Individual Programming Project Option 3: New York State Statewide COVID-19 Testing

Overview

The New York State Department of Health made public the information on the number of tests of individuals for COVID-19 infection performed in New York State beginning March 1, 2020, when the first case of COVID-19 was identified in the state. Clinical laboratories electronically conduct reporting of SARS-CoV2 laboratory testing results to the DOH[1]. The most recent data set is over 600 KB in size. Although it will fit into your computer's memory, it might make a computer behave sluggishly. Even so, the data set present on the Linux Lab is about 99 KB in size for simplicity, containing information from March 2 to April 21 in 2020. The data set is stored in a comma-separated-values file, i.e., a *csv* file. This means that each field in each row is separated from the adjacent fields by a single comma, and that, at least in the dataset, no field contains any commas. *(The version of the data set provided on the Linux Lab server still has this first row, which contains the column headers describing each column of data.)*

This programming project is designed to give you practice in processing datasets. It should process information based on a data structure of your design to answer specific questions about the data set.

Information About the Original Data Set

The data set is part of the New York State Department of Health website and can be found here:

https://health.data.ny.gov/Health/New-York-State-Statewide-COVID-19-Testing/xdss-u53e

You may find it interesting to visualize the data as a customized visualization:

https://health.data.ny.gov/d/xdss-u53e/visualization

or on the NYSDOH COVID-19 Tracker: https://covid19tracker.health.ny.gov/.

The *NYSDOH* website for the testing results gives you the means to download the data in various formats. A file in *csv* format, in case you are not familiar with it, is a comma-separated-values file. A comma-separated-values file is a plain text file in which each line represents a single record, and within the line, commas separate the individual fields of the record. (Fields can also contain commas if they are within quoted strings, e.g., "Manhattan, New York" is a

[1]Testing Results are reported via the Electronic Clinical Laboratory Reporting System (ECLRS) before the DOH Division of Epidemiology's Bureau of Surveillance and Data System (BSDS) monitors the ECLRS reporting and ensures that all positives and negatives are accurate. Starting September 30, 2020, this data also includes pooled/batch tests reported by institutions of higher education. This is also known as *surveillance testing* and not performed by a clinical laboratory.

single field.) Spreadsheet applications let you import `csv` files to view their contents by rows and columns. The dataset that can be downloaded on the Linux Lab contains count records for the first 51 testing days across all New York state counties. Each row represents testing counts, and has six columns, which means that there are six different pieces of information for each.

The data set that is downloaded will have as its first row, the labels of its columns in this order from left-to-right:

1. `Test Date`; the date that the test result was processed by the NYS ECLRS[2].

2. `County`; the county of residence for the person tested.[3]

3. `New Positives`; the number of new persons tested positive for COVID-19 infection on the test date in each county.

4. `Cumulative Number of Positives`; the running total for the number of persons tested positive for COVID-19 infection in each county as of the test date.

5. `Total Number of Tests Performed`; the number of tests of individuals performed[4] on the test date in each county.

6. `Cumulative Number of Tests Performed`; the running total for the number of tests of individuals performed[5] in each county as of the last update to the dataset.

The simplified dataset is available on our server is in the `data` subdirectory of the `cs132` directory. Each valid line in that dataset contains the same six columns. Some of these columns may be empty, but it is extremely rare. Two commas with no intervening characters represent an empty column, and the commas separating each entry determine the columns. This means that a valid line has to contain at least five commas separating the entries (even if the entries are found to be empty). There should not be any invalid lines in the file at all, but if any are found, the program should handle them by ignoring them.

---

[2] *Data is updated daily to reflect tests completed by 12:00 am (midnight) the day of the update (i.e., all tests reported by the end of the day on the day before the update).*

[3] [3] Test counts are assigned to a county based on this order of preference:
   1. the patient's address,
   2. the ordering healthcare provider/campus address, or
   3. the ordering facility/campus address.

[4] This total includes positives, negatives, and inconclusive results.

[5] This total includes positives, negatives, and inconclusive results.

Instructions

The program must be run with one command line argument, which is the name of the `csv` file containing the data. The program should be named `report.py`. If the testing data were stored in a file named `testresults.csv`, then proper usage of this program would be

```
report.py testresults.csv
```

which will cause the program to read the `testresults.csv`, and for line in the file, it will process that line according to the instructions below.

