QUIZ 1

Issue Date : 15.03.2024 - Friday
Due Date : 17.03.2024 - Sunday (23:00)
Advisor : Berra Nur ÖZTÜRK, R.A. Görkem AKYILDIZ
Programming Language : Java 8u401 (1.8.0_401)

1 Introduction

Classes are the building blocks of the Object Oriented Programming Languages such as Java. Class is a way of abstracting the concepts to the programming language field, these concepts can be either real life entities, imaginary things, or just concepts from other fields. The main focus is, abstracting that concept for giving a use case of it at OOP field, while abstracting, encapsulating is one of the crucial steps that has to be followed. Encapsulation is a concept about separating the questions "What does it do?" and "How does it do?". The "what" part is the field of interest of the users while the "how" part is the field of interest of the developers. It should not be forgotten that the developers mentioned here is the developers that develop this abstraction and encapsulation process while the users can be either just the regular users or other developers.

In this quiz, to warm you up to the concept of OOP with abstraction and encapsulation, a task about implementing a basic dice game is given to you. To play this game, players need two dices, and this game can be played with at least two players. In this game, aim of the players are getting the highest score, for this cause, they have to do some trade-offs according to the rules of the game. The rules are straight-forward, each player throws the dices, if neither of the dices are one, summation of them are added to the score of the player, if just one of them is one, player neither gains nor loses any points, but if both of them are one, player loses all his/her points and the game. At each turn, player can either select to throw the dices or not, if he/she selects to throw the dices, the policy given above applies, otherwise, nothing happens and the game turns to the next player. This game continues until there is only one player left in the game. Note that it is guaranteed that there will be only one winner and the input file ends when there is only one player left.

2 Definition of Input

First line contains the number of the players at the game, second line contains the names of the players separated with comma (,) character, following lines contain two numbers that corresponds to dices separated with dash (-) character, if the line is 0-0, that means player skips his/her turn. Note that game turns are according to the order of the given player names.

3 Definition of Output

Each line contains the latest information about the game. Say that it is Aydın's turn and his score is 65. The possible scenarios with the corresponding outputs are as follows:

- If he threw the dice as 6-6: Aydın threw 6-6 and Aydın's score is 77.
- If he threw dice as 1-5: Aydın threw 1-5 and Aydın's score is 65.
- If he threw dice as 1-1: Aydın threw 1-1. Game over Aydın!
- If he skips his turn (0-0): Aydın skipped the turn and Aydın's score is 65.
- If he is the last player standing: Aydın is the winner of the game with the score of 65. Congratulations Aydın!

4 Restrictions

- Your code must be able to execute on our department's developer server (dev.cs.hacettepe.edu.tr).
- You must obey given submit hierarchy and get score (1 point) from the submit system.
- **You must benefit from OOP, abstraction, and encapsulation; any solution that does not contain even one of these concepts will not be accepted, moreover, partial point deductions may exist due to partially erroneous usage of these concepts.**
- Your code must be clean, do not forget that main method is just a driver method that means it is just for making your code fragments run, not for using them as a main container, create classes and methods in necessary situations but use them as required. Moreover, use the at least first two (the ones that are in bold) of four pillars of Object-Oriented Programming (**Abstraction**, **Encapsulation**, Inheritance, Polymorphism) if there is such a need, remember that your code must satisfy Object-Oriented Programming Principles.
- You are encouraged to use lambda expressions which are introduced with **Java 8**.
- You must use JavaDoc commenting style for this project, and you must give brief information about the challenging parts of your code, do not over comment as it is against clean code approach. Design your comments so that if someone wants to read your code they should be able to easily understand what is going on. You can check here to access Oracle's own guide about JavaDoc Sytle.

- You can benefit from Internet sources for inspiration but do not use any code that does not belong to you.
- You can discuss high-level (design) problems with your friends but do not share any code or implementation with anybody.
- Do not miss the submission deadline.
- Source code readability is a great of importance. Thus, write READABLE SOURCE CODE, comments, and clear MAIN function. This expectation will be graded as "clean code".
- Use UNDERSTANDABLE names for your variables, classes, and functions regardless of the length. The names of classes, attributes and methods must obey to the Java naming convention. This expectation will be graded as "coding standards".
- You can ask your questions through course's Piazza group, and you are supposed to be aware of everything discussed in the Piazza group. General discussion of the problem is allowed, but **DO NOT SHARE** answers, algorithms, source codes and reports.
- All quizzes must be original, individual work. Duplicate or very similar quizzes are both going to be considered as cheating.

5 Execution and Test

Your code must be executed under **Java 8u401 (1.8.0_401)** at **dev.cs.hacettepe.edu.tr**. If your code does not run at developer server during the testing stage, then you will be graded as 0 for code part even if it works on your own machine. Sample run command is as follows:

- Either `javac DiceGame.java` or `javac *.java` command for compilation.
- `java DiceGame input.txt output.txt` command for run.

6 Grading

Task	Point
Correct Output	100*
Total	100

*** Even though producing correct output seems like enough to get full credit, you must obey to the given rules in the PDF (for example using concept of OOP, abstraction, and encapsulation, commenting etc.) otherwise you may face with some point deductions which may result with a grade that is as low as zero.**

7 Submit Format

File hierarchy must be zipped before submitted (Not .rar, only not compressed .zip files because the system just supports .zip files).

```
- b<StudentID>.zip
    - <src>
       - DiceGame.java
       - *.java (Optional)
```