**Lab 7: File reading / writing**

Learning Outcomes:

* Get familiar with reading and writing text files
* Get to apply file handling in problems

Instructions:

* Suggest that you create a working folder **is111\lab7** in your **C** or **D** drive. Store all your solutions in this working folder.
* Challenging questions are marked with \*.

To submit:

* Please submit your working solutions via your assignment Dropbox in eLearn **within 1 week**. The Dropbox will be closed after the due date.
* Zip up all your source files into a single zip file called **<your email ID>\_lab7.zip** (e.g. **ahlian.lim.2011\_lab7.zip**). You should only submit a single zip file for each lab.

1. [\*] In a file named **lab7\_1.py**, write a program to read from the file called words.txt. Every line in words.txt contains one single word. The program is supposed to read words in the file and print all words along with the count of distinct vowels present in the word as shown in the sample run. You may want to write additional functions in the file, one function perhaps to count the number of distinct vowels in every word etc.

Here is the output when **lab7\_1** is run (with the given test words.txt file):

|  |
| --- |
| D:\is111\lab7>**python lab7\_1.py**  python 1  dynamic-typed 3  function 3  parameter 2  argument 3  str 0  int 1  float 2  … *rest not shown for brevity* |

To submit**: lab7\_1.py**

1. [\*\*] In a file named **lab7\_2.py**, write a program that reads inputs from a file called phone.txt. Every line in phone.txt contains one phone number. The program is supposed to verify if lines in the text file contain valid phone numbers and write out only valid phone numbers one number in a line into a file named valid.txt

A valid phone number has the format

Nddd-dddd where N is 6 or 8 or 9 and d is any digit 0-9

Here is the output when **lab7\_2** is run (with the given phone.txt file):

|  |
| --- |
| D:\is111\lab7>**python lab7\_2.py**  Invalid phone numbers:  7346-8895  123445384  846-7894  6457-q088  Writing valid phone numbers to out.txt ...  Done. |

To submit**: lab7\_2.py**

1. [\*] In a file named **lab7\_3.py**, write a program that reads inputs from a file called sales.txt. Every line in sales.txt contains sales amount of a particular month in the format.

Month name, Sales amount

Your program is supposed to print the total sales amount of all months in the file on the screen.

Here is the output when **lab7\_3** is run (with the given phone.txt file):

|  |
| --- |
| D:\is111\lab7>**python lab7\_3.py**  Total sales amount for the whole year $64481.90 |

To submit**: lab7\_3.py**

1. [\*\*\*] There is a belief that humans are able to read even when the order of the letters are misplaced. Try reading this:

I cnduo't bvleiee taht I culod uesdtannrd waht I was rdnaieg

Interesting read here

<https://www.mrc-cbu.cam.ac.uk/people/matt.davis/cmabridge/>

In a file named **lab7\_4.py**, write a program that reads text from a file called talk.txt and prints the content on the output screen scrambling the words in the content as described below:

* Scramble every word in the text leaving untouched the first and last letter
* You may assume that the text has words separated by a single space. Maintain single space between words in the scrambled version
* Challenge yourself to omit the punctuations [, ' ! .] in the content untouched in the original place

(Adapted from the book “The practice of Computing using Python” - Programming Project on Page 340)

As an example, if the text reads “Programming is really fun”

One version after scrambling could be “Pgmnarirmog is rlelay fun”

Note that since we scramble only the middle letters, “is” and “fun” in the example are not scrambled.

As you think of designing solution for this question, define and make use of functions to do simpler tasks. Test the function before building on to complete the solution.

**Note**: To scramble middle letters of a 6-lettered word, you could take each letter in the indices 1-4 and put them in positions randomly between 1 and 4. A 6-letter word after scrambling should contain all the six letters.

To make this exercise simpler, if after scrambling one word if output happens to remain the same as the original, you may ignore.

You may want to use random module to help you with scrambling letters.

A sample run of **lab7\_4** could be (with the given talk.txt file):

|  |
| --- |
| D:\is111\lab7>**python lab7\_4.py**  … *partial output shown for brevity*  We do not need miagc to chgnae the wldor, we crray all the pwoer we need inisde oleeursvs aedyral: we have the pweor to imngiae betert. |

To submit**: lab7\_4.py**

**\*Use shuffle**

**Import random**

**Random.shuffle(obj)**