As mentioned above, each row in the data file has six fields. Following is a brief description of each field and what it contains, from left-to-right:

- `Test Date`; a string formatted as a date timestamp in the format of `MM/DD/YYYY`, e.g. `03/01/2020` for the date of March 1, 2020
- `County`; a string, corresponding to one of the 63 counties of New York, such as `"Albany"` or `"New York"`
- `New Positives`; a positive integer
- `Cumulative Number of Positives`; a positive integer
- `Total Number of Tests Performed`; a positive integer
- `Cumulative Number of Tests Performed`; a positive integer

A line might look like this:

```
04/01/2020,New York,470,7843,1002,19599
```

Your program should read all of the lines in the file, creating a `count` record for each line and storing that into a complex data structure. Once it has done that, it must enter an interactive loop that prompts the user for the name of a county, such as "New York" or "Nassau", e.g.

```
Enter the name of a county, or enter "quit" to quit:
```

to which the user might respond with a valid name.

The program should compute the following information, sorted by increasing `Test Date`:

- the total number of new positives
- the total number of test performed
- the frequency ratio of positives to tests performed on the specified test date as a percentage
- the frequency ratio of cumulative positives to cumulative tests performed on the specified test date as a percentage

and the following statistics for the specified county:

- The test date when new positives were first discovered,
- The highest total number of tests performed and the test date of that performance
- The highest number of positives, and the test date of that discovery
- the average positivity frequency ratio, as a percentage
- the average cumulative positivity frequency ratio, as a percentage

For example, with this fictional data, the information should be presented on the standard output stream like this (*this fictional example outputs is shortened as . . . for discussion simplicity*).

```
$ report.py testresults.csv
Enter the name of a county, or enter "quit" to quit: Suffolk

Suffolk County:
Test Date    Positive    Tests Performed    Frequency(Cumulative freq.)
03/02/2020          0                  0                0.00%  (0.00%)
03/03/2020          0                 10                0.00%  (0.00%)
. . .
04/05/2020       1082               2211               48.97%  (43.64%)
04/06/2020       1030               2323               44.34%  (43.69%)
. . .
04/20/2020        492               1482               33.19%  (40.76%)
04/21/2020        700               2188               31.99%  (40.49%)

Statistics for Suffolk County
First Positives discovered: 03/08/2020
Highest number of tests performed: 3889 on 04/08/2020
Highest number of positives discovered: 1569 on 04/08/2020
Average positivity frequency: 24.48%
Average cumulative positivity frequency: 27.66%

Enter the name of a county, or enter "quit" to quit:
```

Entering '`quit`' should terminate the program. Although it is not necessary, the program could output a departure message before the program terminates.

Error checking:
If the program is called without the command-line argument, it is an error and the program should report it and exit immediately.

If the file on the command line cannot be opened, the program should display a specific message related to the error and then exit.

The program does not have to check that the data file is in the correct format.

If the user passes an invalid entry at the prompt, the program should report that the name is not one of the county names and should report an alphabetical list of all possible county names to the user.

Typing the '`quit`' command should be the only way for the user to exit out of the program and enter back to the command line.

Program Considerations:
As a reminder, the program must have proper documentation and comply with the Programming Guidelines. The output should be what is described above and nothing else.

The program should ideally use a data structure of your design and use functions to perform the computations before the interactive prompt begins. The main program should be very simple – after accepting and validating the argument of the script and creating your data structure, it should prompt the user and get the user's commands within a loop to proceed with the user's responses.

- It should write an error message if the user does not enter a valid entry.
- When the user enters a valid entry, it should call a function to do whatever is requested.

Testing:
All programs must be thoroughly tested before they are released to users. I suggest that you create small subsets of the data to test your program. To do this you can use various filters. Which ones and how to use them I leave to you to figure out. The full, simplified data set is titled `New_York_State_Statewide_COVID-19_Testing.csv`, located in the `cs132/` sub-directory below:

```
/data/biocs/b/student.accounts/cs132/data/open_datasets/
```

Do not copy this entire file into your home directory on the Linux Lab because it may use up your valuable disk space allocation. Instead, make small subsets of it in your home directory. You can of course secure-copy it onto your personal computing device.
Try creating some sample input files of a small size and manually figure out what the outputs should be. Run your program and make sure that your output matches the one you manually computed.