Beijing-Dublin International College

SEMESTER 1 FINAL EXAMINATION - (2016/2017)

School of Computer Science

COMP2005J Object Oriented Programming Programming Exam

Prof. Pádraig Cunningham
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

Students are permitted to bring any printed notes and textbooks into the examination.

- All questions are based on the same program. Each question adds some new functionality to the code. You should not move to the next question until you have finished the one before it.
- You should only submit working code. If your code does not compile or work correctly you should comment it out.
- To improve your grade you should submit well designed code with light comments and correct indentation.
- The main method of the program should be in a class named `Main`.
- There are bonus marks for providing good JUnit test cases for the classes you have written. This is not required to get 100%, but it will improve your score.
- There are files with some input data for the program on moodle. Download these at the beginning of the exam. Internet access will be turned off after 10 minutes. If you have not downloaded the input files you will have to create them yourself. These files should be put in the project folder in eclipse, not in the src folder with your code.
- The classes you define should have good encapsulation. If you need to access an instance variable you should add a getter or setter method.

Question 1:

The Mixed Doubles Badminton (MDB) Agency provides a service whereby it matches players with other players of the opposite gender to create teams of two players. Every player has a name, a proficiency (integer, 1..10 where 1=beginner and 10=elite player) and the minimum desired proficiency of their team-mate (also integer, 1..10). A sample data set for the women players is provided in `women_players.txt`, and for the men in `men_players.txt`. The format of each line in the file is:

```
<name> <proficiency> <minimum desired proficiency>
```

You may assume there is always the same number of men as women.

a. Implement classes in Java to model the MDB Agency and its players. Write a main method in Main.java that loads the agency with data from `women_players.txt` and `men_players.txt` and then outputs the state of the agency. The expected output for the given input files is:

```
Male Players:
David's proficiency is 10. Seeks partner with proficiency >= 7
John's proficiency is 6. Seeks partner with proficiency >= 5
Paul's proficiency is 4. Seeks partner with proficiency >= 4
Luke's proficiency is 3. Seeks partner with proficiency >= 4
Mark's proficiency is 3. Seeks partner with proficiency >= 6
Female Players:
Jane's proficiency is 10. Seeks partner with proficiency >= 6
Kate's proficiency is 10. Seeks partner with proficiency >= 10
Rachel's proficiency is 4. Seeks partner with proficiency >= 5
Laurel's proficiency is 4. Seeks partner with proficiency >= 3
Ellen's proficiency is 3. Seeks partner with proficiency >= 4
```

(40%)

b. A team is a male and female player that the agency has paired together. The satisfaction of each player on the team is zero (perfect satisfaction) if the proficiency of their team-mate is greater than, or equal to, the minimum proficiency they require in a team-mate. Otherwise it's the gap between these two numbers, which is always negative. The satisfaction of a team is defined as the average satisfaction of its two members.

Add a `createTeams` method to the Agency class that pairs each man and woman together based on proficiency, i.e. the most proficient man with the most proficient woman, and so on. Override the `toString` method in the Agency class, and add code to the main method to create and output the teams created. Expected output for the given input files is:

```
Teams:
(Jane, David) Satisfaction: 0.0
(Kate, John) Satisfaction: -2.0
(Rachel, Paul) Satisfaction: -0.5
(Laurel, Luke) Satisfaction: 0.0
(Ellen, Mark) Satisfaction: -2.0
```

(30%)

c. Add a method named `iterator` to the `Agency` class. This method should return an Iterator for all of the players in the agency. In the main method of the program, use the iterator to output the names of all the elite players (have a proficiency of 10). Expected output for the given input files is:

```
Elite Players:
Jane
Kate
David
```

(10%)

d. Implement a method in the `Agency` class to calculate the fitness of the team selection. The fitness of the team selection is defined as the sum of all of the satisfaction values of each of the teams. **(5%)**

e. Implement a method in the Agency class to improve the fitness of the teams. This method should swap the men between the teams in turn. If a swap does not improve the fitness of the team selection, it should be reversed. The `createTeams` method achieves a fitness of -4.5 on the data given. Successive swapping can improve this to -2.5. Update your main method to output the new, improved set of teams. **(15%)**

(Total 100%)

Submission

Create a single **zip** file containing your entire project. This file should be named using your UCD student number.