Python Lab 5.1: Python Collections

> The purpose of this practice is to help you apply the concepts discussed up to **now**:
>
> - Choose a approprate data collection to solve a given problem
> - Use collection methods to manage elements
> - Iterate over collection elements

In `lab5_1.py` in the text editor at top-right, write a program which will read a file and calculate letter frequency. At the end it should print the letters with the largest frequency. Your program should convert all characters to uppercase and only count the letters A-Z, ignoring spaces or punctuation marks or numbers.

You can use the text file `article1.txt` provided in the workspace as sample input, but you can upload your own text file if you want...

{% next %}

Algorithm

The first thing we have to decide, before solving this exercise is what kind of data structure we will use... Since we have to count stuff... a dictionary would seem appropriate...

- Step 1: open the file and read all contents in a text variable
- Step 2: convert it to uppercase

▶ HINT 1: Read File and Convert Text to Uppercase

```
text = fhand.read()
text = text.upper()
```

- Step 3: Loop through each character in the loaded text variable
- Step 4: Check if the current character is in the list for uppercase A-Z Characters
    - If yes, then look for that character in the letter_frequency dictionary and increment its count by 1
    - If no, proceed to the next character

▶ HINT 2: loop, search and count

```
letter_frequency = {}
for ch in text:
    if ch in az_Upper:
        letter_frequency[ch] = letter_frequency.get(ch, 0) +1
```

```
print(letter_frequency) # to test the outcome of this operation
```

- Step 5: Create an empty list
- Step 6: Loop through the values in the dictionary and append to the list a tuple with first the count and then the character
- Step 7: Sort the list in reverse order, with the hightest frequency appearing first

▶ HINT 3: Reverse the dictionary keys-values

```
lst = []
for key, value in letter_frequency.items():
    lst.append((value,key))

lst.sort(reverse=True)
```

- Step 8: Print the first tuple in the list

{% next %}

Execute your program

Remove any other output we used for testing appart from the final output.

For the article1.txt the correct output should be: E 661

Remember in order to execute your code you type in the terminal:

```
python lab5_1.py
```

Check that your code produces correct results.

Check Your Code

Execute the below to evaluate the correctness of your code using `check50`, but be sure to test it yourself also.

```
check50 mkotsovoulou/ods6001a/main/labs/lab5_1
```

Execute the below to evaluate the style of your code using `style50`.

```
style50 lab5_1.py
```

{% next %}

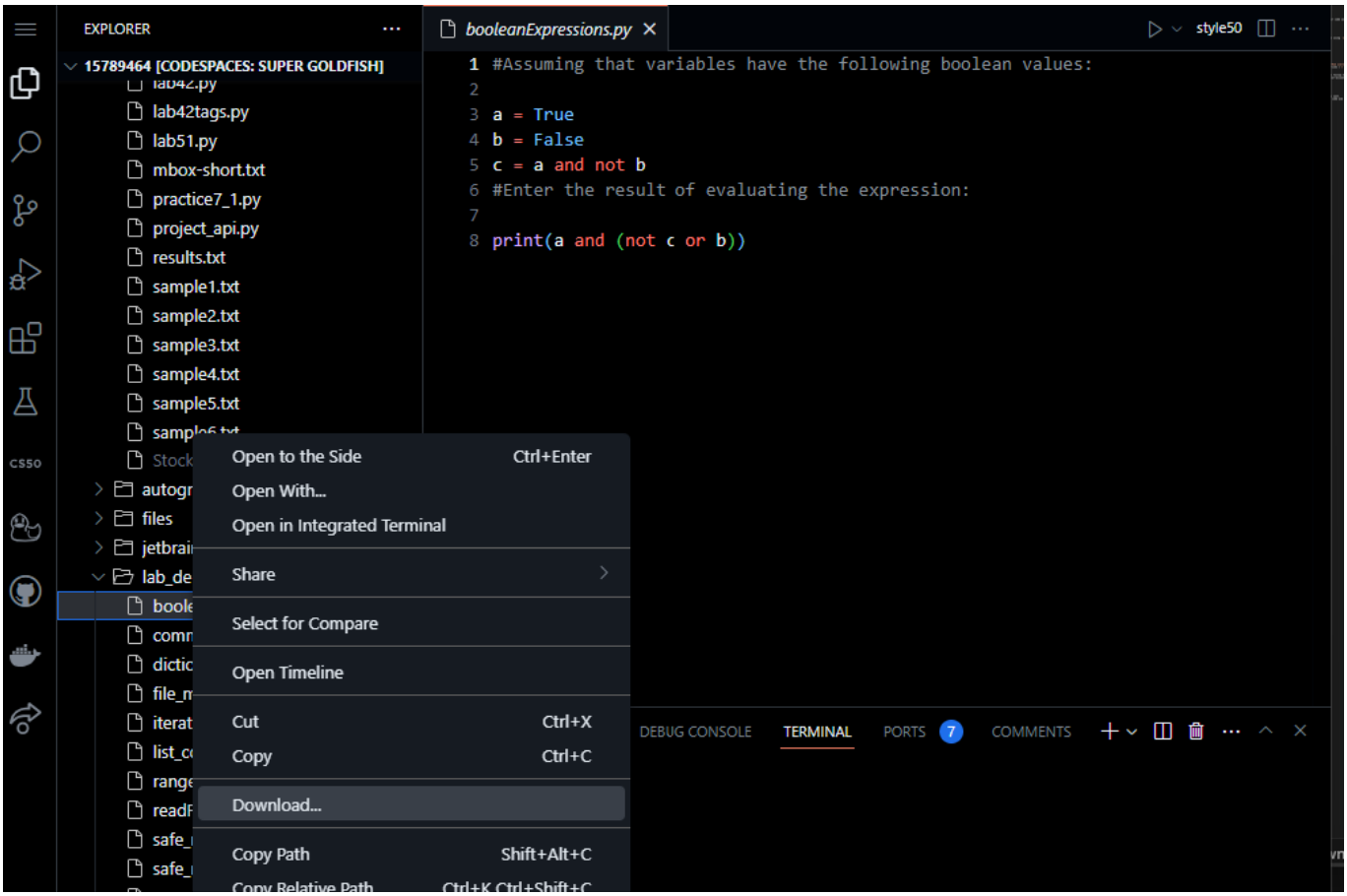
Submit your code

Execute the command below, logging in with your `GitHub username` and `Personal Access Token` when prompted. For security, you'll see asterisks (`*`) instead of the actual characters in your token.

If you do not have generated a Personal Access ToKen follow the instructions: https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token

```
submit50 mkotsovoulou/ods6001a/main/labs/lab5_1
```

You can re-submit your solution as many times as you want. When you are happy with your solution, download the code and upload it to Canvas.



Done!

🎉