Computing Structures – Fall 2022
Project 3
Due: October 21st 2022 at 11:59 PM

Objective:
The goal of this project is to create the DataFrame class and perform queries in C++ as outlined in the project description.

Description:
In this project you will implement the DataFrame class along with the all the methods that are represented in the class definition. You are required to read a csv file, do operations on the data stored and display the queried contents from the dataframe.
DataFrame is a table of data with rows and columns – columns have names associated with them also - we call these as headers. You may or may not have headers and this will be mentioned in the input file(more details follow). You are going to store the given information in a csv file as a vector of DFrow objects and answer queries from the input file.

Input explanation:
The input file is different from the csv document that is given. The input file is read in via redirected input (like project 1). The input file contains in this specific order, the number of rows/records, number of columns, if the csv file contains headers or not, the csv file's name and followed by a series of set of character options/command and a corresponding information. Here is a sample input file:

```
18                          <- number of rows
6                           <- number of columns
true                        <- header exists or not (can be either true or false)
fileInput1.csv              <- name of the csv file

F Ivan                      <- F is the option to find the record with the name Ivan
F Alex                      <- F is the option to find the record with the name Alex
D                           <- D is the option to display all the records along with the headers if exists

A Age                       <- find the average of the Age column
A Sex                       <- find the average of the Sex
A Height(in)                <- find the average of the Height(in) column

Q Sex                       <- find the frequency of unique values in the Sex column

X Age                       <- find the max value of the Age column
I Age                       <- find the min value of the Age column

C 2 Name Sex                <- display a subset of 2 columns - Name and Sex
C 3 Age Height(in) Weight(lbs)  <- display a subset of 3 columns - Age, Height(in) and Weight(lbs)

R 5 10                      <- display rows 5 through 10
R 12 13                     <- display rows 12 through 13
```

S 2 Name Sex 5 8 *<- display subset of the dataframe 2 columns(Name and Sex) with rows from 5 through 8*

You may assume that the frequency query will be performed only on a categorical column. You'll have to throw an ***exception*** when you are trying to search for a name that is not present in the dataframe.

CSV file:

The name of the csv file to open and read is given in the input file. You will use the fstream functionalities to open and read the contents of the csv file. Store the names of the headers if there are any from the CSV file to the array of strings called headers in the DataFrame class. Continue to read in all the data from the csv file and store it in the vector called data in the DataFrame class. Notice that the data type of the vector elements is DFrow object. Each row is an element (of type DFrow) in the vector data.

Class definition:

A boiler plate source file has been provided along with this project specification. The class definition and a main function have also been given with comments to get you started with the project.

A sample output file will be provided (in the next few days) for the given corresponding input file. Your output is supposed to exactly be the same as this output. You need to essentially follow this format of the output. This is necessary for your program to pass the autograder.

Fstream input:

The csv file is to be read using fstream. The name of this file to be read is given in the redirected input file. The data read from the csv file is to be stored in the class DataFrame.

Redirected input:

Redirected input provides you a way to send a file to the standard input of a program without typing it using the keyboard. To use redirected input in Visual Studio environment (on windows), follow these steps: After you have opened or created a new project, on the menu go to project, project properties, expand configuration properties until you see Debugging, on the right you will see a set of options, and in the command arguments type <"input filename". The < sign is for redirected input and the input filename is the name of the input file (including the path if not in the working directory).

If you use macOS or linux (or windows using powershell), you may use the terminal to compile and run your program using g++. This is the compiler that we use in the autograder on GradeScope (you may need to install g++ on your computer). Once you installed g++, you may compile your program using the following command:

```
g++ project3.cpp -o p3
```

You may then run the program with the redirected input using the following command:

```
./p3 < input1.txt
```

Make sure your program and your input file are in the same folder.

A simple program that reads the input file and displays everything that was read is given below.

```
#include <iostream>

using namespace std;

int main ()
{
      int numRows, numCols;

      cin >> numRows >> numCols;
      cout << numRows<< numCols;

      return 0;
}
```

Submission:

Submission will be through GradeScope. Your program will be autograded with test cases and then hand graded to check for documentation and if you have followed the specifications given. You may submit how many ever times to check if your program passes the test cases provided. Test cases will be released at the beginning and later other test cases will be released while grading. For the autograder to work, the program you upload must be named as **project3.cpp**.

Your final submission must contain your source file named **project3.cpp**. You access GradeScope using the tab on the left in our course page on canvas.

Sample output file for corresponding input files will be released. The input1.txt file given is a simple input file that will help you understand the project, more complicated ones will be released later and used for grading.

Constraints:

1. In this project, the only header you will use the header files that are given to you in the boiler plate code. Using other or excess header files will be subject to a heavy grade penalty for the project.
2. None of the projects is a group project. Consulting with other members of this class on programming projects is strictly not allowed and plagiarism charges will be imposed on students who do not follow this.
    - Please review academic integrity policies in the modules.

How to ask for help:

1. Check FAQ discussion board of the project first.
2. Email the TA with your precise questions.
3. Upload your well commented program to CodePost.
    a. This is a website which is used to share code.
    b. You will upload your program and I can view it and comment on it.

c. Here is the invite link to our class for the summer.
d. Once you join the class on CodePost, you can upload your program to the Project 1 assignment.

    Note: Your program will not be auto graded at CodePost, this is just for when you want to ask a question and a place where I can look at your program and comment on it. CodePost is great for live feedback. GradeScope is the place where you should submit and where your program will be autograded.