Name: Nguyen Phuong Le

ID: 104178943

**Week 10**

**Task 10:**

package week10;

import java.io.\*;

import java.util.\*;

class Trainer {

    private String Name;

    private String Addr;

    private String level;

    public Trainer(String Name, String Addr, String level) {

        this.Name = Name;

        this.Addr = Addr;

        this.level = level;

    }

    public String getName() { return Name; }

    public String getAddress() { return Addr; }

    public String getTeachingLevel() { return level; }

    public void setTeachingLevel(String level1) { this.level = level1; }

    @Override

    public String toString() {

        return "Name: " + this.Name + ", Address: " + this.Addr + ", Teaching level: " + this.level+".";

    }

}

class Student {

    private String Name;

    private String Addr;

    private int Age;

    private String Swim\_Level;

    public Student(String Name, String Addr, int Age, String Swim\_Level) {

        this.Name = Name;

        this.Addr = Addr;

        this.Age = Age;

        this.Swim\_Level = Swim\_Level;

    }

    public String getName() { return Name; }

    public String getAddress() { return Addr; }

    public int getAge() { return Age; }

    public String getLevel() { return Swim\_Level; }

    public void setLevel(String Swim\_Level1) { this.Swim\_Level = Swim\_Level1; }

    @Override

    public String toString() {

        return "Name: " + this.Name + ", Address: " + this.Addr + ", Age: " + this.Age + ", Swim Level: " + this.Swim\_Level;

    }

}

class Intermediate\_level  extends Student {

    private double freestyle\_50\_meters;

    private double backstroke\_25\_meters;

    public Intermediate\_level(String Name, String Addr, int Age, double freestyle\_50\_meters, double backstroke\_25\_meters) {

        super(Name, Addr, Age, "Intermediate");

        this.freestyle\_50\_meters = freestyle\_50\_meters;

        this.backstroke\_25\_meters = backstroke\_25\_meters;

    }

    public double getFreestyle\_50meters() { return freestyle\_50\_meters; }

    public double getBackstroke\_25meters() { return backstroke\_25\_meters; }

    @Override

    public String toString() {

        return super.toString() +

               "\n- 50m freestyle: " + this.freestyle\_50\_meters + " seconds" +

               "\n- 25m backstroke: " + this.backstroke\_25\_meters + " seconds";

    }

}

class Advanced\_level extends Student {

    private double freestyle\_100\_meters;

    private double backstroke\_50\_meters;

    private double breaststroke\_25\_meters;

    public Advanced\_level(String Name, String Addr, int Age, double freestyle\_100\_meters, double backstroke50m, double breaststroke25m) {

        super(Name, Addr, Age, "Advanced");

        this.freestyle\_100\_meters = freestyle\_100\_meters;

        this.backstroke\_50\_meters = backstroke50m;

        this.breaststroke\_25\_meters = breaststroke\_25\_meters;

    }

    public double getFreestyle\_100meters() { return freestyle\_100\_meters; }

    public double getBackstroke\_50meters() { return backstroke\_50\_meters; }

    public double getBreaststroke\_25meters() { return breaststroke\_25\_meters; }

    @Override

    public String toString() {

        return super.toString() +

               "\n- 100m freestyle: " + this.freestyle\_100\_meters + " seconds" +

               "\n- 50m backstroke: " + this.backstroke\_50\_meters + " seconds" +

               "\n- 25m breaststroke: " + this.breaststroke\_25\_meters + " seconds";

    }

}

public class task10 {

    private static ArrayList<Trainer> trainers = new ArrayList<>();

    private static ArrayList<Student> students = new ArrayList<>();

    static int trainerCount = 0;

    static int studentCount = 0;

    public static void printmenu() {

        System.out.println("\n----- Swim Center Management System -----");

        System.out.println("1. Add Trainer\n"

                + "2. Add Student\n"

                + "3. Change Student Level\n"

