CS150 Spring 2016 Project 1 Game Application

All projects in CS150 are to be completed individually.

- Do not work with anyone on it.
- Do not discuss it with anyone other than the instructors and Teaching Assistants.

Do Not Collaborate on the project. It's a violation of the ODU Honor Code.

All projects in CS150 are to be completed individually. Do not work with anyone on it, and do not discuss it with anyone other than the instructor and Teaching Assistants. Do Not Collaborate on this project.

Do NOT Discuss this assignment with anyone other than the instructor and Teaching Assistants. Do Not work together with anyone else, and do not discuss this project, or your solution with anyone. Do NOT Discuss this assignment with anyone other than the instructor and Teaching Assistants.

All projects in CS150 are to be completed individually. Do not work with anyone on any of your projects. Do not discuss it with anyone other than the instructor and Teaching Assistants.

Important:
All projects are to be completed individually. Do NOT collaborate on this project.
Do NOT discuss the project with anyone other than the instructor and Teaching Assistants.
All projects are to be completed individually; duplicate programs will not receive credit.

CS150 Spring 2016 Project 1 Game Application

Due: Electronic copy of .cpp file on Blackboard for lab no later than Friday Feb/19/2016 at 11:59 p.m.
Late submissions will not be accepted under any circumstances.

Important:
All projects are to be completed individually. Do NOT collaborate on this project.
Do NOT discuss the project with anyone other than the instructor and Teaching Assistants.
All projects are to be completed individually; duplicate programs will not receive credit.

Your code will be executed on the systems in Dragas. Make sure that your code compiles and runs on the systems in Dragas before you submit your .cpp file. Review the grading algorithm before you submit your project!

Applications for mobile devices (Mobile Apps) are in great demand for good reason - they can be useful learning tools. You have been tasked with writing part of a simple game program that will help beginner programmers study and understand C++.

The program will begin by prompting for the user information. Next the user is asked for the name of the user's game input file. The file is processed, and calculated results are displayed and also written to the default output file as shown in the sample output shown below.

You have to write an algorithm to solve this project. You are required to demonstrate the use of an algorithm in completing this assignment – meaning that you must include the number and short description of each step in the program documentation (comments)… showing your stepwise approach to solving this problem. See tips at bottom. Example output is shown at the end of this document.

Prompt the user to input the following *player information*, and echo the information to the screen.

- **Full Name** of user - a string that may contain blanks
- Player **Alias** – string that does not contain blanks
- **Gender** – a single character
- **PID** (player ID number) – an integer value

Prompt the user for the name of the file containing the *game score history:*

- **Open** the file and process the following for each game attempt.
    - **Date** – string that does not contain blanks (ex: Feb/14/2016)
    - **Level** – an integer value representing difficulty (1 or 2)
    - **Score** of game – a floating point value.
    - Example shown below – open the sample input files provided.

	difficulty	raw
date	level	score
Nov/1/2015	1	3.996
Nov/1/2015	1	15.111
Dec/8/2015	1	21.919
Jan/6/2016	2	91.014
Feb/1/2016	2	4.445
Feb/3/2016	2	81.001

Calculate and display the following game player information:

- **Total Number** of all plays entered – an integer value
- **Total Number** of level 1 plays – an integer value
- **Total Number** of level 2 plays – an integer value
- **Average** Score – a floating point value. (sum of scores / total number of plays)
- **Average Adjusted Score** – a floating point value.
    - $Average\ Adjusted\ Score = \sqrt{\frac{Average\ \ Score}{Number\ Level\ 2\ plays} \times (SCORE\_ADJ\ )^3}$
    - SCORE_ADJ is a constant value declared in the provided code template.
    - You ***must*** use the math functions pow and sqrt for the above formula.

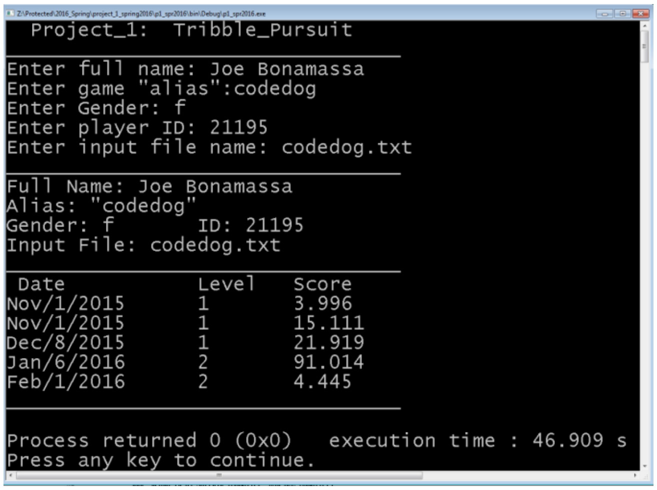
Notes:

