Tutoring M1 - Bash & Python

Amélie Gruel

September 2019

1 Part 1 - Bash

In order to start this tutoring session, we'll begin by creating a directory on the Desktop using bash commands, that we'll name `Tutoring`. Navigate inside this directory, and there create another directory named `Tutoring1`.

Afterwards, stroll around your different directories. A little tip : it would be best to have a main directory titled `M1`, which would have a sub-directory named `S1` and sub-sub-directories corresponding to each of your classes !

If you have not yet organised your workspace, create at least a directory `M1` in your "Documents", and move there the `Tutoring` directory that you created earlier on your Desktop.

Navigate inside this directory, then inside the `Tutoring1` one : once inside, create a file named `exercise1`.

Finally, you've changed your mind and want to bring joy in your terminal : rename your `Tutoring` directory in `TutoringByTheBestM2InTheWorld` (yes I swear, it does spark joy in your life !).

Now that you have arranged your desktop in order to work in optimal conditions, you can finally tackle Python : modify the name of your `exercise1` file un order to use it as a Python script.

2 Part 2 - Python : the essentials

2.1 Exercise 1

Open your `exercise1.py` file in a code editor to start coding in Python.

> Tip : there are many commands allowing you to open your script in a specific editor directly from your terminal. Such as :
>
> - `code nomdufichier` to open your script in Visual Studio Code
> - `atom nomdufichier &` to open your script in Atom
> - `emacs nomdufichier &` to open your script in Emacs

Question a

Write a script that asks the user to enter an integer, then displays successively the integers going from 1 up to this number, with an increment of 1.

The terminal will display 1, 2, 3, etc until it reaches the given number. Little tip : you have to use a "for" loop.

Question b

Pick up your precedent script, and add some code in order to indicate whether each number is even or odd.

Little tip : you have to use "if"/"else" logic.

2.2 Exercise 2

Question a

Write a script that picks a random integer between 1 and 10, then asks the user to enter a number till they guess the correct one. As long as the user doesn't find the right number, the terminal will ask them for a new number. Once they find the correct number, the terminal will display a message congratulating them.

Little tip : you need to use a "while" loop.

Question b

Pick up your precedent script, and modify it. After each failed try, it now has to indicate to the user whether their number was too high or too low compared to the random number.

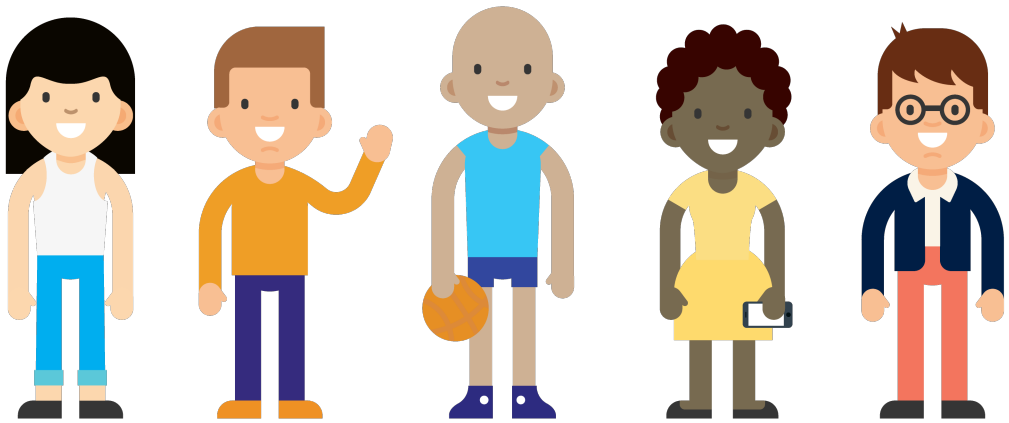
2.3 Exercise 3 - BONUS

Fibonacci's series is a recursive series that works as such :

$$\begin{cases} U_{n+2} &= U_{n+1} + U_n \\ U_0 &= 0 \\ U_1 &= 1 \end{cases}$$

Create a program asking the user for which n they want to get the corresponding U_n, then display the result.