                + "4. Change Trainer Level\n"

                + "5. List All Trainers\n"

                + "6. List All Students\n"

                + "7. Save All.");

        System.out.print("Enter your choice: ");

    }

    public static void importTrainersData(File file) throws IOException {

        if (!file.exists()) {

            System.out.println("No existing trainer records found. Initializing a new database");

            file.createNewFile();

        } else {

            try (Scanner fileReader = new Scanner(new FileReader(file))) {

                while (fileReader.hasNextLine()) {

                    String TC1 = fileReader.nextLine();

                    String[] fields = TC1.split("\t");

                    if (fields.length != 3)

                        System.out.println("Encountered an incorrectly formatted entry: " + TC1);

                    else {

                        trainers.add(new Trainer(fields[0], fields[1], fields[2]));

                        trainerCount++;

                    }

                }

            }

        }

    }

    public static void importStudentsData(File file) throws IOException {

        if (!file.exists()) {

            System.out.println("Student database not detected. Creating a new record system");

            file.createNewFile();

        } else {

            try (Scanner fileReader = new Scanner(new FileReader(file))) {

                while (fileReader.hasNextLine()) {

                    String line = fileReader.nextLine();

                    String[] fields = line.split("\t");

                    if (fields.length < 4 || fields.length > 7)

                        System.out.println("Data entry error: Incorrect field count in line: " + line);

                    else {

                        String name = fields[0];

                        String address = fields[1];

                        int age = Integer.parseInt(fields[2]);

                        String level = fields[3];

                        if (level.equalsIgnoreCase("Intermediate")) {

                            double freestyle\_50\_meters = Double.parseDouble(fields[4]);

                            double backstroke\_25\_meters = Double.parseDouble(fields[5]);

                            students.add(new Intermediate\_level(name, address, age, freestyle\_50\_meters, backstroke\_25\_meters));

                        } else if (level.equalsIgnoreCase("Advanced")) {

                            double freestyle\_100\_meters = Double.parseDouble(fields[4]);

                            double backstroke\_50\_meters = Double.parseDouble(fields[5]);

                            double breaststroke\_25\_meters = Double.parseDouble(fields[6]);

                            students.add(new Advanced\_level(name, address, age, freestyle\_100\_meters, backstroke\_50\_meters, breaststroke\_25\_meters));

                        } else {

                            students.add(new Student(name, address, age, level));

                        }

                        studentCount++;

                    }

                }

            }

        }

    }

    public static void exportData(File trainersFile, File studentsFile) throws IOException {

        // Save trainers data

        try (PrintWriter trainerWriter = new PrintWriter(new FileWriter(trainersFile))) {

            for (Trainer trainer : trainers) {

                trainerWriter.println(trainer.getName() + "\t" + trainer.getAddress() + "\t" + trainer.getTeachingLevel());

            }

            System.out.println("Trainer information successfully archived.");

        } catch (IOException e) {

            System.out.println("An error occurred while saving trainer data: " + e.getMessage());

        }

        // Save students data

        try (PrintWriter studentWriter = new PrintWriter(new FileWriter(studentsFile))) {

            for (Student student : students) {

                studentWriter.print(student.getName() + "\t" + student.getAddress() + "\t" + student.getAge() + "\t" + student.getLevel());

                if (student instanceof Intermediate\_level) {

                    Intermediate\_level intermediateStudent = (Intermediate\_level) student;

                    studentWriter.print("\t" + intermediateStudent.getFreestyle\_50meters() + "\t" + intermediateStudent.getBackstroke\_25meters());

                } else if (student instanceof Advanced\_level) {

                    Advanced\_level advancedStudent = (Advanced\_level) student;

                    studentWriter.print("\t" + advancedStudent.getFreestyle\_100meters() + "\t" + advancedStudent.getBackstroke\_50meters() + "\t" + advancedStudent.getBreaststroke\_25meters());

