Due Date: 03.11.2025 23:55

CENG211 – Programming Fundamentals
Homework #1

In this homework, you are expected to implement a "The E-Sports Tournament Challenge" in Java. You should fulfill the concepts of:

- Defining Classes
- CSV file I/O
- Arrays
- 2-dimensional Arrays
- Constructors, Getters & Setters

A major E-Sports organization is preparing for a large-scale tournament. Hundreds of gamers from different backgrounds will compete, and each gamer will participate in 15 matches during the season. To ensure fair competition and proper scoring, the organization needs a system to simulate matches, calculate points, and assign medals to the gamers at the end of the season.

- In each **match**, exactly **3 different games** are played. Each game has a 1-D array of 3 games, the number of rounds for each game, and the calculated points.
- The games of each match are randomly determined according to the game ID.
- Multiple matches could contain the same game.
- Each game consists of a random number of **rounds** between **1 and 10, randomly determined for each game in the match**.
- Every game type has a **base point per round** defined in 'games.csv' (for example: playing one round of CS2 gives 18 points, while one round of Fortnite gives 9 points).

At the end of a match, the gamer's raw points are computed as the total of all rounds multiplied by each game's base point.

To encourage high-performance matches, the tournament also gives bonus points depending on the total skill points calculated with the raw score of the match.

Finally, after all 1500 matches are played, each gamer's total season points are calculated, and gamers are awarded medals based on their performance.

Your task is to implement this tournament management system.

System Description

Input:
The system will read two CSV files:
1. games.csv: Each row contains the ID, the name of the game, and its base point per round.

2. gamers.csv: Each row contains the ID, nickname, real name, phone number, and experience years of a gamer.

Match Points Calculation
1. Raw Points:
   The raw points of a match are calculated as the sum of (rounds × basePointPerRound) for each game.

$$\text{rawPoints} = \Sigma\ (\text{rounds}[i] \times \text{basePointPerRound}[i]) \quad \text{for } i = 1..3$$

2. Skill Points:
   The gamer's experience years are applied as a multiplier:

   $$skillPoints = floor(rawPoints \times (1 + min(experienceYears,10) \times 0.02))$$

   * "If experienceYears > 10, treat it as 10 for this calculation."

   *Hint:*Skill Points are calculated once per match

3. Bonus Points:
   Awarded depending on the raw points of the match:

Skill Points	Bonus Points
$0 \leq rawPoints \leq 199$	+10
$200 \leq rawPoints \leq 399$	+25
$400 \leq rawPoints \leq 599$	+50
$rawPoints \geq 600$	+100

4. Match Points:

   $$matchPoints = skillPoints + bonusPoints$$

Medal Assignment

At the end of the tournament, each gamer's total season score is the sum of all match points from 15 matches. Based on this total, medals are assigned:

Total Score	Medal
$totalScore \geq 2000$	GOLD
$1200 \leq totalScore \leq 1999$	SILVER
$700 \leq totalScore \leq 1199$	BRONZE
$totalScore \leq 700$	NONE

Season totals (per gamer)

- Total Points **= sum of** matchPoints **over that gamer's** 15 matches**.**
- Average Per Match **= Total Points / 15.0 (you may format to 2 decimals).**
- Medal assignment **(based on** Total Points**):**
  - GOLD**: ≥ 2000**
  - SILVER**: 1200–1999**
  - BRONZE**: 700–1199**
  - NONE**: < 700**

Type & rounding rules

- rawPoints, skillPoints, bonusPoints, matchPoints, Total Points are **integers**.
- The **only** rounding is the **floor** in skillPoints.
- Average Per Match may be printed with 2 decimal places.

Tie-breaking

- If multiple results are tied (same score/points), **printing any one** is acceptable.

Minimal example

- Suppose a match has games with base points 12, 15, 18 and rounds 6, 4, 3.
  - rawPoints = 6×12 + 4×15 + 3×18 = 72 + 60 + 54 = 186
  - Gamer has ExperienceYears = 5 → skillMultiplier = 1 + 5×0.02 = 1.10
  - skillPoints = floor(186 × 1.10) = floor(204.6) = 204
  - Bonus by table: 200–399 → +25
  - matchPoints = 204 + 25 = 229

In this homework, you are expected to implement the necessary classes to load the data from the given CSV files and create the desired queries. You are expected to implement classes for Game, Gamer, Match, MatchManagement, PointsBoard, Query, EsportsManagementApp (the class with main method), and other helper classes (e.g., FileIO) with the information given below:

CSV files to load:
games.csv: ID, GameName, BasePointPerRound
gamers.csv: ID, Nickname, Name, Phone, ExperienceYears

Game:
- ID
- Game Name
- Base Point Per Round

Gamer:
- ID
- Nickname
- Name
- Phone Number
- Experience Years

Match:
- ID
- 1-D Array of 3 Games
- Match Points
- Necessary information to calculate match points
  - *Hint:* Remember that each game may differ in the number of rounds played.

