Investigating Texts and Calls

Project Overview

In this project, you will complete five tasks based on a fabricated set of calls and texts exchanged during September 2016. You will use Python to analyze and answer questions about the texts and calls contained in the dataset. Lastly, you will perform run time analysis of your solution and determine its efficiency.

What will I learn?

In this project, you will:

- Apply your Python knowledge to breakdown problems into their inputs and outputs.
- Perform an efficiency analysis of your solution.
- Warm up your Python skills for the course.

Why this Project?

You'll apply the skills you've learned so far in a more realistic scenario. The five tasks are structured to give you experience with a variety of programming problems. You will receive code review of your work; this personal feedback will help you to improve your solutions.

Step 1 - Download the Files

Download and open the zipped folder here. In the folder you will find five python files `Task0.py`, `Task1.py`, ...,`Task4.py` and two csv files `calls.csv` and `texts.csv`

About the data

The text and call data are provided in csv files.

The text data (`text.csv`) has the following columns: sending telephone number (string), receiving telephone number (string), timestamp of text message (string).

The call data (`call.csv`) has the following columns: calling telephone number (string), receiving telephone number (string), start timestamp of telephone call (string), duration of telephone call in seconds (string)

All telephone numbers are 10 or 11 numerical digits long. Each telephone number starts with a code indicating the location and/or type of the telephone number. There are three different kinds of telephone numbers, each with a different format:

- Fixed lines start with an area code enclosed in brackets. The area codes vary in length but always begin with 0. Example: "(**022**)40840621".
- Mobile numbers have no parentheses, but have a space in the middle of the number to help readability. The mobile code of a mobile number is its first four digits and they always start with 7, 8 or 9. Example: "**9341**2 66159".
- Telemarketers' numbers have no parentheses or space, but start with the code 140. Example: "**140**2316533".

Step 2 - Implement the Code

Complete the five tasks (`Task0.py`, `Task1.py`, ...,`Task4.py`). Do not change the data or instructions, simply add your code below what has been provided. Include all the code that you need for each task in that file.

In Tasks 3 and 4, you can use in-built methods `sorted()` or `list.sort()` for sorting which are the implementation of Timsort and Samplesort, respectively. Both these sorting methods have a worst-case time-complexity of *O(n log n).* Check the below links to learn more about these methods:

- How to use the above methods - https://docs.python.org/3/howto/sorting.html
- Complexity analysis of Timsort and Samplesort - http://svn.python.org/projects/python/trunk/Objects/listsort.txt

The solution outputs for each file should be the print statements described in the instructions. Feel free to use other print statements during the development process, but remember to remove them for submission - the submitted files should print only the solution outputs.

Step 3 - Calculate Big O

Once you have completed your solution for each problem, perform a run time analysis (Worst Case Big-O Notation) of your solution. Document this for each problem and include this in your submission.

Step 4 - Check again Rubric and Submit

Use the rubric to check your work before submission. A Udacity Reviewer will give feedback on your work based on this rubric and will leave helpful comments on your code