CIS 5357

Assignment # 10

(35 points)

**Due Date: by 11:59 pm on Monday, November 30, 2020**

**Caution:**

**The instructor has found many submissions with duplicate or similar code. The instructor expects individual effort on each and every submission. Assignments and exams assigned in this course are neither group projects nor any kind of group activity or collaboration are sanctioned. Such activities will be treated per the Academic Dishonesty policy as stated in the course syllabus. Each submission will be closely examined for plagiarism**. **A slightest hint of duplicate or similar code will be examined for academic integrity violation. Such submissions will automatically receive a grade of zero and reported to the Graduate College for further disciplinary action. To avoid such scrutiny, please do your own work. Please consult your instructor if you any questions or need clarification about assignments and examinations.**

1. **Requirements for Assignment 10:**
2. Name your Jupyter Notebook ‘YourName-Assignment10.ipynb’
3. Include your name and submission date as level 2 headings in the first cell of the notebook. Include the following text after the level 2 heading in the same cell:

<font color= “red”>

## Assignment Grade:

##

</font>

1. Insert a markdown cell below the cell containing your name and submission date. Include the following text in that cell:

<font color=’red’ size=1>

### Grading Comments – Totality of all points noted below resulted in a reduction of xx Points

</font>

1. Insert a markdown cell with level 3 heading for the Program Name and its objective
2. Please access the ‘mileage.csv’ file in the following path: **“/Users/cis\_developer/CIS5357Fall2020/mileage.CSV”.** If you are not sure how to create this path on your computer (windows or macs), now is the time to learn it. The instructor will stop grading and assign a zero if the filepath used is different from the one listed above.
3. Upload your source code file (.ipynb) to the Assignments section of Canvas BEFORE 11:59 pm on Monday, November 30, 2020. No other files need to be uploaded.
4. **NO LATE ASSIGNMENTS WILL BE ACCEPTED. ASSIGNMENTS SENT VIA EMAIL AS ATTACHMENTS WILL ALSO NOT BE ACCEPTED. YOU ARE ALLOWED A MAXIMUM OF TWO ATTEMPTS TO SUBMIT YOUR FILE BEFORE THE DUE DEADLINE. YOU MISS THE DEADLINE, YOU LOSE IT. So, please start early to have a chance at getting any problem resolved before the submission deadline.**

**Introduction**:

You will design a python program that will read mileage data from the mileage.CSV file, compute MPG, and save the results to another CSV file. The focus is here is on the inclusion of Exception Handling to prevent the program from crashing or an abrupt halt due to errors.

**Specifications:**

* 1. Open the mileage.csv file to bulk read all mileage records from that file into a Pandas Dataframe. Each record in the mileage.csv file contains two fields, mileage driven and gallons of gas consumed. The first line of the CSV file contains column headers.
     1. You should catch any errors with invalid file path or filename errors. When an exception is handled, display appropriate error message and terminate the program.
     2. In addition, ensure that you convert the data read in to floating point values. Handle the exception if non-numeric data are present by dropping the rows containing non-numeric value in either the miles or gas column. Display an appropriate error message, drop those rows, and then display the final dataframe after the exceptions are handled.
     3. Return the clean dataframe.
  2. For each mileage record in the clean dataframe, calculate miles per gallon as the third column labeled ‘mpg’
     1. Compute the miles per gallon as:

mpg = miles driven/gallons of gas consumed, round the mpg values to 2 decimal places.

Since a division is involved, you should check for divide by zero exceptions. If such an error occurs, display an appropriate error message, drop those rows from the dataframe, and then display the resulting clean dataframe.

