Programming Project 05

This assignment is worth 45 points (4.5% of the course grade) and must be **completed and turned in before 11:59 PM on Monday, February 17, 2020.**

Assignment Overview

1. Functions
2. File input and output
3. try-except

Assignment Background

Vaccinations have become a concern for some people resulting in an increase in diseases that had become increasingly rare. Measles which is part of the MMR (measles, mumps, and rubella) vaccine has been in the news. According to the CDC, in 2017, measles caused an estimated 110,000 deaths worldwide. An important part of the science of vaccines is herd immunity. Herd immunity describes how a population is protected from a disease after vaccination by stopping the germ responsible for the infection being transmitted between people. In this way even people who cannot be vaccinated can be protected, e.g. babies. For example, measles is very contagious. Before the use of the measles vaccine, every person with measles would infect another 10-15 people and so the disease would spread very quickly. To achieve herd immunity for measles at least 90-95% of the population need to be vaccinated. In this project we will use CDC data to examine measles herd immunity in the US.

Use of advanced data structures such as lists, sets, and dictionaries are prohibited.

Project Specification

We provide the file, `MMR.txt,` which has a title line followed by a header line and then lines of data in the format. This format allows you to use slicing to extract the data:

- `state`: the leftmost 25 characters of the line (string)
- `percentage`: the rightmost 4 characters of the line (float)

We also provide a program with skeletons of the functions: `proj05.py`

Your code should include at least the following functions. Feel free to create more

1. **`open_file() → fp :`**
   This function takes no parameters, and then uses the try-except format to prompt for a filename, open the data file and return the file pointer if successful. Your function should be able to catch the error and print the error message if it fails to open; and then reprompt. It will reprompt until successful.
2. **`get_us_value(fp) → float:`**
   This function takes as an input, `fp:` the file pointer for the data file.

The value for the overall United States MMR coverage is in the line with the label “United States”. Find and return that value. Read the file line-by-line, use slicing to extract the “state” and value, keep reading until you find “United States”. (Use `fp.seek(0)` to rewind the file to the beginning as your first line of the function. Use a pair of `fp.readline()` statements to skip the title and header lines.) This function does not print anything.

a. Parameter: file pointer
b. Returns: `float`

3. **`get_min_value_and_state(fp)` → `string, float`:**

This function takes as an input, `fp:` the file pointer for the data file.

This function should return the minimum value in the file and the associated state. Read the file line-by-line, use slicing to extract the “state” and value. The algorithm for finding a minimal value is the begin with some large value, let’s name it `min_value` (since these are percentages any value over 100 will work). You will need a second variable, let’s name it `min_state` to record the associated state name. Examine the value in each line, if you find a value that is smaller than `min_value` , you have found a better minimal value. Remember to update the `min_state`. If you do that with every line, you will have found the minimal value after reading all lines. Return the minimal value and the associated state.

Your program should ignore the lines where the values are `“NA”` (Use `fp.seek(0)` to rewind the file to the beginning as your first line of the function. Use a pair of `fp.readline()` statements to skip the title and header lines. There is only one minimal value so we do not have to consider ties.) This function does not print anything.

a. Parameter: file pointer
b. Returns two values: `string, float`

4. **`get_max_value_and_state(fp)` → `string, float`:**

This function takes as an input, `fp:` the file pointer for the data file.

This function should return the maximum value in the file and the associated state. It is similar to the **`get_min_value_and_state`** function. There is only one maximal value so we do not need to consider ties. This function does not print anything. Your program should ignore the lines where the values are `“NA”.`

a. Parameter: file pointer
b. Returns two values: `string, float`

5. **`display_herd_immunity(fp)` → `None`:**

This function displays all the states whose coverage is less than 90%, the minimal value needed for measles herd immunity. Use these strings for the title and header:

- `"\nStates with insufficient Measles herd immunity."`
- `"{:<25s}{:>5s}".format("State","Percent")`

Format the data lines similar to the header

- `state`: string, left justified with a field width of 25
- `value`: float, right justified with a field width of 5 and 1 decimal digit of precision

Your program should ignore the lines where the values are `“NA”`.

a. Parameter: file pointer
b. Returns: nothing

6. **`write_herd_immunity(fp) → None:`**
This function writes into a file named `herd.txt` all the states whose coverage is less than 90%, the minimal value needed for measles herd immunity.

Specifications are exactly the same as the previous function `display_herd_immunity`, the difference is that instead of displaying on the screen you will be writing to a file.

Remember to close the file after writing to it.

a. Parameter: file pointer
b. Returns: nothing

7. **`main()→ None:`**
In `main()` begin by calling `open_file()` to get the file pointer. Then read the header line and display it. Finally, call the other functions to extract, calculate and print values. We provide a `strings.txt` file so you can get the strings and formatting to match the tests.

Assignment Notes

1. Coding Standard 1-9 will be enforced.
2. Use of `fp.seek(0)` to force starting reading at the beginning of a file is inefficient so it should rarely (as in never) be used. In fact, we will prohibit its use in future projects. Why? Reading from files on disk is slow compared to getting information from memory (that is thousands to millions of times slower). The proper way to do it is to read information from a file ONCE into a data structure in memory and thereafter always refer to the data structure. However, we haven’t learned the appropriate data structures yet so we are using `fp.seek(0)`in this project.

Assignment Deliverable

The deliverable for this assignment is the following file:

`proj05.py` – the source code for your Python program

Be sure to use the specified file name and to submit it for grading via the **Mimir** before the project deadline.

Test Cases

Function Test: `get_us_value()`

```
instructor: 91.9
student: 91.9
```

Function Test: `get_min_value_and_state ()`

```
instructor: ('Guam', 76.3)
student: ('Guam', 76.3)
```

Function Test: `get_max_value_and_state ()`

```
instructor: ('New Hampshire', 98.8)
student: ('New Hampshire', 98.8)
```

Test 1

```
Input a file name: MMR.txt

MMR ( measles, mumps, and rubella (German measles)) vaccination coverage
among adolescents aged 13-17 in 2018

Overall US MMR coverage: 91.9%
State with minimal MMR coverage: Guam 76.3%
State with maximum MMR coverage: New Hampshire 98.8%

States with insufficient Measles herd immunity.
State                    Percent
Arizona                   88.3%
Idaho                     88.5%
Kansas                    86.5%
Missouri                  89.4%
Nevada                    89.0%
Ohio                      89.5%
Oklahoma                  87.7%
Texas                     83.1%
Utah                      89.5%
Washington                88.3%
West Virginia             86.5%
Guam                      76.3%
```

Grading Rubric

```
Computer Project #05

General Requirements:

Scoring Summary

  ( 5 pts) Coding Standard 1-9

              (descriptive comments, function headers, etc...)
```

Implementation:

(5 pts) open_file function (no Mimir test)

-3 Did not use try/except

(7 pts) get_us_value function

(7 pts) get_min_value_and_state function

(7 pts) get_max_value_and_state function

(7 pts) write_herd_immunity function

(7 pts) Test1 (includes testing display_herd_immunity)

Note: hard coding an answer earns zero points for the whole project -
10 points for not using main()