Beijing-Dublin International College

AUTUMN TRIMESTER FINAL EXAMINATION - (2022/2023)

School of Computer Science

COMP2011J Programming Exam

Dr. Robert Ross
Assoc. Prof. Neil Hurley
Dr. Seán Russell*

Time Allowed: 180 minutes

Instructions for Candidates:

Answer all questions.

BJUT Student ID:_____ **UCD Student ID:**_____

I have read and clearly understand the Examination Rules of both Beijing University of Technology and University College Dublin. I am aware of the Punishment for Violating the Rules of Beijing University of Technology and/or University College Dublin. I hereby promise to abide by the relevant rules and regulations by not giving or receiving any help during the exam. If caught violating the rules, I accept the punishment thereof.

Honesty Pledge:_____ **(Signature)**

Instructions for Invigilators

N/A

Exam Project

To complete this exam you should download the Exam Project zip file from moodle. This file contains a IntelliJ IDEA project with some code for you to use and some input data.

Project Contents

The important files and folders are listed here:
Exam - project folder

- **src** - Folder where you should create your classes
  - **Fixture.java** - This interface you should implement in question 2. Do not change.
  - **Game.java** - This interface you should implement in question 2. Do not change.
  - **Group.java** - This interface you should implement in question 2. Do not change.
  - **Team.java** - This interface you should implement in question 2. Do not change.
  - **Result.java** - This enum is used in question 2. Do not change.
- **test** - Folder where the test classes are stored. You can use these test cases to judge if you have completed some of the tasks.
  - Q1Test.java - A number of test cases assessing Question 1 a
  - Q2aTest.java - A number of test cases assessing Question 2 a
  - Q2bTest.java - A number of test cases assessing Question 2 b
  - Q2cTest.java - A number of test cases assessing Question 2 c
  - Q2dTest.java - A number of test cases assessing Question 2 d
  - Q2eTest.java - A number of test cases assessing Question 2 e
  - Q2fTest.java - A number of test cases assessing Question 2 f
  - Q2gTest.java - A number of test cases assessing Question 2 g

Tests

The test cases in the files above do not cover all of the situations that your code will be tested for. If you pass all of these tests, then you are guaranteed that you have **passed the exam**. To ensure that you have earned a good grade you need to carefully consider the contents of each question and make sure that your code satisfies all of the requirements.

Rules

- This exam is open book, this means that you are allowed to use the course notes and any textbook during the exam.
- This is also an **individual** exam. All submissions will be compared and any students found to have cheated will be reported to the School of Computer Science plagiarism committee and potentially to the university plagiarism committee.

- The consequences of plagiarism are a best case of **failing** the exam, potential **fines**, **not being allowed** to repeat the module for a year, or a worst case of being **removed from the degree**.
- You should only submit working code. If your code does not compile or work correctly you should remove the part that is not working. You will **fail** if you submit code that **does not compile**.
- The classes you define should have good encapsulation. A portion of your grade in **every** question is based on this.

Submission

In order to complete the exam, you must submit your code to the exam quiz on the class page of csmoodle.ucd.ie. It is your responsibility to ensure that this is done correctly. The quiz close automatically at the end of the exam and will not be reopened for any reason. If you are having trouble submitting to moodle, you must email your code to be **before** the deadline has passed.

If your exam is not submitted correctly and on time you will **fail the exam**.

The Exam Quiz

The exam is similar to the quizzes that you have completed in the past, however you will not receive any feedback or score before the quiz is completed. You must ensure that you have tested you code fully before submitting.

Question 1: Basic Programming Problem (Class and Static Method)

a. Define a class called `Q1`, containing a class/static method called `diagonals`. This method should return a `String` containing a pattern and take 2 `int` and 2 `char` parameters. The first `int` is the height and width of the pattern, the second `int` is the size of the gap between the diagonals in the pattern, and the `char` parameters are the characters that should be used in the pattern.

The method should return a string representing some diagonals shown using the first char parameter with the rest of the shape filled in with the second char parameter. The primary diagonal in the shape should be from top left to bottom right and, there should always be a constant number of characters between the diagonals which is determined by the second int parameter.

A runtime exception should be thrown in any of the following situations (The class `IllegalArgumentException` would be a good one to use, but you can use another or define your own if you want):

- The char parameters are equal
- The size is not in the range 7-31 (inclusive)
- The size of the gap between diagonals must be at least 1

The following are examples of the string returned by the code when the method is called with the parameters shown. Note there is no new line character at the end of the pattern.

```
Q1.diagonals(10, 3, '+', ' ');
```

```
1  +     +     +
2   +     +     +
3    +     +
4     +     +
5  +     +     +
6   +     +     +
7    +     +
8     +     +
9  +     +     +
10  +     +     +
```

```
Q1.diagonals(12, 1, 'X', '!');
```

```
1  X!X!X!X!X!X!
2  !X!X!X!X!X!X
3  X!X!X!X!X!X!
4  !X!X!X!X!X!X
5  X!X!X!X!X!X!
6  !X!X!X!X!X!X
7  X!X!X!X!X!X!
8  !X!X!X!X!X!X
9  X!X!X!X!X!X!
10 !X!X!X!X!X!X
11 X!X!X!X!X!X!
12 !X!X!X!X!X!X
```

(20%)

(Question Total 20%)

Question 2: Modelling and Processing Files and Objects

The world cup is currently under way, but the organisers have been very forgetful. They need a system to help them keep track of the progress of the tournament and determine which teams have won in the groups and which games should be played next.