                }

                studentWriter.println();

            }

            System.out.println("Student records successfully updated and stored.");

        } catch (IOException e) {

            System.out.println("Unable to save student data due to an error: " + e.getMessage());

        }

    }

    public static void main(String[] args) throws IOException {

        Scanner scanner = new Scanner(System.in);

        File trainersFile = new File("trainers.txt");

        File studentsFile = new File("students.txt");

        int choice;

        importTrainersData(trainersFile);

        importStudentsData(studentsFile);

        do {

            printmenu();

            choice = scanner.nextInt();

            switch (choice) {

            case 1:

                registerTrainer(scanner); break;

            case 2:

                enrollStudent(scanner); break;

            case 3:

                updateStudentLevel(scanner); break;

            case 4:

                updateTrainerLevel(scanner); break;

            case 5:

            case 6:

                displayRecords(choice); break;

            case 7:

                System.out.println("Saving all records...");

                exportData(trainersFile, studentsFile); break;

            default:

                System.out.println("Invalid choice. Please try again."); break;

            }

        } while (choice != 7);

    }

    public static void registerTrainer(Scanner scanner) {

        System.out.print("\nTrainer Name: ");

        scanner.nextLine();

        String name = scanner.nextLine();

        System.out.print("Trainer Address: ");

        String address = scanner.nextLine();

        System.out.print("Trainer teaching level (Beginners, Intermediate, Advanced): ");

        String level = scanner.nextLine();

        trainers.add(new Trainer(name, address, level));

        System.out.println("\nTrainer registration completed successfully.");

        trainerCount++;

    }

    public static void enrollStudent(Scanner scanner) {

        System.out.println();

        System.out.print("Student's Name: ");

        scanner.nextLine();

        String name = scanner.nextLine();

        System.out.print("Student's Address: ");

        String address = scanner.nextLine();

        System.out.print("Student's Age: ");

        int age = scanner.nextInt();

        scanner.nextLine();

        System.out.print("Student swim level (Beginners, Intermediate, Advanced): ");

        String level = scanner.nextLine();

        if (level.equalsIgnoreCase("Intermediate")) {

            System.out.print("Enter timing for 50 meters freestyle (seconds): ");

            double freestyle\_50\_meters = scanner.nextDouble();

            System.out.print("Enter timing for 25 meters backstroke (seconds): ");

            double backstroke\_25\_meters = scanner.nextDouble();

            scanner.nextLine();

            students.add(new Intermediate\_level(name, address, age, freestyle\_50\_meters, backstroke\_25\_meters));

        } else if (level.equalsIgnoreCase("Advanced")) {

            System.out.print("Enter timing for 100 meters freestyle (seconds): ");

            double freestyle\_100\_meters = scanner.nextDouble();

            System.out.print("Enter timing for 50 meters backstroke (seconds): ");

            double backstroke\_50\_meters = scanner.nextDouble();

            System.out.print("Enter timing for 25 meters breaststroke (seconds): ");

            double breaststroke\_25\_meters = scanner.nextDouble();

            scanner.nextLine();

            students.add(new Advanced\_level(name, address, age, freestyle\_100\_meters, backstroke\_50\_meters, breaststroke\_25\_meters));

        } else

            students.add(new Student(name, address, age, level));

        System.out.println("\nStudent enrollment process completed.");

        System.out.println(students.get(studentCount).toString());

        assignTrainer(students.get(studentCount));

        studentCount++;

    }

    public static void assignTrainer(Student student) {

        for (int i = 0; i < trainerCount; i++) {

            if (trainers.get(i).getTeachingLevel().equalsIgnoreCase(student.getLevel())) {

                System.out.println("Assigned Trainer: " + trainers.get(i).getName() + " for student: " + student.getName());

                return;

            }

        }

        System.out.println("No trainer available for this level.");

    }

    public static void updateStudentLevel(Scanner scanner) {

        System.out.print("Enter student's name: ");

        scanner.nextLine();

