Report-Assigment\_Part1

# Your Decisions

You should provide a report on your program, consisting of the following information (nothing else):

* Which extra competitor attributes you chose (level, number of scores etc).
  + **Age** as an Integer.
  + **Number** of Scores in an Array.
  + **Levels** as a String.
* How the overall score is calculated. Include an activity diagram.

The getOverallScore method calculates the overall score of a competitor by considering all their scores individually and their level chosen. The totalScore variable is used to keep track of the overall score, and a double array weightedAverage is used to add weight to each event based on its importance. This means the from each game the level gets difficult which result in the score being multiplied by the current weight number. This is read in the loop that iterates through the scores array. For each sore, it multiplies it by the corresponding weight from the weightAverage and adds the result to the totalScore.

After that, an if statement is created to filter the competitor’s level of experience.If the level is “Beginner” the overall score is multiplied by 0.2, if it's "Novice" it's by 0.5, "Veteran" is by 0.8 and finally if the overall experience is "Expert" the overall score is 1.

Once the final totalScore is generated, the return value is restricted to only be in 2 decimal place to ensure a clean and organized output.

Activity Diagram can be found in Assigment\_Part1 folder named: **getOverallScore\_ActivityDiagram.png.**

Status report. Does your program meets the specification fully, or is it incomplete? This can be very brief (don’t provide a list of everything that you were asked to do – the marker knows this).

This application meets the specification fully. Unless the report needed to include this format as specified in the example report:

Text

Description automatically generated

Since the Task 3 never mentioned to use this format in the report. **It has been excluded**.

# Status Report

For the following lists and tables, there are examples in the Testing lecture.

* If necessary, a list of known bugs and a list of requirements that you have not   
  implemented.

As far as I’m aware there are no bugs as everything was already thought of and dealt with.

* Include a table of limitations – these are things that you decided not to handle, either through lack of time or because you think that they are unlikely to occur.

The system is designed to only accept the first name and last name without having the option of a third name. This is mostly due to time constraints to include more complexity to the program and as long as the program works as intended, I’ve ignored that potential scenario.

* Include a table of tests done.

|  |  |
| --- | --- |
| **Testing to do:** | **Complete?** |
| Check the code works by using entryPoint to populate the competitors list. | Done |
| Create Junit testing to test competitor and competitorList to test out the getters and setters and methods | Done |
| Checked every type of scenario of reading external files.   * Mistype CN number by including a letter. * File not found * Read file process got cancelled mid-way. | Done |
| Test writeReport to ensure that each class is communicating effectively with each other. | Done |

# Diagrams.

• Class diagram showing all classes with instance variables, methods and associations.

Class Diagram can be found in Assigment\_Part1 folder named:

**ClassDiagram.png.**

• A sequence diagram showing a task of your choice, but using at least 2 classes.

Sequence Diagram can be found in Assigment\_Part1 folder named:

**SequenceDiagram.png.**

# JavaDoc

Write javadoc style comments for the CompetitorList class, and generate the html. Hand

in a printed copy of the html generated for the CompetitorList class.

For this section I already did Javadoc on most of the classes. However, I had issues with generating html for the CompetitorList. Got an error about not finding api.Jupiter which after re-installing eclipse, Junit5 and trying a different IDE I gave up.