***BE 1600***

***Introduction to***

***Programming and Computation***

***Python***

**Assignment 07**

40 points

**Due 11/27/2023 (11:45 A.M.)**

Assignment Objectives:

* To use text files to store large data sets
* To open a file, read/write data from/to a file
* To access online data sources in CSV format
* To analyze real data.
* To introduce the string data type
* To demonstrate the use of string methods and operators

*Solution for this assignment will not be posted on Canvas; however, the solution of any of the assignment problems can be discussed in the class upon request of a student.*

All assignments must be submitted by the Canvas. **No email or hard copy** is accepted. You must follow the following format:

1. For non-programming questions, use a word file to type your answers. Don’t use the text box on the Canvas to answer the questions or to write comments, we will not read it. State your answer clearly.
2. For programming questions, include only the source file of each programming problem.
3. Submit your files to the Canvas. You must submit your files on time; otherwise, you will receive zero.
4. Use “Add Another File” feature on Canvas to upload each additional file; do not upload the files as a compressed folder.
5. You can upload your files as many times as you like. Only the last attempt counts because the last files you uploaded are the only files your instructor will see.
6. There will be several modules on the Canvas. You need to upload your files using the correct module on the Canvas.
7. Name each file: *Assignment (assignment number)* for the word file [e.g. Assignment 02] and *Assignment (assignment number) \_ (Question number)* for each programming question [e.g. Assignment 02\_Q03].
8. To upload your file(s):

* In Course Navigation, click the ASSIGNMENTS module.
* Click the title of the assignment.
* Click the **Submit** Assignment button.
* Add **File**. ...
* Add Another **File**. ...
* **Submit** Assignment. ...
* View **Submission**.

*It is your responsibility to make sure that each file is uploaded correctly. If you uploaded a wrong file, you receive zero; files will not be accepted after due date even if you have a prove that the file is created before the due date.*

***Make sure you review the Cheating & Plagiarism policy on Canvas.***

Write a program for questions Q.1. to Q.10. Save the files as text files. Submit 10 .txt files to Canvas by the due date.

**Question 01 (3 points)**

Suppose that a text file contains an unspecified number of scores; scores are written on one line separated by blanks. Write a ***main function*** that reads the scores from the file and stores them in a list; the program displays the list of scores, number of scores, total of scores, and average of scores. Your program should prompt the user to enter a filename.

Here is a sample run:

|  |  |
| --- | --- |
| **Enter a filename: testFile.txt**  **[45, 22, 77, 12, 6, 80, 44, 15, 71]**  **There are 9 scores**  **The total is 372**  **The average is 41.33** |  |

**Question 02 (5 points)**

Write a ***function*** that creates a data file with 1000 lines. Each line in the file consists of a faculty first name, last name, rank, and salary. Faculty’s first name and last name for the *i*th line are FirstName*i* and LastName*i*. The rank is randomly generated as assistant, associate, and full. The salary is randomly generated as a number with two digits after the decimal point. The salary for assistant professor should be in the range from 50,000 to 80,000, for associate professor from 60,000 to 110,000, and for full professor from 75,000 to 130,000. Save the file in Salary.txt.

Write another ***function*** that takes an input filename (Salary.txt). The function read the data from Salary.txt file and displays the total salary for assistant professors, associate professors, full professors, and all faculty, respectively, and the average salary for assistant professors, associate professors, full professors, and all faculty, respectively.

Write a ***main function*** that calls the above two functions.

Here is a sample run:

Output to Salary.txt file

FirstName1 LastName1 assistant 60055.95

FirstName2 LastName2 associate 81112.45

. . .

FirstName1000 LastName1000 full 92255.21

Output to shell

**Total salary for assistant professors: 21322373.00**

**Total salary for associate professors: 28449384.00**

**Total salary for full professors: 35013898.00**

**Average salary for assistant professors: 65406.05**

**Average salary for associate professors:84419.54**

**Average salary for full professors: 103898.81**

**Question 03 (5 points)**

Consider the following two files included with this assignment:

* GirlNames.txt—This file contains a list of the 200 most popular names given to girls born in the United States from the year 2000 through 2009.
* BoyNames.txt—This file contains a list of the 200 most popular names given to boys born in the United States from the year 2000 through 2009.

Write a ***main function*** that reads the contents of the two files into two separate lists. The function allows the user to enter a boy’s name, a girl’s name, or both, and displays a message indicating whether the names were among the most popular.

Here are sample runs:

**Enter a boy's name, or N if you do not wish to enter a boy's name N**

**Enter a girl's name, or N if you do not wish to enter a girl's name Ana**

**You chose not to enter a boy's name**

**Ana is one of the most popular girl's names**

**Enter a boy's name, or N if you do not wish to enter a boy's name Sam**

**Enter a girl's name, or N if you do not wish to enter a girl's name N**

**Sam is not one of the most popular boy's names**

**You chose not to enter a girl's name**

**Enter a boy's name, or N if you do not wish to enter a boy's name Jacob**

**Enter a girl's name, or N if you do not wish to enter a girl's name Grace**

**Jacob is one of the most popular boy's names**

**Grace is one of the most popular girl's names**

**Question 04 (5 points)**

Products.csv contains the below data.

