Part 2 - Write a program using Python

To practice using python, and to help you investigate Python's features, you will implement a program that reads in some data and allows the user to search and view the data in different ways.

The Social Security Administration annually publishes a list of the most popular baby names on their web site: http://www.ssa.gov/OACT/babynames. The data can be reformatted into a text file with the following format. Each line consists of the data for a name over 110 years. The line begins with the name followed by 12 numbers (separated by spaces). The numbers indicate the rank (popularity) of the name in 1900, 1910, 1920, ..., 2010. A rank of one (1) is the most popular baby name, and a rank of 999 is not very popular at all. A zero means that it did not appear in the list at all that year.

Here is an example segment of the file:

```
....
Adrianne 0 0 0 0 0 0 939 648 575 0 0 0
Adriel 0 0 0 0 0 0 0 0 0 0 928 0
Adrien 0 0 0 0 0 0 0 0 0 845 935 0
Adrienne 699 730 633 405 432 336 342 201 189 335 529 0
Afton 0 964 0 0 0 0 0 0 0 0 0 0
Agatha 383 454 593 850 0 0 0 0 0 0 0 0
Agnes 37 46 68 134 229 372 612 0 0 0 0 0
Agustin 553 589 517 437 543 555 678 642 633 659 578 0
Agustina 0 981 0 0 0 0 0 0 0 0 0 0
....
```

You can see from the above data that the popularity of "Agnes" has been declining since 1900, until finally dropping off of the list in 1970. The name "Agustin" has been on the list for a century, but has never broken into the top 400.

Your python program should work the following way. When it begins, it should read in the data into an appropriate data structure (I suggest that you use Python's dictionary structure). Then present the user with a list of options (text-based menu-driven system). The four options are "(1) Search," "(2) Popularity Over Time," "(3) Top Ten," and "(4) Quit."

When the user chooses "Search," your program will prompt the user for a name, and then search all names that contain the input as a substring (e.g. a search for "sam" will match "Samson," "Samantha," "Sam," etc.). Then you will display a list of all names followed by the year in which they were most popular. The string matching must be case insensitive. If the user inputs the empty string, that should match all of the data, so all names should be printed. The following is example output:

```
...
Agatha 1900
Agnes 1900
Agustin 1930
...
```

After displaying the results, return to the main menu.

When the user chooses option 2, you should ask the user for a name, then display the rankings for that name over all 12 decades. Make your output clean and easy to read. The following is an example of what your output could look like:

Rankings for "Agatha":

```
1900  1910  1920  1930  1940  1950  1960  1970  1980  1990  2000  2010
--------------------------------------------------------------------
383   454   593   850   0     0     0     0     0     0     0     0
```

In this case, you look for an exact (case insensitive) match to the name that was input. If no match is found, display an appropriate message and return to the main menu.

When the user chooses option 3, you will ask the user for a year, and then display the ten most popular names (in order) for that year. For example, if the user input 2000, the output should look like the following:

```
Most popular names for 2000:
----------------------------
1. Emily, Jacob
2. Hannah, Michael
3. Madison, Matthew
....
```

Note that there will be a male and female name for each rank, although gender is not indicated in the file and you may display them in either order. After displaying the results, return the user to the main menu.

If the user chooses option 4, the program ends.

A text file containing the name data is available in the class Drive.

Part 3 - Give your overall evaluation of Python

Write a one-page **evaluation of Python** based on the criteria in Chapter 1 of your text. Please clearly state and justify your arguments.

What to Hand In

On the due date, submit all of your files to canvas.