a. Define a class named `FootballTeam` which implements the `Team` interface provided. This class will be used to keep track of the progress of teams in the group stages of the tournament. This means we need to be able to remember or calculate all of the relevant statistics. The methods in the `FootballTeam` class should be implemented based on the following descriptions:

- `public String getName()` - Returns the name of the team
- `public int getWins()` - Returns the number of games the team has won
- `public int getLosses()` - Returns the number of games the team has lost
- `public int getDraws()` - Returns the number of games the team has drawn
- `public int getPoints()` - Returns the number of points the team has scored (3 for a win, 1 for a draw and 0 for a loss)
- `public int getGoalsFor()` - Returns the number of goals that this team has scored in all of their games
- `public int getGoalsAgainst()` - Returns the number of goals that other teams have scored when playing this team
- `public int addGame(Game g)` - Adds the result of a game that was played by this team.[1] In order to function in the testing system, you must also add a constructor with the following signature:
  `public FootballTeam(String name)`.

(15%)

b. Define a class named `FootballGame` which implements the `Game` interface provided. This class will be used to remember the statistics of a single game in the tournament between two teams. The methods in the `FootballGame` class should be implemented based on the following descriptions:

- `public Result getResult(Team team)` - Returns the result of the game (Result.WIN, Result.LOSS, or result.DRAW) for the team passed as a parameter
- `public int getGoalsFor(Team team)` - Returns the number of goals that were scored by the team passed as a parameter
- `public int getGoalsAgainst(Team team)` - Returns the number of goals that were against by the team passed as a parameter

For all of these methods, if the team parameter that is passed does not match either of the teams in the game then an `IllegalArgumentException` should be thrown.

[1] This can be used to either update statistics that are remembered in instance variables or can be remembered so that the statistics can be calculated later.

In order to function in the testing system, you must also add a constructor with the following signature:
`public FootballGame(Team teamA, int scoreA, Team teamB, int scoreB).`

(5%)

c. Define a class named `WorldCupGroup` which implements the `Group` interface provided. This class will be used to determine which teams can progress to the next round of the tournament as well as producing the table of results. The methods in the `WorldCupGroup` class should be implemented based on the following descriptions:

- `public Team getWinner()` - Returns the team object of the team that came first in the group
- `public Team getRunnerUp()` - Returns the team object of the team that came second in the group
- `public String getTable()` - Returns a string containing the current table for the group.

The `getTable` method should have output in the following format. The group name should be on the first line, the column headers should be on the second line, and the teams and their statistics should be on the remaining lines. An example of the output is given here:

```
Group A:
Team                W   D   L   F   A   P
Netherlands         2   1   0   5   1   7
Senegal             2   0   1   5   4   6
Ecuador             1   1   1   4   3   4
Qatar               0   0   3   1   7   0
```

In this the following abbreviations are made:

- **W** - For the number of games the team has won
- **D** - For the number of games the team has drawn
- **L** - For the number of games the team has lost
- **F** - For the number of goals the team has scored
- **A** - For the number of goals that have been scored against this team
- **P** - For the number of points the team has scored

The teams should be primarily ordered from largest to smallest by points. If teams are equal on points, then they should be secondarily ordered by goal difference (F - A). Finally, if all of these are equal then they should be orders based on their names. All of the methods in this class rely on the ability to correctly order the teams in this way. Consider what approach you will use to sort the objects into the correct order.

In order to function in the testing system, you must also add a constructor with the following signature:
`public WorldCupGroup(String groupName, Team a, Team b, Team c, Team d).`

(15%)

d. Define an class named `KnockoutFixture`, which implements the `Fixture` interface provided. This class represents a game that has yet to take place. Override the `toString` method in the class such that when called it returns a String in the following format `"Team Name vs Team Name"`.

In order to function in the testing system, you must also add a constructor with the following signature:
`public KnockoutFixture(Team teamA, Team teamB).`

(5%)

e. Define a class named `Q2`. In this class you should define the following method:

- `public static void readGroups(String filename, Map<String, Team> teams, Map<String, Group> groups)`

This method should read the contents of the file name passed as a parameter. Each line of the file contains the information about one group in the world cup. There is the name of the group followed by the name of the four teams all separated by commas.

Example: `A,Qatar,Ecuador,Senegal,Netherlands`

Based on the data from the file, you must create Team and Group objects and insert them into the maps passed as the second and third parameters. For each team, you should use the team name as the key of the map, and for each group you should use the group name as the key of the map.

(15%)

f. Continuing to work in the class named `Q2`. You should define the following method:

- `public static void readGames(String filename, Map<String, Team> teams, List<Game> games)`

This method should read the contents of the file name passed as a parameter. Each line of the file contains the information about one game in the world cup. There is the name of the one of the teams followed by their score in the game, followed by the name of the other team and their score in the game. All of these values are separated by commas.

Example: `South Korea,0,Portugal,4`

Based on the data from the file, you must create Game objects which are added to each of the teams involved (using the `addGame` methods) and also added to the list which is the last parameter of the method. This will required that you find the original team object that were created in the previous questions, so the map containing all of the teams is also passed as a parameter.

(15%)

g. Continuing to work in the class named `Q2`. You should define the following method:

- `public static void calculateKnockouts(Map<String, Group> groups, List<Fixture> fixtures)`

This method should use the map containing the groups to determine the fixtures for the next part of the tournament. Games are played by the top two teams in each group and are paired between groups A and B, C and D, E and F, and G and H. So The winner of group A will play the runner up of group B and the winner of group B will play the runner up of group A and so on.

(10%)

(Question Total 80%)

(Exam Total 100%)