TASK

In this assignment, you will perform some basic data analysis on a dataset obtained from the [Gapminder](http://www.gapminder.org/) website which collects and presents authentic statistics of all countries worldwide.

Download [this zip package](https://doms.csu.edu.au/csu/file/8ecc7393-0664-44fc-8288-8a5a29de687b/1/ITC558_202030_A3_dataset.zip) which contains three dataset files: ‘life.csv’, ‘bmi\_men.csv’ and ‘bmi\_women.csv’. First file contains data about average life expectancy (in years) for most countries worldwide. Other two files contain data about men and women average Body Mass Index (BMI) for the same set of countries. These are plain text files with all data separated by commas. You can also open the files in a spreadsheet application to better understand their contents. All three files have a similar structure — first row contains the year headers and first column contains the country names. There is data about 186 countries for a period of 1980 to 2008.

Your program should perform the following steps.

(1) Read all the data from files and save into a 2D list and two dictionaries.

The life expectancy data should be stored in the form of two dimensional list where the outer list has 186 elements. Each inner list contains data for specific countries.

The BMI data from both files should be stored in two dictionaries which map country names to a list of data values. Both dictionaries will contain 186 keys, with each key associated with a list of 29 values (BMI data from 1980 to 2008).

Following diagram illustrates the required data structures. Note that all numbers have been converted from string to float data types.

You should use these collections for the next five steps — do not read the files again.

(2) Some users may be interested in gender neutral BMI data. For this purpose, create another Python dictionary bmi\_all of the same structure and size as bmi\_men (or bmi\_women) and populate it with worldwide gender-average BMI values. For example bmi\_all for Zimbabwe in 2008 would be 23.3.

(3) Use the bmi\_all dictionary from step 2 to calculate worldwide statistics (min, max and [median](https://en.wikipedia.org/wiki/Median#Finite_data_set_of_numbers)) for a user-selected year. See example in the sample-run below. Median value should be displayed with a precision of 3 decimal places.

(4) Compare the latest 5-year BMI data for men against women for the three most populous countries in the world (China, India, United States). First work out the 2004 to 2008 men’s BMI average for these countries. Repeat the same for women’s BMI. Then display the men and women BMI values and the [percentage difference](https://www.mathsisfun.com/percentage-difference.html) between the two. Display all values with 2 decimal places precision.

(5) Plot life expectancy trend of a user selected country. Your program will prompt the user for a country name (case insensitive) and then create a line chart showing life expectancy variation over the years. Sample run below shows an example.

(6) To explore the correlation between BMI and life expectancy, plot worldwide average values of the two on the same chart. For this purpose, your program will create two lists of 29 elements each to store worldwide average BMI and life expectancy data for each year. Refer to sample run for an example.

[Disclaimer: [Correlation does not imply causation](https://www.tylervigen.com/spurious-correlations).]

For plotting charts in step 5 and 6, use the matplotlib library. Consult the textbook section 7-8 to learn how to draw simple charts. The chart for step 6 is rather complex because it contains two y-axis. For this part, please review and adapt the sample code below.

|  |
| --- |
| import matplotlib.pyplot as plt  x\_data = [20, 21, 22, 23, 24]  y1\_data = [1, 3, 5, 8, 10]  y2\_data = [100, 150, 190, 180, 115]  fig = plt.figure()  ax1 = fig.add\_subplot()  ax1.set\_xlabel('X data')  ax1.plot(x\_data, y1\_data,'b\*-')  ax1.tick\_params(axis='y', labelcolor='b')  ax1.set\_ylabel('Y1 data', color='b')  ax2 = ax1.twinx() # create a second axes that shares the same x-axis  ax2.plot(x\_data, y2\_data, 'ro-')  ax2.tick\_params(axis='y', labelcolor='r')  ax2.set\_ylabel('Y2 data', color='r')  plt.show() |

**Important Note:**Other than matplotlib, you can NOT use any library module or third party module in this assessment.

Your program should be able handle following invalid inputs or error situations.

* Any of the three dataset files do not exist or can’t be read.
* Non-numeric or out of range year value provided by user.
* Incorrect country name provided by user.

A sample run of the program is given below to clearly demonstrate all the requirements.

|  |
| --- |
| A simple data analysis program  --- Step 1 ---  All dataset has been read into memory.  --- Step 2 ---  Gender-average BMI data stored in a new dictionary.  --- Step 3 ---  Select a year to find statistics (1980 to 2008): **garbage**  <error> That is an invalid year.  Select a year to find statistics (1980 to 2008): **1990**  In 1990, countries with minimum and maximum BMI values were 'Vietnam' and 'Tonga' respectively.  Median BMI value in 1990 was 24.450  --- Step 4 ---  Men vs women BMI in highest population countries:  \*\*\* China \*\*\*  Men: 22.82  Women: 22.86  Percent difference: 0.18%  \*\*\* India \*\*\*  Men: 20.92  Women: 21.22  Percent difference: 1.42%  \*\*\* United States \*\*\*  Men: 28.30  Women: 28.18  Percent difference: 0.42%  --- Step 5 ---  Enter the country to visualize life expectancy data: **jupiter**  <error> 'jupiter' is not a valid country.  Enter the country to visualize life expectancy data: **sRi laNka**  Plot for 'Sri Lanka' opens in a new window.  --- Step 6 ---  Correlation plot opens in a new window.  Goodbye. |

Your assignment should consist of following tasks.

**Task 1**

Draw a flowchart that represent the algorithms of **step 2** and **step 6**. Include flowcharts of any functions that are called during these steps. You can draw the flowcharts with a pen/pencil on a piece of paper and scan it for submission, as long as the handwriting is clear and legible. However, it is strongly recommended to draw flowcharts using a drawing software.

**Task 2**

Select six sets of test data that will demonstrate the 'normal' operation of your program; that is, test data that will demonstrate what happens when a VALID input is entered. Select four sets of test data that will demonstrate the 'abnormal' operation of your program.

Set out the test cases in a tabular form as follows. It is important that the output listings (i.e., screenshots) are not edited in any way.

|  | | | | |
| --- | --- | --- | --- | --- |
| **Test Data Table** | | | | |
| **Test data type** | **Test data** | **The reason it was selected** | **The output expected due to the use of the test data** | **The screenshot of actual output when the test data is used** |
| Normal |  |  |  |  |
| Normal |  |  |  |  |
| Abnormal |  |  |  |  |
| Abnormal |  |  |  |  |

**Task 3**

Implement your algorithm in Python. Comment on your code as necessary to explain it clearly.  Run your program using the test data you have selected and complete the final column of test data table above.

**Your submission will consist of:**

1. Your algorithm through flowchart/s
2. The table recording your chosen test data and results (it should be a PDF file)
3. Source code for your Python implementation

Thus your directory for Assignment will at least contain two or three files (depending on whether you put the flowchart and the test table in the same file). Next, these files should be compressed into a single ZIP before uploading in TURNITIN.  
  
It is critically important that your test runs are unmodified outputs from your program, and that these results should be reproducible by the marker running your saved .py python program.