        String name = scanner.nextLine();

        int i;

        for (i = 0; i < studentCount; i++) {

            if (students.get(i).getName().equals(name)) {

                System.out.print("Enter new swim level (Beginners, Intermediate, Advanced): ");

                String level = scanner.nextLine();

                students.get(i).setLevel(level);

                System.out.println("Student's swim proficiency level has been updated.");

                System.out.println(students.get(i).toString());

                break;

            }

        }

        if (i >= studentCount)

            System.out.println("No student found.");

    }

    public static void updateTrainerLevel(Scanner scanner) {

        System.out.print("\nEnter trainer's name: ");

        scanner.nextLine();

        String name = scanner.nextLine();

        int i;

        for (i = 0; i < trainerCount; i++) {

            if (trainers.get(i).getName().equals(name)) {

                System.out.print("Enter trainer's new teaching level (Beginners, Intermediate, Advanced): ");

                String level = scanner.nextLine();

                trainers.get(i).setTeachingLevel(level);

                System.out.println("Trainer's expertise level has been successfully modified.");

                System.out.println(trainers.get(i).toString());

                break;

            }

        }

        if (i >= trainerCount)

            System.out.println("No trainer found.");

    }

    public static void displayRecords(int option) {

        if (option == 5) {

            System.out.println("\nAll trainers:");

            for (int i = 0; i < trainerCount; i++)

                System.out.println(trainers.get(i).toString());

        } else {

            System.out.println("\nAll students:");

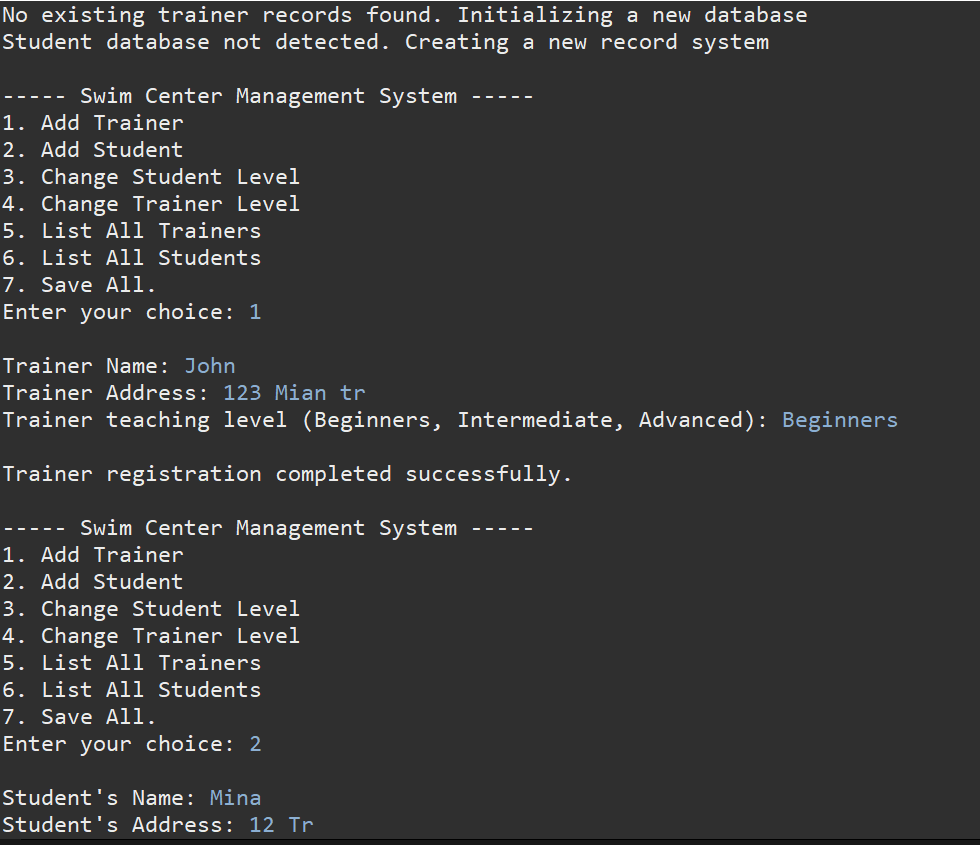
            for (int i = 0; i < studentCount; i++)

