EGR 125 Due date: ________________
Introduction to Engineering Methods (C++)
File: N125P4G

Programming Assignment #4: Class Runner

Write a program using a class named ***Class Runner*** (which you will also write) that will allow a runner to keep a permanent record of runs for exercise. A class diagram for ***Class Runner*** is shown below.

Class Runner	
Member Data: - First: string - Last: string - FileName: string - Hours: int - Minutes: int - Seconds: int - Month: int - Day: int - Year: int - Distance: double - Average: double	Comments: All data members must be private (-)
Member Functions: + Login( ); void + EnterRun( ): void + FindAverage( ): void + DisplayRunResults( ); void + Summary( ): void . . + etc (other member functions)	Comments: - Let all member functions be public (+) Other member functions can be added, but are not required.

Program Requirements:

1) Creating a class: The program should define **class Runner** (or pick your own name) according to the class diagram provided.
   - Use the data members indicated and member functions indicated. Additional data members and member functions can be added if you wish.
   - The main program should be brief. It should create an object and use the functions. Avoid adding new variables in main to perform class functions.
   - All data members must be private. All member functions must be public.
   - Use separate header and implementation files for the class (Runner.h and Runner.cpp)
   - Allow the user to re-enter bad inputs (with appropriate error messages), such as:
     - Hours: Must be non-negative
     - Minutes, Seconds: Must be 0-60
     - Month: Must be 1-12
     - Day: Must be 1-31
     - Year: Must be current year
     - Distance: Must be positive
     - Menu selection: Check for appropriate value

2) Main program: The main program should make an object using class Runner. Additionally, use class functions to:

- Prompt the user to enter their first and last name (in member function Login( ).
- Ask the user to indicate if they are a new user or a returning user (also in member function Login( ).
  - If they are a new user, create a new file to store their data.
  - If they are a returning user, attempt to open and close the file. Give an error message if the file cannot be located.
- Give the user options to:
  A. Add a new run.
     - Prompt the user to enter the date (month-day-year), distance (in miles), and time (in hours, minutes, and seconds)
     - Display a summary of the run, including average in minutes/mile.
     - Give the user the option of saving the run to the file.

  B. Display a summary of all runs, including date, distance, time, and average

  C. Exit the program.

Testing your Program

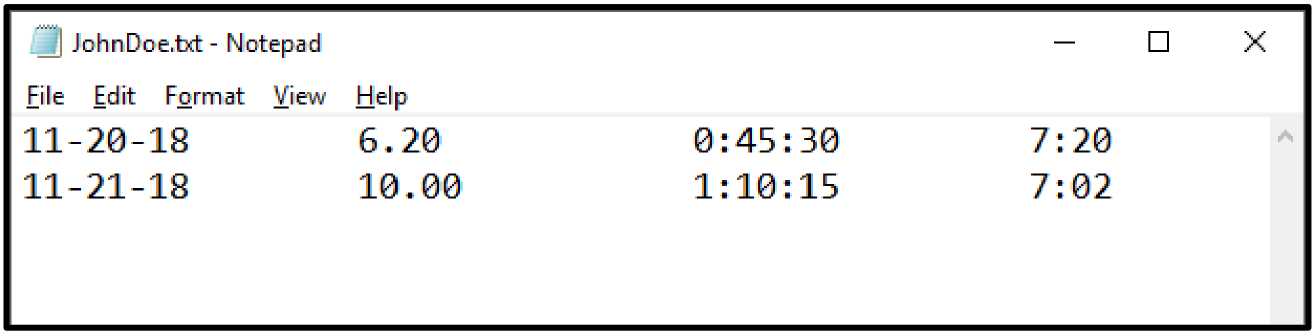
Test your program for the following cases:

- Case 1: Use the example provided above (new user and add one run), but use your name. Print the program output and also print the data file.
- Case 2: Use the example provided above (returning user and add a second run), but use your name. Print the program output and also print the data file.
- Case 3: Add 8 more runs to the file created above, with distances varying from 1-20 miles. Running averages should be reasonable (5 min/mile to 15 min/mile perhaps).

Extra Credit Suggestions: (for a maximum of 10 additional points on the program grade)

1. Add an additional member function (and menu option) allowing the user to find the total number of miles run, the total time, and the total average for all runs in their file.
2. Give the user the option of working in miles or kilometers. Display both values and store both values in the file. (So if a distance in km is entered, calculate distance in mi, but if a distance in mi is entered, calculate a distance in km. In either case, show both values. Additionally, show two averages: one in minutes/mile and one in minutes/km.
3. Give the user the option of displaying a summary of runs for each month.
4. Use your imagination!

Sample data file: (For the example shown on the next page)



```
11-20-18          6.20              0:45:30           7:20
11-21-18          10.00             1:10:15           7:02
```

Sample program output: (The output doesn’t need to look exactly like this.)

```
Case 1:
Welcome to the Online Running Log
Enter your first name:  John
Enter your last name:  Doe
Are you a new user (N) or a returning user (R)? N
New file created:  JohnDoe.txt

Would you like to enter a run (Yes or No)?  Yes

Enter date for run:  11-20-18
Enter distance (in miles): 6.2
Enter run time in hours, minutes and seconds: 0 45 30

Run summary:
Date       Distance (mi)  Time       Average (min/mile)
11-20-18   6.20           0:45:30     7:20

Would you like to save the run (Yes or No)? Yes
Run added to JohnDoe.txt

Make a selection:
(R) Enter a run
(D) Display a summary of all runs
(E) Exit the program
Enter R, D or E: E


Case 2:
Welcome to the Online Running Log
Enter your first name:  John
Enter your last name:  Doe
Are you a new user (N) or a returning user (R)? R
File located:  JohnDoe.txt

Would you like to enter a run (Yes or No)?  Yes

Enter date for run:  11-21-18
Enter distance (in miles): 10
Enter run time in hours, minutes and seconds: 1 10 15

Run summary:
Date       Distance (mi)       Time         Average (min/mile)
11-21-18   10.00               1:20:15      7:02

Would you like to save the run (Yes or No)? Yes
Run added to JohnDoe.txt

Make a selection:
(R) Enter a run
(D) Display a summary of all runs
(E) Exit the program
Enter R, D or E: D

Summary of all runs:
Date       Distance (mi)  Time        Average (min/mile)
11-20-18   6.20           0:45:30     7:20
11-21-18   10.00          1:20:15     7:02

Make a selection:
(R) Enter a run
(D) Display a summary of all runs
(E) Exit the program
Enter R, D or E: E
```