SI 601 Fall 2015 Homework 1 (100 points)

Due Date: Tuesday, Jan 20, 2015. 5:00pm

The purpose of this assignment is to (a) to help you become more familiar with Python's basic programming constructs, data-handling methods and libraries, and (b) to get you started doing some basic manipulation of real data.

This assignment uses a summary dataset from the World Bank, containing 18 'indicators' (statistics) from over 200 countries, collected for each year in the period 2000-2010. (This is a subset of a larger dataset with 331 indicators spanning 1960-present: more information about each indicator and those not included here can be found at: http://data.worldbank.org/indicator)

The main dataset is stored in "tab-delimited format" a standard exchange format for simple tabular data. Tab-delimited format store one table row per line, with the first row being a header row with the names of the columns; the other rows consist of a list of tab-separated values, corresponding to the indicators named in the columns of the header row.

This homework involves multiple steps: you will get credit for each step you complete successfully. You should complete the steps one by one and verify that you've completed each correctly before moving on to the next one.

Getting started:

Download the zip file si601_w15_hw1.zip and unzip it. You'll see the following files:

world_bank_indicators.txt
world_bank_regions.txt
country_data_step1_desired_output.csv
country_data_step2_desired_output.csv
urban_population_ratio_vs_birth_rate.pdf

The first two files are in tab-separated format and they are your input files. For each file, the first row is a header row that contains the column names, which should be read, but is not actually part of the data to be used/analyzed.

The world_bank_indicators.txt file has country data, formatting into 20 columns per line, which are:

Column Number	*Column Name*
0	Country Name
1	Date
2	Transit: Railways, (million passenger-km)
3	Transit: Passenger cars (per 1,000 people)
4	Business: Mobile phone subscribers
5	Business: Internet users (per 100 people)
6	Health: Mortality, under-5 (per 1,000 live births)
7	Health: Health expenditure per capita (current US$)
8	Health: Health expenditure, total (% GDP)
9	Population: Total (count)

10	Population: Urban (count)
11	Population:: Birth rate, crude (per 1,000)
12	Health: Life expectancy at birth, female (years)
13	Health: Life expectancy at birth, male (years)
14	Health: Life expectancy at birth, total (years)
15	Population: Ages 0-14 (% of total)
16	Population: Ages 15-64 (% of total)
17	Population: Ages 65+ (% of total)
18	Finance: GDP (current US$)
19	Finance: GDP per capita (current US$)

The world_bank_regions.txt file has 3 columns per line:

Column Number	*Column Name*
0	Region
1	Subregion
2	Country Name

Your code will load these two files and do some manipulation on them to get the desired outputs.

You should review all steps before starting, to think about what kinds of data structures you're going to need that can be used for both the earlier and later steps.

Step 1 (50 points) Finding out correlations between urban population ratio and birth rate

In this step, you need to write Python code that reads the input file world_bank_indicators.txt, and generates an output file that looks the same as country_data_step1_desired_output.csv

To do this, first read the file world_bank_indicators.txt, parsing each line in tab-delimited format into an internal data structure of your choice, where the individual values for each column can be used by later code.

HINT: For platform independence, using 'rU' to turn on Python universal newline support in the open function. For example:

```
input_file = open('inputfile.txt', 'rU')
```

Read about file methods at http://docs.python.org/2/library/stdtypes.html#file-objects

As you'll notice when processing the file, some rows in the world_bank_indicators.txt data file have missing values for the columns you're extracting (step 1). How to deal with missing data is an important and recurring issue with real-world datasets. One strategy is just to ignore those rows, and that's what you should do for this homework. We'll talk about that and other ways of dealing with missing data in an upcoming lecture. (It's a research topic all on its own..)

We are only interested in the following columns in world_bank_indicators.txt:

- Country Name

- Total population
- Urban population
- Birth rate. The birth rate data in world_bank_indicators.txt is the number of births per 1000 people, divide it by 1000 to convert it to birth rate per person.

But then, we want the data from all rows.

(20 points) Store the total population, urban population, and birth rate from all years for each country in a suitable data structure.

(15 points) For each country, you should calculate the average total population based on the data from all years in the file, and similarly, calculate average urban population and average birth rate. Store the result in appropriate data structure and then calculate

- Average urban population ratio (divide average urban population by average total population)
- −log(average birth rate per person)

Here, log(x) is the natural logarithm function: you'll need to use the log function in the math package. Check out http://docs.python.org/2/library/math.html. We're using log(x) here as a commonly-used data analysis tool that "stretches" out the scale for smaller values along each of these axes, which will help us see relationships between these variables more clearly later.

(10 points) Once you have all data you need, output it to a file in "comma-separated values" (CSV) format. Note that the provided country_data_step1_desired_output.csv is sorted by the "average urban population ratio" column in decreasing order, and in case of ties, by the "average birth rate" column in decreasing order. Your step 1 output file should be sorted in the same way.

CSV format is a standard data exchange format for simple tabular data, typically used by spreadsheet programs like Excel. CSV format uses one row per line, with the first row being a header row with the names of the columns; the other rows consist of a list of comma-separated values (hence the name), corresponding to the indicators named in the columns of the header row.

You don't need to write all this code from scratch: you should use the existing csv package for this task as described in lecture slides. See http://docs.python.org/2/library/csv.html

Your step 1 CSV output file should be named as si601_hw1_step1_ ***youruniquename***.csv

(5 points) Load you step 1 output CSV file into either Excel or Google Docs Spreadsheet, and generate a scatter plot of the relationship between urban population ratio and average birth rate. Your plot should look like the sample given in urban_population_ratio_vs_birth_rate.pdf.

Save your plot file as si601_hw1_plot_***youruniquename***.pdf

Step 2 (25 points) Adding region data

(10 points) Add code to read the region file (world_bank_regions.txt) into an appropriate data structure.

(10 points) Use the region data structure you created to look up, for each country, a country's corresponding region, and add the region data to the data structure you created for generating step1 output. If you do not find some country in the region data file, use 'No region' as the region for that country.

(5 points) Save your step 2 results in a CSV file names as as si601_hw1_step2_ ***youruniquename.***csv. It should contain the same data as the provided country_data_step2_desired_output.csv file.

Step 3 (25 points) Finding the country with the highest population in each region

The goal of step 3 is to generate a CSV file that looks the same as the provided region_data_desired_output.csv file. There are multiple ways to do it. But to get the full credit, you should use the itertools package (https://docs.python.org/2/library/itertools.html) and in particular, use the itertools.groupby() function.

Right, I did not talk about itertools in class. The point here is to encourage you to read the documentation of an unfamiliar package and figure out how to use it. You do not have to use itertools, but if you do not, you get 15 points instead of 25.

Your step 3 CSV output file should be named as si601_hw1_step3_ ***youruniquename.***csv

Submission instructions

The name of your python file should be si601_hw1***_youruniquename***.py. When you run it, it should write out three output files named si601_hw1_step1_ ***youruniquename.***csv, si601_hw1_step2_ ***youruniquename.***csv, and si601_hw1_step3_ ***youruniquename.***csv.

What to submit:

A zip file, **named as above**, that contains: your python source code file, your output CSV files, and your plot file.