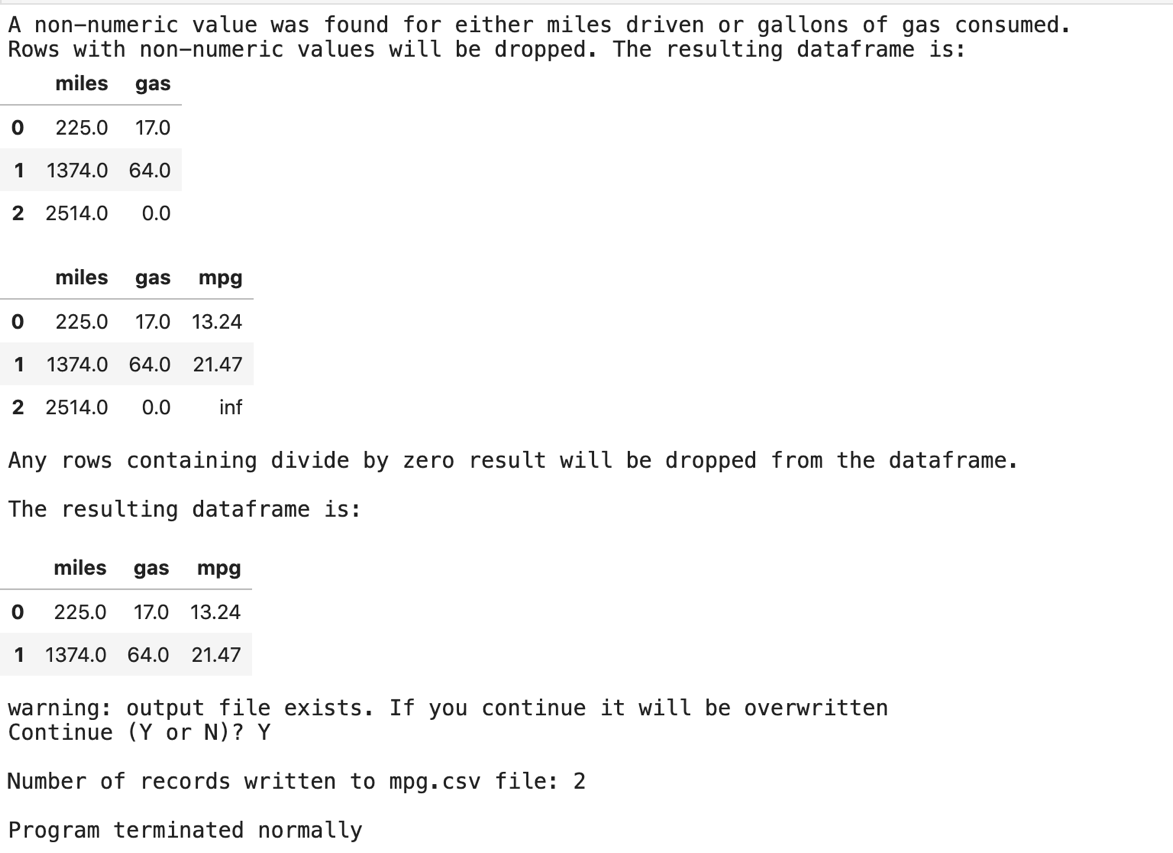
* + 1. Save the data in the computed dataframe from (b-i) above (miles driven, gas consumed, and mpg) to another CSV file named ***mpg.csv***.
    2. Because the program will write to an output file, it should check for invalid path and filename exceptions, display appropriate error message if such an exception is raised and then terminate the program.
    3. If the file path and filename are valid, the program should warn the user that existing file will get overwritten and then give user a choice to continue or terminate the program to prevent overwriting an existing file. If the user chooses to continue and overwrite an existing file, the program should write the data in the computed dataframe to the CSV file; inform the user as to the number of records written to the csv file and that program will terminate normally.
  1. Make sure that both the mileage.csv and mpg.csv files are closed at appropriate time.
  2. Download and copy the mileage.csv file to the folder indicated in requirement # 5 above. Your code should read the data from this location. Do not put this file in local path on your computer. I will execute your notebook such that it will access the mileage.csv file from this path on my computer. Similarly, the output file, mpg.csv, should also be written to the same folder where mileage.csv file is stored. **The instructor will stop grading and a grade of zero will be assigned if your submission cannot find the mileage.csv and mpg.csv files in the specified location.**
  3. *The objective is to add Exception Handling to python program.*
  4. **The Dataframe(s) you create must have meaningful names,** not DF, DF1, etc. The name you assign must have the letters **df** appended to it e.g., actors\_df or movies\_df
  5. Where questions are asking you to provide a written response, please include your answers in **markdown** cells.

**SPECIAL INSTRUCTIONS:**

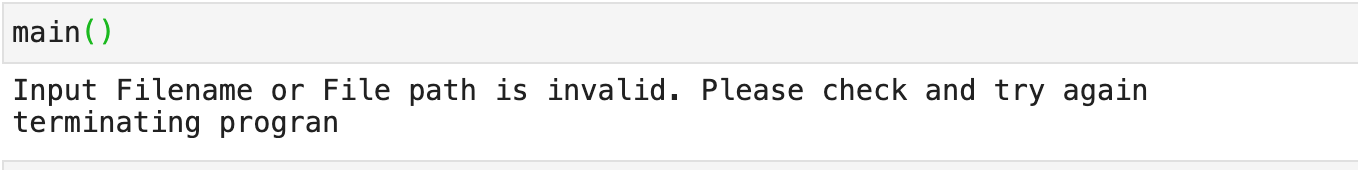
Since the focus of this assignment is on exception handling, your entire code will be divided into three modules

1. get\_data() which will read the mileage data from the mileage.csv file. Returns the populated dataframe with only numeric data to the main() function.
2. compute\_mpg() which will accept populated dataframe returned by the get\_data() function, compute the mpg, and return the dataframe containing miles, gallons, and mpg columns to the main() function. This dataframe must have all rows with divide by zero indicator dropped before returning to the main() function.
3. write\_results() which will accept the dataframe returned by the compute\_mpg() function, write the data to mpg.csv file, and display needed information to the user.
4. Main() function will call get\_data(), compute\_mpg(), and write\_results() functions in turn.
5. Your program should handle the following exceptions:
   1. any IOError exceptions that are raised when the files are opened to read data from or write results to.
   2. any ValueError exceptions that are raised when the items that are read from the file are converted to numeric data.
   3. Any DivideByZero exceptions that may be raised when miles driven are divided by gallons of gas consumed, and gallons of gas consumed value read in may be zero.
   4. Any FileNotFoundError exception raised when an invalid file path or filename are used.