*product, color, price*

*suit, black, 250*

*suit, gray, 275*

*shoes, brown, 75*

*shoes, blue, 68*

*shoes, tan, 65*

Write a ***function*** that creates a list of lists from the file; each list (row) includes a product (one line of data). For example, the first data line would be:

['suit', 'black', '250']

Print the list of lists in tabular format, total suits price and total shoes price.

Use “products.csv” included with this assignment

Here is a sample run:

**product color price**

**suit black 250**

**shoes brown 75**

**suit gray 275**

**shoes blue 68**

**shoes tan 65**

**Total suits price 525**

**Total shoes price 208**

**Question 05 (3 points)**

Write a ***main function*** that removes all the occurrences of a specified string from a text file. Your program should prompt the user to enter a filename and a string to be removed.

Here is a sample run:

**Enter a filename: testFile.txt**

**Enter a string to be removed: to**

|  |  |
| --- | --- |
|  |  |

**Question 06 (3 points)**

Write a ***function*** that takes the length of a string and returns a string of random letters. Write a main ***function*** that prompts the user to enter the length of a string and prints a random string.

Here are sample runs:

**Enter length of a string 4**

**The random string is: zofa**

**Enter length of a string 13**

**The random string is: ddyeqadkfsiir**

**Question 07 (5 points)**

Write a ***function*** that accepts a string as an argument and returns the number of vowels that the string contains. The application should have another ***function*** that accepts a string as an argument and returns the number of consonants that the string contains.

Write a main ***function*** that prompts the user for a string and displays the number of vowels and the number of consonants it contains.

Here is a sample run:

**Enter a string: To Be Or Not To Be**

**The string you entered includes 6 vowels and 7 consonants**

**Question 08 (3 points)**

Some Web sites impose certain rules for passwords. Write a **function** that takes a string and checks whether a string is a valid password. Suppose the password rules are as follows:

■ A password must have at least eight characters.

■ A password must consist of only letters and digits.

■ A password must contain at least two digits.

Write a main **function** that prompts the user to enter a password and displays if the password is valid or not.

Here are some sample runs:

**Enter a string for password: JohnWayne1980**

**Valid password**

**Enter a string for password: John1980Wayne**

**Valid password**

**Enter a string for password: JohnWayne9**

**Invalid password**

**Question 09 (3 points)**

Write a ***function*** that checks whether two words are anagrams.

Two words are anagrams if they contain the same letters. For example, silent and listenare anagrams. The header of the function is:

defisAnagram(s1, s2):

Write a main ***function*** that prompts the user to enter two strings and, if they are anagrams, displays is an anagram; otherwise, it displays is not an anagram. Do not use method sort or lists.

Here are sample runs:

**Enter the first string: silent**

**Enter the second string: listen**

**silent and listen are anagram.**

**Enter the first string: spam**

**Enter the second string: slam**

**spam and slam are not anagram.**

**Question 10 (5 points)**

Write a ***function*** that takes an input filename and an output filename; the function reads the text from the input file, encodes the text using scramble2Encrypt function discussed in chapter 8, and writes the encrypted text in the output file.

Suppose a file is encrypted using the scramble2Encrypt function. Write a ***function*** that takes an input filename and an output filename; the function reads the text from the input file, decodes the text using scramble2Decrypt function discussed in chapter 8, and writes the decrypted text in the output file.

Your program should have a ***main function*** that prompts the user to enter an input filename, an output filename, and whether the user wants to encrypt or decrypt the input file; based on user choice, the main function calls the appropriate function.

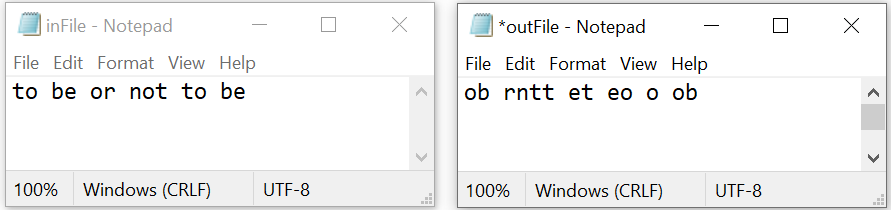
Here are some runs:

**Enter a source filename: inFile.txt**

**Enter a target filename: outFile.txt**

**Enter E to encrypt or D to decrypt the input file: E**

**Done**

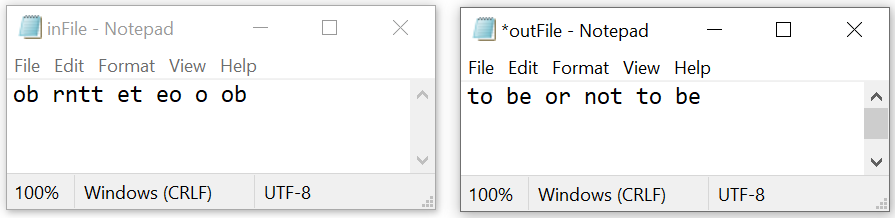
****

**Enter a source filename: inFile.txt**

**Enter a target filename: outFile.txt**

**Enter E to encrypt or D to decrypt the input file: D**

**Done**

****