Vanier College
Computer Science Department

Programming 2

Assignment 3

Due Date: By 11:59pm Friday April 25, 2025
Evaluation: 5% of final mark (see marking rubric at the end of handout)
Late Submission: none accepted
Demo: Students may need to demo their assignment if time allows. If a demo is required, submissions without a demo will be considered invalid. If the correct solution has been submitted but the student cannot properly present it, points may be deducted, or a zero may be assigned.

General Guidelines When Writing Programs:

Include the following comments at the top of your source codes :

```
// -----------------------------------------------------
// Assignment (include number)
// Written by: (include your name and student id)
// For "Programming 2" Section (include number) - Winter 2025
// -----------------------------------------------------
```

- In a comment, give a general explanation of what your program does. As the programming questions get more complex, the explanations will get lengthier.
- Include comments in your program describing the main steps in your program.
- Display a welcome message.
- Display clear prompts for users when you are expecting the user to enter data from the keyboard.
- All output should be displayed with clear messages and in an easy to read format.
- End your program with a closing message so that the user knows that the program has terminated.

What to Submit:

- Make **one ZIP** file that includes all of the **.java files.**
- Submit the ZIP file via Omnivox.
    - Do not use the RAR (or some other) format!

Assignments not submitted to the correct location or not in the requested format will not be graded.

Q1: Design a Fitness Tracking System to manage and analyze a user's workout data stored in a file. The system must use interfaces, support reading/saving from/to a file, and allow user interaction through a menu-driven console application.

Requirements:

Interface FitnessOperations:

To support the following operations:

-displayAllSessions();
- Displays all workout sessions stored in the system, ideally in a user-friendly format.
- Typically sorted by date, leveraging the natural ordering defined in the WorkoutSession class.

-displayByDurationThreshold(int threshold);
- Filters and displays all sessions that lasted longer than a given duration (in minutes).
- Useful for analyzing more intense workouts.

-addWorkoutSession(WorkoutSession session);
- Adds a new workout session to the current collection.
- Helps users dynamically log their fitness activity.

- removeWorkoutByDate(String date);
- Removes a workout session based on the date (assumed to be unique per session).
- Returns true if removal was successful, false otherwise.

- findLongestWorkout();
- Searches through all stored sessions to find the one with the maximum duration.
- Can be used for setting personal best records.

- saveToFile(String filename);
- Writes the current list of workout sessions to a text file.
- Ensures persistence of user data between sessions.

Class WorkoutSession:

The WorkoutSession class represents an individual workout entry in the fitness tracking system. It models key attributes of a workout and provides comparison logic to support sorting operations.

-Implements Comparable<WorkoutSession>:
-The class overrides the compareTo() method to allow sorting of workout sessions by date. Since dates are stored as String in the format "YYYY-MM-DD", alphabetical comparison works effectively for ascending chronological order.

Attributes:

```
String date (e.g., "2025-04-11")
String activityType (e.g., "Running", "Swimming")
int duration (minutes)
int caloriesBurned
```

Class FitnessManager:

The FitnessManager class is responsible for managing the entire list of workout sessions and implementing all the functionalities declared in the FitnessOperations interface. It acts as the main logic controller of the system.

- Implements FitnessOperations:
-Manages a list of workout sessions:
-Reads from and writes to a file:

Note: Your program should handle different exception types including invalid input or file errors gracefully (e.g., wrong date format, missing file).

File Format: workouts.txt

Example:

```
2025-04-01,Running,45,300
2025-04-05,Swimming,30,250
```

Sample Menu:

```
Health Tracker Menu:
1. Display All Workout Sessions
2. Display Workouts Longer than X Minutes
3. Add New Workout Session
4. Remove Workout by Date
5. Show Longest Workout
6. Save and Exit
```

Sample Input & Output:

```
Enter your choice: 2
Enter minimum duration: 40

Workouts longer than 40 minutes:
- 2025-04-01 | Running | 45 min | 300 cal
```

Q2: Triangle Path Puzzle

Write a recursive program to solve a triangle path puzzle, where a user must move from the top of a triangle to the bottom by choosing adjacent numbers in the row below to maximize total sum.

Example:

Sample Input:

```
7
3 8
8 1 0
2 7 4 4
4 5 2 6 5
```

Each row represents a level in the triangle.

Sample Output:

```
Max Path Sum: 30
Path Taken: [7, 3, 8, 7, 5]
```

Evaluation Criteria for Assignment3/Demo (150 points)

Source Code		
Comments (for both questions) (20 pts.)		
Description of the program (authors, date, purpose)	5	pt.
Description of variables and methods	10	pt.
Description of the algorithm	5	pt.
Programming Style (for both questions) (10 pts.)		
Use of significant names for identifiers	3	pt.
Indentation and readability	5	pt.
Welcome Banner/Closing message	2	pt.
Question 1 (85pts.)		
Interface Implementation	10	pts.
WorkoutSession Class	10	pts.
File Handling (Read/Write)	10	pts.
Functionality and Menu	15	pts.
Sorting & Filtering Logic	10	pts.
Exception Handling & Validation	10	pts.
Format/clarity/completeness/accuracy of output	10	pts.
Proper use of required concepts	10	pts.
Question 2 (35pts.)		
Format/clarity/completeness/accuracy of output	20	pts.
Proper use of required concepts	15	pts.
TOTAL	**150**	**pts.**