Module 7 Assignment

Processing Text Data

# Instructions

Create a jupyter notebook called ‘module7\_assignment’ and use it to answer the homework problems below.

## Problem 1

Create a function called calc\_file\_length. This function will accept one argument which will be a file path that points to a text file. The function will first check if the file exists. If the file does not exist, the function will return False. Otherwise, if the file does exist, the function will open the file and count the number of lines in the file. The function will return the number of lines. Please be sure to use variable names that make sense. For example, when you open the file, the variable name you use for the file object should not be ‘filepath’. That is because it is *not* a file path, it is a file. So, call it something like ‘my\_file’. Here is an examples of how the function should work:

### Example 1

|  |
| --- |
| my\_filepath = os.path.join('Users', 'william', 'file.txt' ) file\_length = calc\_file\_length(my\_filepath) print(file\_length) # The number of lines in the file is the expected result. For example, if the file is 101 lines long then 101 should be the value of file\_lenght |

## Problem 2

Define a function called parse\_weather\_data\_file. This function will accept one argument which will be a file path that points to a text file. Assume the file has lines of weather data, where the first 8 characters are a weather station identifier. The next three characters are temperature in celsius. The next two characters after that are the relative humidity. For example, a line of data might be:

|  |
| --- |
| 12345678-0176 |

In this line, the station id is 12345678, the temperature is -01, and the relative humidity is 76.

The function parse\_weather\_data\_file should open the file and then split each line into the three fields, as it reads through the file line. It should create a dictionary for each line, as it reads through the lines. The dictionary should have the keys ‘station\_id’, ‘temperature’, and ‘relative\_humidiy’. The values should be filled in from the line. Each dictionary should be added to a list. The function should then return this list. For example, let’s say the files has these three lines:

|  |
| --- |
| 12345678-0176 14342378+0321 92745371+0753 |

The function should return the following list of dictionaries:

|  |
| --- |
| [{'station\_id': 12345678, 'temperature': -1, 'relative\_humidity': 76}, {'station\_id': 14342378, 'temperature': 3, 'relative\_humidity': 21}, {'station\_id': 92745371, 'temperature': 7, 'relative\_humidity': 53}] |

## 

## Problem 3

Write a function called word\_count. This function will accept one argument, which will be a file path to a file. The function will read this file and return a dictionary that has the counts of each word in the file. For example, if the contents of the file are these lines:

|  |
| --- |
| I like apples, bananas, and pears. I also like pumpkins, but I like apples the most. |

The function would return this dictionary:

|  |
| --- |
| {'I': 3, 'like': 3, 'apple': 2, 'bananas': 1, 'and': 1, 'pears': 1, 'also': 1, 'pumpkins': 1, 'but': 1, 'the': 1, 'most': 1} |

You will need to use what we learned in lecture to split each into multiple words. I also show you how to remove punctuation in one of the lectures.

## Problem 4

Write a function called write\_to\_file. It will accept two arguments. The first argument will be a file path to the location of a file that you want to create. The second will be a list of text lines that you want written to the new file. The function should create the file and then write the lines of text to the file. The function should write each line of text on its own line in the file; assume the lines of text do not have carriage returns. Do not use the csv module to complete this assignment.

## Problem 5

Write a function called copy\_file. This function will accept two arguments. The first argument will be a file path that points to a text file that exists. The second argument will be a file path that points to a location where you’d like to copy the first file. The function will open both files (creating the second), it will then read in the lines of the first file and then write them to the second file. It will do this reading and writing process line by line; it should not read in all of the first file, before writing the contents to the second file.

# How to Turn in the Assignment

First, remove any testing/ scripting code so that only the functions definitions remain in your notebook. If you want to make a copy of the notebook to keep a copy of any scripting or testing code that is fine. But, otherwise - ONLY turn in the functions definitions.

Then, please download the notebooks as a Python file (Go to the file menu, in Jupyter notebooks, and choose "Download as…", then choose python to download as a python file) and submit the assignment.