3 Part 3 - Python : who's in this master already ?



3.1 Exercise 1

Create a list named `master`, which will contain the name of 5 Master2 students. Then, add the name of a teacher to this list.

In the end, you realize that the M2 by themselves are way cooler. Ergo you remove the teacher from the list ! Warning : do it without rewriting your whole list.

Finally, display the content of your list in your terminal, as well as its length.

Each one of these actions has to take place in the same script, on successive lines.

3.2 Exercise 2

At present, we would like to identify every member of the Master, whether they're M1, M2, M3 or even teachers !

But what should we use in order to store their names, and join to those names the corresponding status (M1, M2, teacher) ?

Use this data structure, that we will name `complete_master`, in order to identify at least 6 members.

After initializing it, add another member to it.

Afterwards, display the status of a person of your choice.

Remove a member of your choice.

Now, display the content of your whole structure.

Finally, display solely the students and teachers, then solely the status. This is possible using a method related to your data structure.

Just like in Exercise 1, each one of these actions has to take place in the same script, on successive lines.

3.3 Exercise 3 - BONUS

The user would like to identify the members of their Master however they like, by chosing progressively the actions they want to do. Therefore, write a program with a menu, in which the user will chose between :

1. creating an empty dictionary
2. adding the name and status of a member of the Master
3. removing the member of their choice
4. displaying the content of the dictionary. Each name will be displayed on a different line, followed by the corresponding bracketed status.
5. getting a specific member's name. The terminal will ask the user for a name, then return the corresponding status.
6. getting the name of every member with a particular status. The terminal will ask the user for a status, then return the corresponding members.
7. exiting the menu.

Once you've reached this point, please tell a M2 in order for them to explain to you how to create a menu.

4 Part 4 - Python : a little stroll inside the CREMI

It's officiel, you have taken your first classes in the CREMI. You are finally starting to get your bearings inside the building. But catastrophe !! It's 8h12, you're in the correct room but you've just got an email saying the class takes place 2 floors up !

You have to hurry there. Model the CREMI using a list of lists. This 2-dimentional list generates a 10x10 grid, within which you'll travel. Your initial position on (x=5;y=5) will be marked by a special character (such as "X").

Create a function displaying the content of your list on your terminal. One way to represent this can be seen in figure 1.

```
(base) amelie@L-E7-SAMMY:~/Documents/M2/Tutorats$ python2 PetitTourDansLeCREMI.py
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 X 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

Figure 1: Terminal displaying the grid depicting the CREMI. The 0 indicate boxes within which you can travel, the "X" your current position. Here, you are situated on x=5 and y=5.

Once this is done, implement a function that asks you in which direction you want to travel, then carry out this action. At the end of each move, the new grid will be displayed in the terminal (you can call the precedent function).

Warning ! You cannot travel outside your grid.

5 Part 5 - Python : once upon a time

Download the file "partie5.txt", which contains the script of an American cinematic masterpiece. Before someone talks, their name is indicated this way : `"> NAMECHARACTER"`, on one line. Save those names inside a list, in order to know which characters take part in the dialogue. Warning : some characters talks more than once, but we only want to have their name once in our list !

We now want to know who is the main character : to do so, create a function that browses your file one line at the time and counts how many times the name of each character appears after a ">". The main character will be the one whose name appears the most.

In order to do this, use a dictionary : the key will be the name of a character, and the value of the corresponding key will be increased by 1 for each word read.

Finally, you want the program to tell you who is the main character, you don't want to have to look at the dictionary. Add a function to your script, which will find out which character talks the most (meaning which character has the biggest value in the dictionnary).

Now that you know who speaks the most, save the content of your dictionnary inside a `blablabla.txt` file. The characters have to be saved according to the number of time they speak : the one who speaks the most is saved on the first line, and the one who speaks the less on the last line.