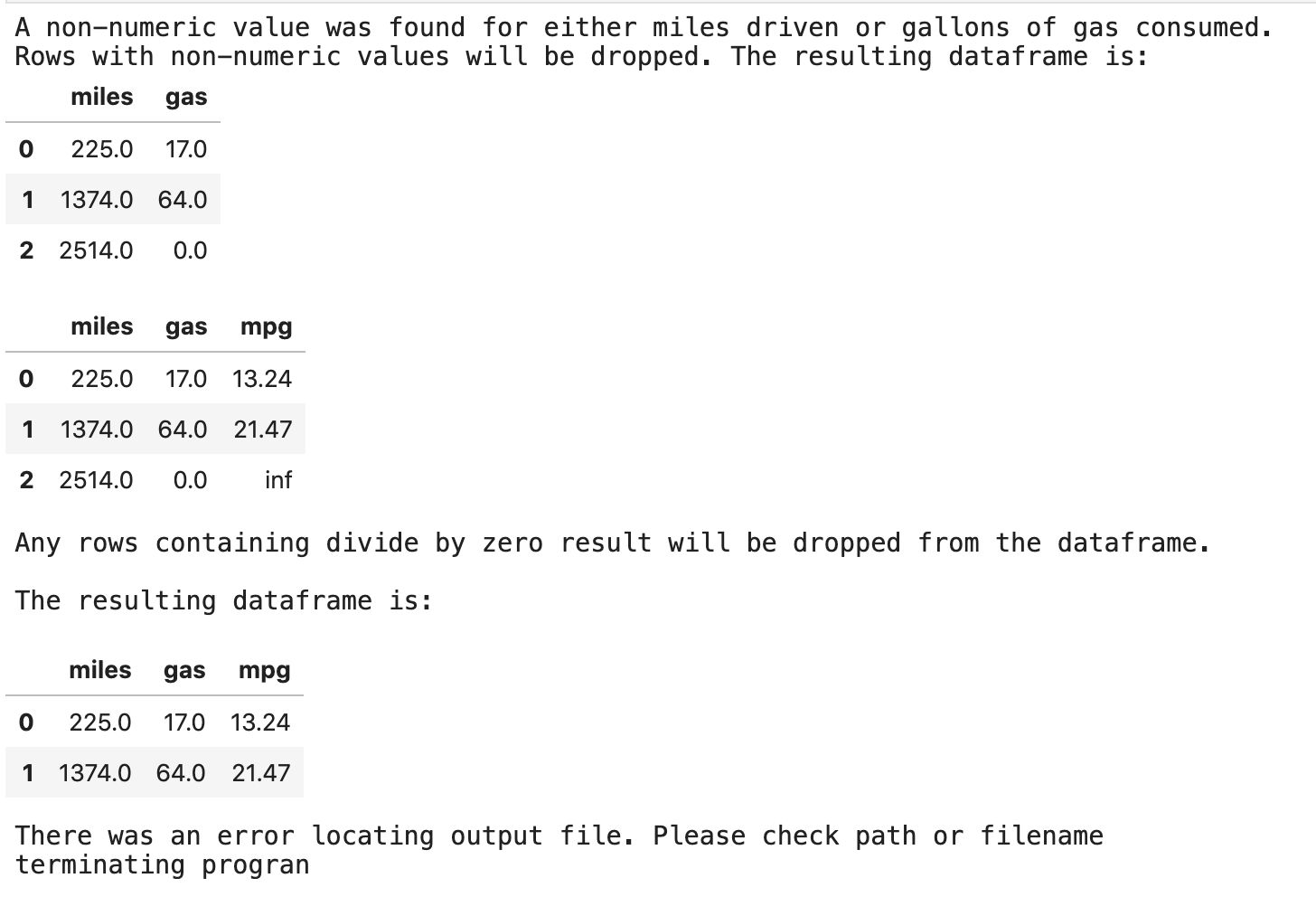
If there were no file path or filename exceptions, the complete output from the program should look like this:



The console output if there was an invalid file path or filename for the input file:



Console output when an invalid file path or file name is provided for the output file.



Output mpg.csv file successfully written:

miles,gas,mpg

225.0,17.0,13.24

1374.0,64.0,21.47