CM1601 Programming Fundamentals Assignment Specification (2021)	
Module Leader	Rajitha Jayasinghe
Unit (Group/Individual)	30
Weighing	30%
Qualifying Mark	40% (both parts)
Learning Outcomes Covered in this Assignment:	LO2. Design, code, compile, test and execute fundamental programming concepts using a high-level programming language. LO3. Construct robust, maintainable programs that use object-oriented analysis and design principles
Handed Out Date	11th Oct 2021
Due Date	20st Dec 2021
Expected Deliverable	Source code and report
Method of Submission	Online. Source + Report (needs to contain whole code, output of the program, assumptions, and the answers for task 1 and 3). You are not allowed to take screenshots of your code. You need to copy-paste code snippets while attaching screenshots of the outputs.
Method of Feedback and Due Date	25th Dec 2021
BCS Criteria Met by this Assignment	

Assessment Regulations

Refer to the "How you are assessed section" in the Student Handbook for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

Penalty for Late Submission

Coursework received late without valid reason shall not be accepted and shall receive no grade, but shall count as one of the assessment opportunities prescribed in paragraph 9 of **RGU Academic Regulation A4 section 4.3.**

It is recognized that on occasion, illness, personal crisis or other valid circumstances can mean that you fail to submit and/or attend an assessment on time. In such cases you must inform the School of any extenuating circumstances through a ***Coursework Extension Form*** or a ***Deferral Request Form***, with valid evidence for non-submission of an assessment up to a maximum of five working days after the assessment submission date. This information will be reported to the relevant Assessment Board that will decide whether a student should be allowed to reattempt without penalty (a deferral). For more detailed information regarding University Assessment Regulations and accessing forms, please refer to the following website: www.rgu.ac.uk/academicregulations www.rgu.ac.uk/academicregulations

Grading

Marks will be awarded for the coursework based on the provided Grading Grid. These marks will be mapped onto a grade scale from A-F as determined by the individual module coordinator

Note:
Subject to approval by external examiner

Coursework Specification

Objective

You are instructed to create a command line application to maintain player and team performances during a T20 cricket tournament. Tournament consists of a one major round. During the round, all teams that are placed in the same group compete each other. Further, the tournament consists of two groups (Group A, Group B). Each group has 4 teams. Overall there will be 8 teams

Tasks

1. All the functionalities mentioned below needs to be fully implemented. You can create a command line menu to navigate

- Store information about teams and players. You have freedom to decide what you need to store exactly. But player and team profiles needs to be realistic and updated after each match. Also you are not allowed to start the league without properly finalizing team and player profiles.
- Before the league starts, the user can delete/edit player and team profiles and system needs to give separate options for each and every functionality. Also you cannot execute these options after starting the tournament.
- Generate random match: Once again, the generations need to be realistic and follow the rules of cricket. There should not be any option to add match information manually. Starting from the toss, everything about the game needs to be random (no hardcoded snippets). The team who won the toss takes the decision (Ball first or bat first). Then program initiates the first innings. Scores of each batsman, balls took for scoring, wickets by bowlers, overs, fall of wickets, the method of the dismissal, final score of the innings will be decided by your algorithm. Next the second innings start and as explained above the algorithm needs to generate random figures.
- Display match summary for a given match.
- Store team/player and match information in a text file. You are not allowed to use a database here.
- Load team/player and match data once you restart the application.
- At a given time the program needs to display the best 5 batsmen(highest run scorers of the tournament) and the bowlers(highest wicket takers)
- At a given time application needs to display the Tournament standings. You can arrange the teams based on the match winnings.

2. Draw flowcharts for the following phase

- Generate random match. If the flowchart is lengthy, please break it into several sub charts.

3. Write a test plan to test the Random match generation. Each test case should have a name, inputs, expected output and the actual output. You do not need to add the status of the test case as it is not required to implement and execute.

Important: If you do not know about the general rules of cricket and T20 games, please refer online materials. Your knowledge about the game will not be tested during the coursework. If you have any modifications, assumptions about the game or the tournament, make sure explain it well in the report. Attributes you need to consider when storing these records were not given and you have the freedom to choose. If you are further developing the application or doing any modifications due to a justifiable fact, please mention that in your report too.

The approaches taken, assumptions made, will be checked thoroughly during the viva. Full marks will be awarded for each criteria **only** if you implement successfully and defend well during the viva.

Marking scheme

Criteria	**Marks**
Flow charts • Generate random match: (no hardcoded snippets/ realistic nature)	5
Implementation • Generate random match: • Store and load player/team and Match information (in a file) • Add/remove/edit information about teams and players. • Display best batsmen and bowlers • Display tournament standings • Display match summary for a given match.	10 5 5 2.5 2.5 2.5
Validations • Following rules and error messages when necessary	2.5
Additional improvement and assumptions • Improvements **only** after the completion of mandatory requirements / Record of assumptions	2.5
Test plan (follow the requested format)	2.5
Viva	10