1. Remember to include programmer documentation in the source code and in the output.
2. Your program should work for any input values. Graders will follow the directions in your prompts. You should test your program with different data sets.
3. Use a separate prompt for each user input value.
4. Format floating point values in the output with **3** decimal places.
5. Remember to save a copy of the file you hand in. Leave the file unchanged after your submission until you receive a grade on the project assignment. Note that if you submit the wrong file as your solution, this will not earn you credit. You must submit your source code solution in the form of a file with the .cpp extension. Once the solution for a project has been released (usually coinciding with the due date), then no credit can be given or late submissions accepted. It is your responsibility to start early, and submit your solution in plenty of time to account for errors or questions for the lab TA on your submission.
6. Name your source code file using your last name and first initial as follows: lastname_firstInitial_Prj1.cpp ;for example, Meg Griffin would save her source code in a file name **Griffin_M_Prj1.cpp**
7. Remember that you can work in the Problem Solving Lab in Dragas 1103G or the lab located in room 3104 in the E & C S building. Hours for the labs are posted on the CS home page. You can also use the Remote Desk-Top Connection.
8. Do not discuss the project with anyone other than the instructor and Teaching Assistants. Do not collaborate on this project. Projects are to be completed individually; duplicate programs will not receive credit, and will be reported as a violation of the ODU Honor Code.

Submission details: Your TA will review the process of submitting your project in class. You will hand in an electronic copy of your source code. (.cpp file) We will run your program so it is important that you hand in a copy of your source file (**Griffin_M_Prj1.cpp). The file you hand in must have a .cpp extension.**
You need to start this project early – do NOT wait until the last day to complete and submit your solution. Consult with your TA BEFORE submitting, if you have any questions. **Your code will be executed on the systems in Dragas**. ***Your code must compile and run on the systems in Dragas.***

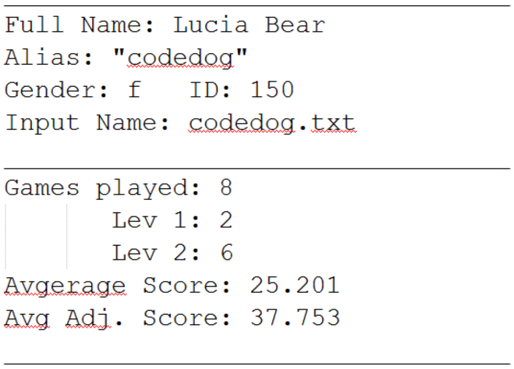
Sample Input and Output

Output to monitor


Z:\Protected\2016_Spring\project_1_spring2016\p1_spr2016\bin\Debug\p1_spr2016.exe

```
  Project_1:  Tribble_Pursuit
__________________________________________
Enter full name: Joe Bonamassa
Enter game "alias":codedog
Enter Gender: f
Enter player ID: 21195
Enter input file name: codedog.txt
__________________________________________
Full Name: Joe Bonamassa
Alias: "codedog"
Gender: f           ID: 21195
Input File: codedog.txt
__________________________________________
 Date              Level     Score
Nov/1/2015         1         3.996
Nov/1/2015         1         15.111
Dec/8/2015         1         21.919
Jan/6/2016         2         91.014
Feb/1/2016         2         4.445
__________________________________________

Process returned 0 (0x0)   execution time : 46.909 s
Press any key to continue.
```