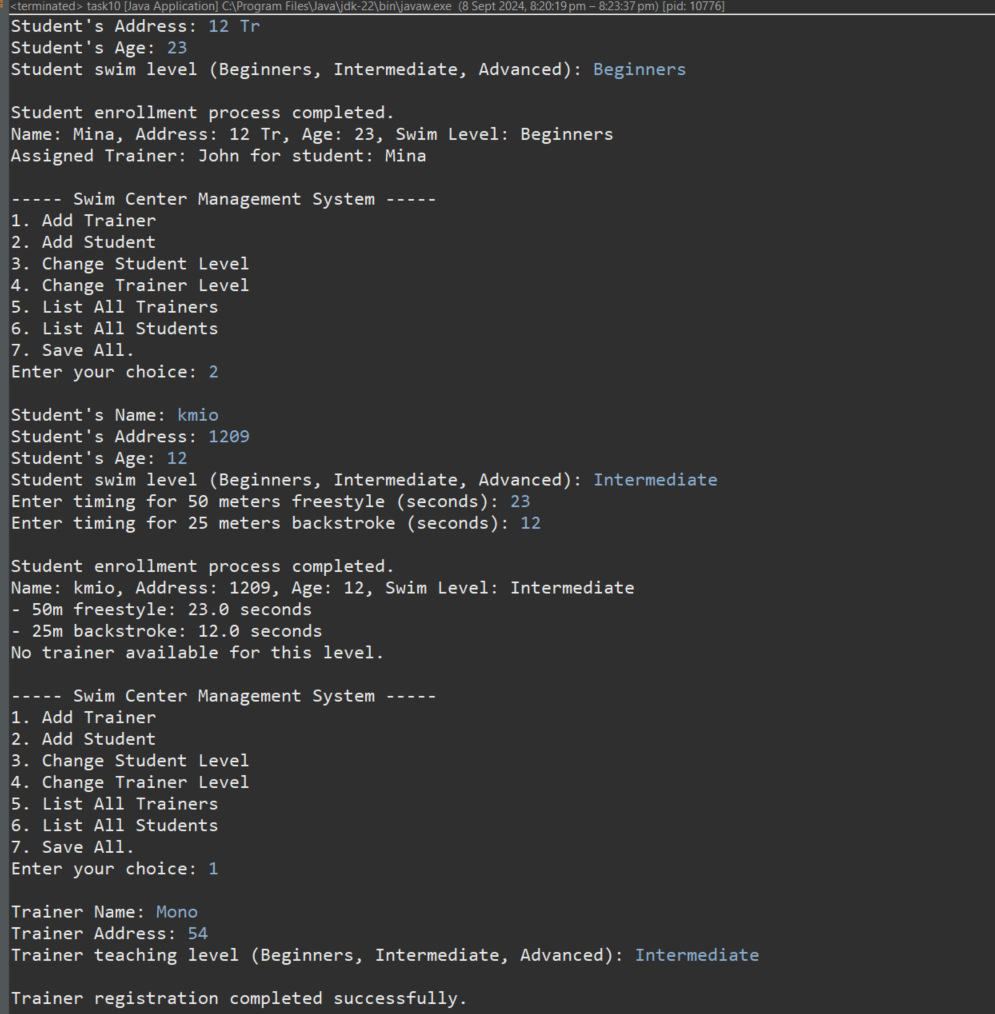
                System.out.println(students.get(i).toString());

        }

    }

}

****

****

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**