PointsBoard:
- Gamer
  - One-dimensional array that holds Gamer objects and computes, for each gamer, after all matches are generated and assigned: Total Points, Average Points Per Match, and Medal

MatchManagement:
- Match
  - Two-dimensional array that holds Match objects for each gamer, with exactly 15 matches per gamer.
  - Ex: For the 3rd gamer's 7th match, it is [3][7].

Implement necessary methods to respond to the following queries in Query class:
1- The highest-scoring match (by Match Points).
2- In the lowest-scoring match, the most contributing game (the game maximizing rounds[i] × basePointPerRound[i]) and its contribution value.
3- The match with the lowest bonus points.
4- The highest-scoring gamer. (Please, include his/her Nickname, Name, Total Points, Average Points Per Match, and Medal.)
5- The total tournament points that is obtained from 1500 matches by summing Match Points of each match.
6- The medal distribution after the season. (Please, report the number of GOLD, SILVER, BRONZE, and NONE gamers and percentages.)

If the result of any of the queries is more than one, please display one result. Since almost all data are randomly determined, the results of your projects will differ each time you execute your code and, also from each other. This is completely normal.

Sample Outputs for Queries:

1. Highest-Scoring Match

Highest-Scoring Match:
Match ID: 872
Games: [Valorant, CS2, Overwatch]
Rounds: [7, 3, 9]
Raw Points: 312
Skill Points: 342
Bonus Points: 50
Match Points: 392

2.Lowest-Scoring Match & Most Contributing Game

Lowest-Scoring Match:
Match ID: 145
Games: [League of Legends, Fortnite, CS2]
Rounds: [2, 1, 3]
Raw Points: 87
Skill Points: 87
Bonus Points: 10
Match Points: 97

Most Contributing Game in this Match:
Game: CS2
Contribution: 3 rounds × 18 points = 54

3. Match with the Lowest Bonus Points

Match with Lowest Bonus Points:
Match ID: 412
Games: [Overwatch, Valorant, Fortnite]
Skill Points: 142
Bonus Points: 10
Match Points: 152

4. Highest-Scoring Gamer

Highest-Scoring Gamer:
Nickname: VoltRider
Name: Kerem Aslan
Total Points: 2678
Average Per Match: 178.53
Medal: GOLD

5. Total Tournament Points

Total Tournament Points across 1500 matches: 198,420

6. Medal Distribution

Medal Distribution:
GOLD: 12 gamers (12.0%)
SILVER: 28 gamers (28.0%)
BRONZE: 35 gamers (35.0%)
NONE: 25 gamers (25.0%)

Important Notes:

1. Do NOT request inputs in your app. Printing the results of the queries will be enough. You should print names of the results instead of printing IDs or indices.

2. You are NOT allowed to use List / ArrayList interfaces in this homework.

3. You can use standard java.io packages to read files. Do NOT use other 3rd party libraries.

4. You should use **relative** paths (e.g. Files/sample.csv) instead of **absolute** paths (e.g. C:\\user\\eclipse-workspace\\MyProject\\Files\\sample.csv). Please be sure of it, otherwise there will be **no output** of your application and you certainly will **lose points**.

5. To support **Turkish characters,** you may need to change your project's text file encoding to UTF8: Right click on your project (in package explorer) → Properties → Text file encoding → Other → UTF8 → Apply.

6. You are expected to write clean, readable, and tester-friendly code. Please try to maximize reusability and prevent from redundancy in your methods.

References

Assignment Rules:

1. In this lecture's homework, there are no cheating allowed. If any cheating has been detected, they will be graded as 0 and there will be no further discussion on this.
2. You are expected to submit your homework in groups. Therefore, only one of you will be sufficient to submit your homework.
3. Make sure you export your homework as an Eclipse project. You can use other IDEs as well, however, you must test if it **can be executed** in Eclipse.
4. Submit your homework through Cloud-LMS.
5. Your exported Java Project should have the following naming format with your assigned group ID (which will be announced on MS Teams) as the given below:

   G05_CENG211_HW1

   Also the zip folder that your project in should have the same name

   G05_CENG211_HW1.zip
6. Please beware that if you do not follow the assignment rules for exporting and naming conventions, you will lose points.
7. Please be informed that your submissions may be anonymously used in software testing and maintenance research studies. Your names and student IDs will be replaced with non-identifying strings. If you do not want your submissions to be used in research studies, please inform the instructor (Dr. Tuglular) via e-mail.