

Day 3 - Regex & Chart

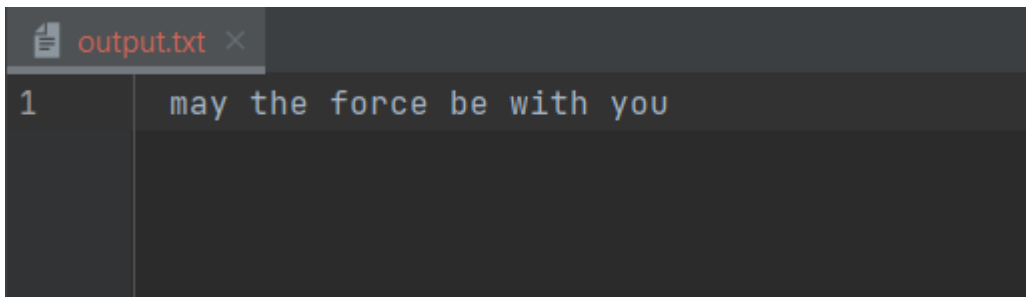
Python, files, charts and regex



Job 3.0

Create a program that asks the user to fill in a character string and writes this string to an "output.txt" file.

```
Entrer une chaîne de caractères may the force be with you
```



```
output.txt x
1 may the force be with you
```

Job 3.1

Create a program that reads the contents of the "**output.txt**" file and displays it in the terminal.

```
may the force be with you  
  
Process finished with exit code 0
```

Job 3.2

Create a program that browses the contents of the "[domains.xml](#)" file and counts the number of domain extensions found in it (.com, .net, etc ...).

Job 3.3

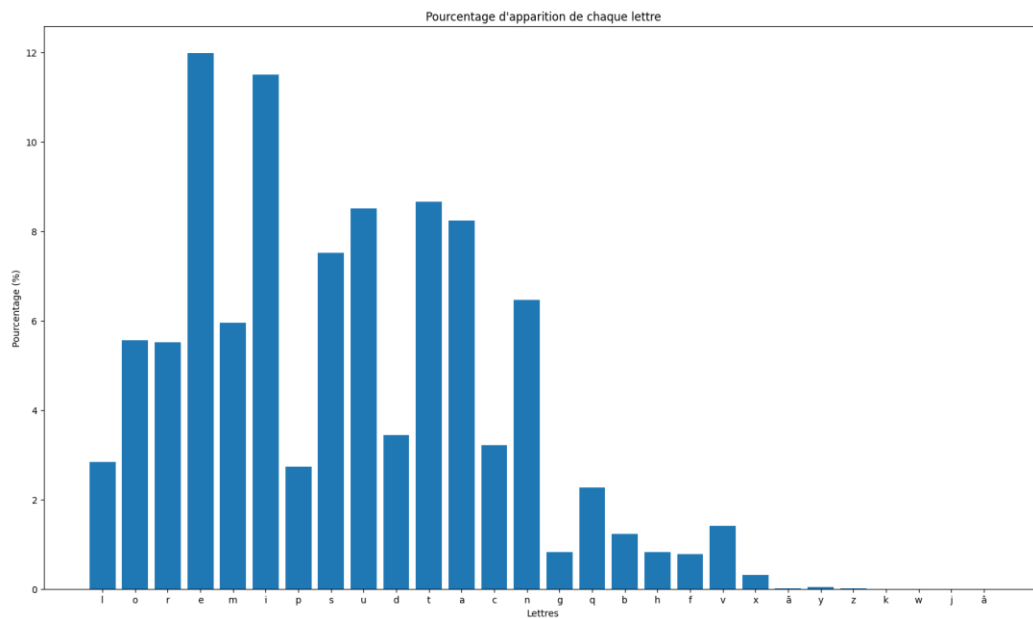
Create a program that browses the contents of the "[data.txt](#)" file and counts the number of words (without special characters) in it.

Job 3:4

Create a program that prompts the user to fill in a whole number. The program will then have to browse the content of the "**data.txt**" file count the number of words of the size entered that are there.

Job 3.5

Write a program that browses the "**data.txt**" file and counts the number of occurrences of each letter (lowercase and capital count as the same letter). Using the **Matplotlib** module, generate a histogram representing the percentage of appearance of each letter.



Job 3.5

Write a program that browses the "**data.txt**" file and counts the number of words of each size. Using the **Matplotlib** module, generate a histogram representing the percentage of appearance of each word size.

Job 3:6

Write a program that goes through the "**data.txt**" file and counts the number of occurrences of each letter (Lowercase and Capitals count as the same letter) at the beginning of a word. Using the **Matplotlib module**, generate a histogram representing the percentage of presence of each letter at the beginning of a word.

Job 3:7

Write a program that browses the "**data.txt**" file and counts the number of occurrences of the next letter for each letter. Then, generate **a graph of superimposed curves**, one curve per letter, showing the percentage of appearance of each letter following it.

For example, for the a: a(2%), b(5%), c(2.3%) For B: A(3%), B(0%), C(1%), ...

Job 3:8

Create a word generator based on previously calculated statistics (word length, first letter of words, sequence of letters).

Job 3.9

By analyzing the "data.txt" **file again**, establish statistics summarizing the number of words per sentence. Produce a **histogram** showing these statistics, and then, using your word generator, create a "Lorem Ipsum" sounding phrase generator.

Job 3.9 Bonus

An English Pokémon hides in "data.txt" ... Which one is it?

Rendering

In your github directory "**runtrack-python**", create a folder "**day03**" and in this folder, for each job, a folder "**jobXX**" where XX is the number of the job.
Don't forget to submit your changes as soon as a step is advanced or completed and use explicit comments.

Competencies targeted

- Manipulating files in python
- Mastering Regex in python
- Install and use the Matplotlib graphics library

Knowledge Base

- [Matplotlib](#)
Official matplotlib website, documentation and download.
- [Python File Open](#)
Manipulate files in python
- [Regex Python](#)
Documentation for using Regex in python