

CompSci 101 Lab 7 —File Processing and Tuples

Topics covered:

- Opening files
- Reading data from files
- Tuples
- Closing files
- Writing to files

Programming Exercises

IMPORTANT: for each of your programs you need to add a docstring at the top of the program. The docstring should contain your name, your username, date and a short description of the program.

Notes

1. The skeleton code for each of the Lab 7 programs **and the relevant input files** are provided. Each program requires you to complete a function. As you complete each function you can test that the function is correct by pasting the whole function (including the header) into CodeRunner3 and pressing the CHECK button.
2. After the description of each program there is a textbox with some example code and beside it a textbox containing the expected output for that code.

Question 1 Program 1

Complete the `get_small_middle_large(a_tuple)` function which is passed a tuple of integers as a parameter. The function returns a tuple made up of the smallest, the middle and the largest values of the parameter tuple.

Notes

- If the parameter tuple has less than two elements, the function returns the parameter tuple unchanged.
- If the parameter tuple has exactly two elements, the function returns the tuple with the two elements in sorted order.
- The middle element is the element which is in the middle of the tuple after the elements have been sorted.
- In all other cases, the function returns a tuple made up of three elements: the smallest, the middle, the largest values. The middle element is the element at the index given by the length of the tuple divided by 2 using the `"//"` operator.

Hint: Convert the tuple into a list and sort the list in order to find the middle element.

```
a_tuple = (-3, 6, 9, 4, 5)
print(get_small_middle_large(a_tuple))
print()

print(get_small_middle_large((2,)))
print()

print(get_small_middle_large((2, -5)))
```

```
(-3, 5, 9)

(2,)

(-5, 2)
```

Question 2 Program 2

Complete the `get_two_words(numbers_tuple, words_tuple)` function which is passed two parameters: a tuple of numbers and a tuple of strings. The parameter tuples have the same length and their elements are related in that the element at each index of the second parameter tuple corresponds to the element at the same index in the first parameter tuple, i.e. the element at index 0 in one tuple corresponds to the element at index 0 in the other tuple, etc. The function returns a tuple made up of the **two strings** corresponding to the two smallest numbers in the first parameter tuple in ascending order.

Note: you can assume that the `numbers_tuple` parameter contains at least two elements and the elements are all unique.

```
words = ('carry', 'stick', 'big', 'a', 'and', 'speak', 'softly')
numbers = (23, 40, 19, 22, 42, 41, 63)
print(get_two_words(numbers, words))
print()
```

```
word_list = ('whole', 'kit', 'the', 'and', 'caboodle')
numbers = (76, 81, 62, 83, 87)
print(get_two_words(numbers, words))
```

```
('big', 'a')
```

```
('the', 'whole')
```

Important Note: for Questions 3, 4, 5 and 6, the relevant input files have been provided. This will enable you to test your code using IDLE. Make sure the input files are stored in the same folder as your Python programs.

Question 3 Program 3

Complete the `get_biggest_difference(filename)` function which is passed a string as a parameter: the name of the file to be read. The file contains integers each separated by white space. The function reads the information from the file and returns the absolute value of the difference between the largest number and the smallest number in the file. For example, if `filename` refers to the following file:

```
14      15 18
18 19 11 15
13
```

the value 8 is returned by the function (the difference between 11 and 19).

```
filename = "Lab07Q03_1.txt"
print(get_biggest_difference(filename))
print()

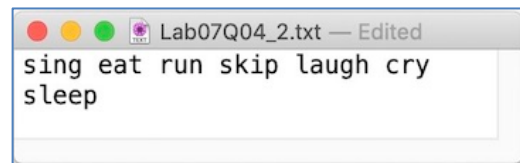
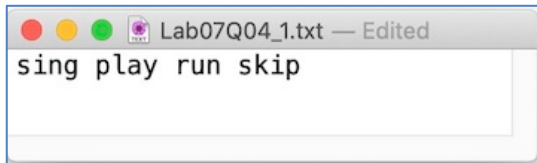
filename = "Lab07Q03_2.txt"
print(get_biggest_difference(filename))
```

```
8
```

```
15
```

Question 4 Program 4

Complete the `get_length_same(filename1, filename2)` function which is passed two string parameters: the names of two text files. The function returns the number of characters in both files which are exactly the same **from the start of the files**. For example if the function is passed the names of the following two files:



the function returns the number 5 because the first 5 characters from the start of both files are exactly the same ("sing ").

```
number_same = get_length_same("Lab07Q04_1.txt", "Lab07Q04_2.txt")
print(number_same)
print()

print(get_length_same("Lab07Q04_3.txt", "Lab07Q04_4.txt"))
```

5

0

Question 5 Program 5

Define the `get_number_of_unique_repeats(filename)` function which is passed the name of a file as a parameter. The file contains names separated by white space. The function reads the information from the file and returns the count of all the names in the file which are repeated at least once. For example, if filename refers to the following file:

```
Penny Charla Charla Palmer
Joya Rochelle Tarra Palmer
Myrtle Conroy Palmer
Gabriel Tarra Tarra Husein
```

Penny

the function returns 4 (the names "Penny", "Charla", "Palmer" and "Tarra" are all repeated at least once).

```
number_repeated = get_number_of_unique_repeats("Lab07Q05_1.txt")
print(number_repeated)
print()

print(get_number_of_unique_repeats("Lab07Q05_2.txt"))
print()

print(get_number_of_unique_repeats("Lab07Q05_3.txt"))
```

4

4

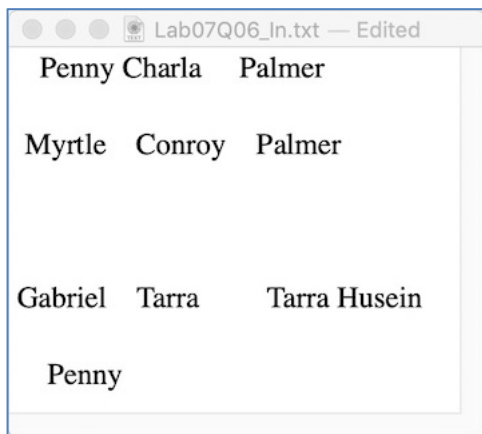
0

Question 6 Program 6

Complete the `write_without_blank_lines(filename_in, filename_out)` function which is passed two strings as parameters. The first parameter is the name of the file to be read and the second parameter is the name of the file to be written. The information read from the first parameter file is written to the second parameter file in the following way:

- every new line of text is numbered starting from the number 1, with a dot and a space (" . ") following the line number,
- any line which is blank (or only contains spaces) is **not** written to the output file.

Example: if the file on the left is the input file, the file on the right is the file written by this function.



Lab07Q06_In.txt — Edited

```
Penny Charla Palmer  
  
Myrtle Conroy Palmer  
  
  
Gabriel Tarra Tarra Husein  
  
Penny
```



Lab07Q06_Out.txt — Edited

```
1. Penny Charla Palmer  
2. Myrtle Conroy Palmer  
3. Gabriel Tarra Tarra Husein  
4. Penny
```

Important Note: for this question you are required to submit ONE line of **text** to CodeRunner, the line of text from the output file which is numbered **100**.