Programming Project 5

Assignment Overview
This assignment will give you experience with lists, file I/O, and exceptions as well as more experience with functions. This assignment is worth 40 points (4.0% of the course grade) and must be **completed and turned in before 11:59 on October 15, 2012.**

Background
The Centers for Disease Control (CDC) keeps data on what it calls "Winnable Battle Risk Factors and Health Indicators". These are negative behaviors or incidents that could possibly be avoided by changes in lifestyle. In this project you will examine some CDC data to discover the states with the best and worst records in regards to a few of these risk factors and health indicators.

Project Specifications
The file "riskfactors.csv" lists data on 20 different risk factors and health indicators for each state. The data is in "comma separated value" format (csv), which means that each entry is separated from the others by a comma. Examining the file in a text editor or a spreadsheet program or both should help you understand the format. We grabbed this file from the CDC web page and have placed a copy in the project directory for you.

Since analyzing 20 different indicators can be a bit confusing, we will only look at five: *Heart Disease Death Rate*, *Motor Vehicle Death Rate*, *Teen Birth Rate*, *Adult Smoking*, and *Adult Obesity*. Your program will read in the data from the csv file and find the states with the best and worst record for each of these indicators (largest and smallest values). It will produce a file called "best_and_worst.txt" which lists the states that have the highest and lowest value for each of the indicators, along with their values.

Your program must be general enough to work with a similar file with states as rows and with the same column headers. That is, if the CDC puts out a new file with different values in the cells (e.g. maybe new research changed some values), your program will work correctly.

Additional Requirements

1. You must create and use at least 2 meaningful functions (your choice).

2. Prompt for the input file. Your program must check if the file exists. If it does not, your program should output a "file not found" message and keep asking until a correct file is entered. (You do not need to prompt for the output file name.)

3. Your program must format the file into columns (see below). You don't have to match our formatting exactly, but columns should line up and it should be readable. Use string formatting.

Output of the file `best_and_worst.txt`:

```
Indicator                      : Min                                Max
-------------------------------------------------------------------------------------
Heart Disease Death Rate (2007): Minnesota             129.8     Mississippi      266.5
Motor Vehicle Death Rate (2009): District of Columbia    4.8     Wyoming           24.6
Teen Birth Rate (2009)         : New Hampshire          16.4     Mississippi       64.2
Adult Smoking (2010)           : Utah                    9.1     West Virginia     26.8
Adult Obesity (2010)           : Colorado               21.4     Mississippi       34.5
```

Deliverables

Turn in proj05.py using handin – this is your source code solution; be sure to include your section, the date, the project number and comments describing your code. Please be sure to use the specified file names, and save a copy of your proj05.py file to your H drive as a backup.

Assignment Notes

1. Not all lines of the file contain state data. In particular, the first few lines are irrelevant.
2. Don't forget to convert strings to numbers where appropriate.
3. Watch out for the fact that some data has a percent sign.
4. To open a file for output, remember:
    a. Open the file with the `'w'` mode string.
    b. You can only write strings to a file, so you must convert each output to a string before you write them.
    c. Also, remember that if you want a separate line to occur in your output file, you must specifically output the carriage return/line feed string `"\n"`.
5. Don't forget to close your file—otherwise the string might not get written. If you find that your file has nothing in it or is missing information, forgetting to close the file is a likely reason.
6. Depending on how you design your program you may find some of the following list functions and methods useful: `sort, sorted, min, and max`. I used some, but not all of those.
7. There exists a module for reading csv files. This csv file is formatted nicely so that using the csv module is not needed—it is an unnecessary complication. However, you are free to use it, if you wish—it is a wonderfully powerful module that is necessary for dealing with most csv files.