a program that takes an argument **N** and ask the user to enter two strings **S1** and **S2**

    ○ If **N** is less that the length of **S1**, the program prints the string obtained by inserting **S2** in **S1** between position **N** and **N + 1**

    ○ Otherwise, the program prints an error message reporting both **N** and the length of **S1**

Function name:

```python
def ft_str_man():
```

```
42~ > python3 ft_str_man.py 1
Insert the first argument: My name is Myriam
Insert the second argument: and i live in Rome
Output: My nand i live in Romeame is Myriam
42~ >
```

> ⚠ Remember, you have to manage errors. Insert a custom message that begin with "Error!"

Chapter VIII

Exercise 05 : ft_summorial

	Exercise 05
	ft_summorial
Turn-in directory : *ex05/*	
Files to turn in : `ft_summorial.py`	
Allowed functions : `exit, len, print, range`	

- Write a program that takes as argument an integer **n** and prints the sum of all positive integers less than n

Function name:

```
def ft_summorial():
```

```
42~ > python3 ft_summorial.py 7
The sum of all positive numbers is 21
42~ >
```

Chapter IX

Exercise 06 : ft_sum_n

	Exercise 06
	ft_sum_n
Turn-in directory : *ex06/*	
Files to turn in : `ft_sum_n.py`	
Allowed functions : `exit, input, len, print, range`	

- Write a program that takes as argument an integer **n** and ask the user to input **n** integers. Then print their sum

Function name:

```python
def ft_sum_n():
```

```
42~ > python3 ft_sum_n.py 7
Insert the 1  number: 17
Insert the 2  number: 12
Insert the 3  number: 42
Insert the 4  number: 21
Insert the 5  number: 1337
Insert the 6  number: 19
Insert the 7  number: 1
The sum of all positive numbers is 1449
42~ >
```

Chapter X

Exercise 07 : ft_multiplication

	Exercise 07
	ft_multiplication
Turn-in directory : *ex07/*	
Files to turn in : `ft_multiplication.py`	
Allowed functions : `exit, len, print, range`	

- Write a program that takes as arguments two integers and computes the product x*y **without** the product operator, but iterating the sum.

Function name:

```
def ft_multiplication():
```

```
42~ > python3 ft_multiplication.py 10 3
Result: 30
42~ >
```

```
42~ > python3 ft_multiplication.py 10
Error! Usage: python3 ft_multiplication.py <x> <y>
42~ >
```

Chapter XI

Exercise 08 : ft_factorial

	Exercise 08
	ft_factorial
Turn-in directory : *ex08/*	
Files to turn in : `ft_factorial.py`	
Allowed functions : `exit, len, print, range`	

- Write a function that takes as argument an integer **N** and prints the factorial of n.

Function name:

```
def ft_factorial():
```

```
42~ > python3 ft_factorial.py 3
The factorial of 3 is 6
42~ >
```

```
42~ > python3 ft_factorial.py
Error! Usage: python3 ft_factorial.py <n>
42~ >
```

Chapter XII

Exercise 09 : ft__read__loop

	Exercise 09
	ft__read__loop
Turn-in directory : *ex09/*	
Files to turn in : `ft_read_loop.py`	
Allowed functions : `exit, len, print, range`	

- Write a function that iteratively reads and sums integers until the user enters 0. In that moment, the function prints the sum

Function name:

```
def ft_read_loop():
```

```
42~ > python3 ft_read_loop.py
Insert the 1  number: 17
Insert the 2  number: 12
Insert the 3  number: 42
Insert the 4  number: 21
Insert the 5  number: 1337
Insert the 6  number: 19
Insert the 7  number: 0
The sum of all positive numbers is 1448
42~ >
```

Chapter XIII

Exercise 10 : ft__division

	Exercise 10
	ft__division

Turn-in directory : *ex10/*
Files to turn in : `ft_division.py`
Allowed functions : `exit, len, print, range`

- Write a function that takes as arguments two integers **n** and **m** and prints the quotient and remainder of the division of **n** by **m**, without using the corresponding operators.

Function name:

```python
def ft_division():
```

```
42~ > python3 ft_division.py 10 2
The quotient is: 5
The reminder is: 0
42~ >
```

Chapter XIV

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.