Output to file



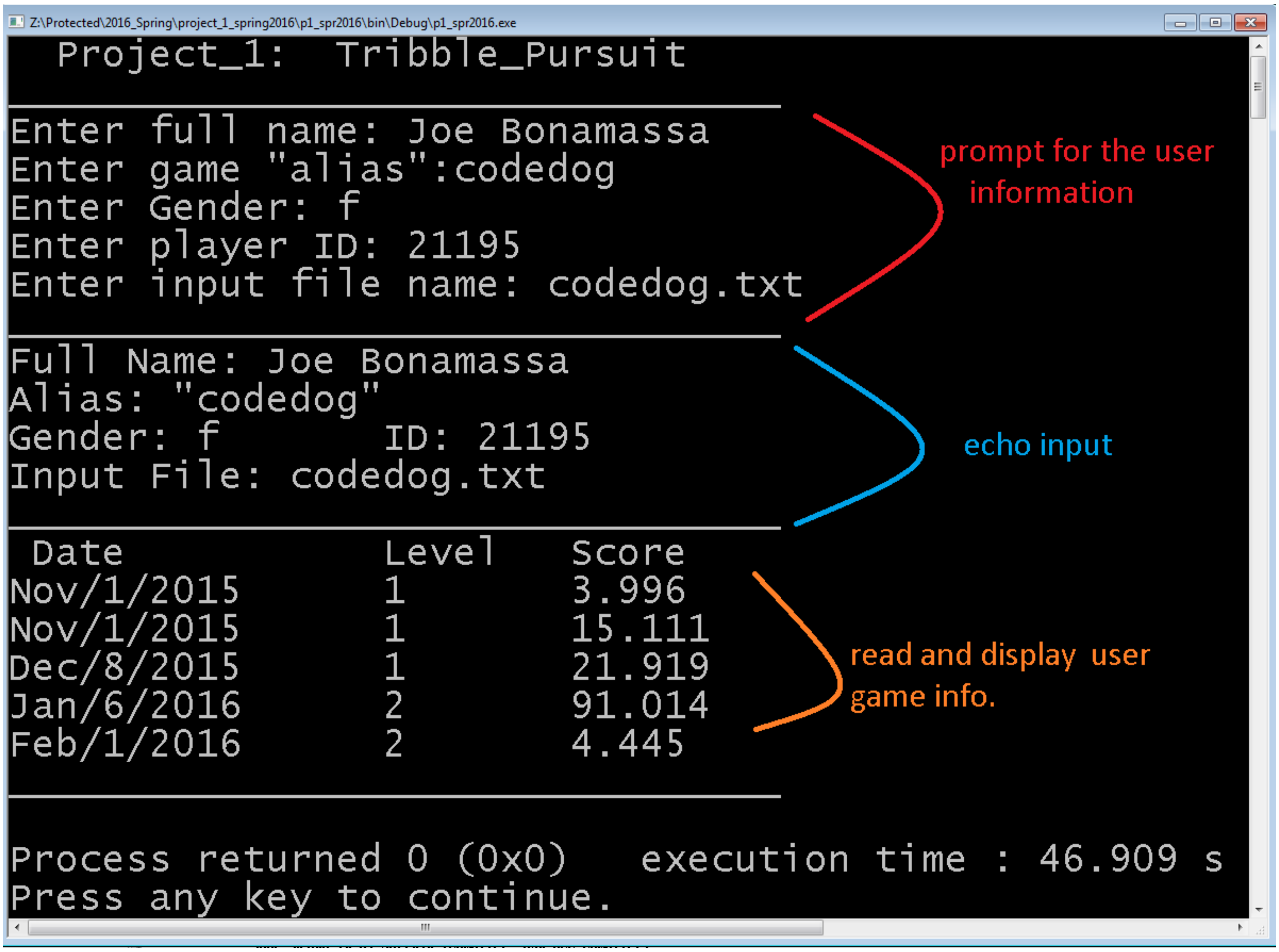
```
Full Name: Lucia Bear
Alias: "codedog"
Gender: f    ID: 150
Input Name: codedog.txt

Games played: 8
              Lev 1: 2
              Lev 2: 6
Avgerage Score: 25.201
Avg Adj. Score: 37.753
```

Two sample input files are provided: nose4code.txt **and** codedog.txt

Helpful Tips

- Start Now. Do not procrastinate.
- Examine the sample code from our textbook, and run the programs in your IDE (codeblocks).
- Go to the recitations to get help from the TAs.
- Go see the TAs during their office hours.
- The algorithm steps you write will prove useful in creating and implementing your own solution. You can expand and fill in a ***preliminary algorithm*** such as the one below.

1. Include header files. Declare the variables, and output file stream variable.
2. Output Program Information.
3. Prompt the user to enter the player info, and store input.
4. Prompt user to enter the input file name, and store input.
5. Open and validate file streams.
6. Set the output formatting.
7. Process file data, and perform calculations.
8. Output the results (to screen and file)
9. Close the file streams.


Z:\Protected\2016_Spring\project_1_spring2016\p1_spr2016\bin\Debug\p1_spr2016.exe

```
  Project_1:  Tribble_Pursuit
_________________________________
Enter full name: Joe Bonamassa
Enter game "alias":codedog
Enter Gender: f
Enter player ID: 21195
Enter input file name: codedog.txt
_________________________________
Full Name: Joe Bonamassa
Alias: "codedog"
Gender: f       ID: 21195
Input File: codedog.txt
_________________________________
 Date           Level   Score
Nov/1/2015      1       3.996
Nov/1/2015      1       15.111
Dec/8/2015      1       21.919
Jan/6/2016      2       91.014
Feb/1/2016      2       4.445
_________________________________

Process returned 0 (0x0)   execution time : 46.909 s
Press any key to continue.
```

prompt for the user information

echo input

read and display user game info.