Name: Kamran Gul

CMS: 023-25-0161

Section: C

OOP Lab: 04

Task 1

Student Class - Basic Structure

Create a Student class with the following requirements:

Requirements:

1. Private attributes: name (String), studentId (int), gpa (double)
2. Default constructor that initializes name to "Unknown", studentId to 0, gpa to 0.0
3. Parameterized constructor that takes name, studentId, and gpa
4. Getter methods: getName(), getStudentId(), getGpa()
5. Setter methods: setName(String name), setStudentId(int studentId), setGpa(double gpa)
6. A method displayInfo() that prints all student information
7. Use this keyword appropriately in constructors and setters

Create a Main class to test:

```
public class Main {
   public static void main(String[] args) {
      Student s1 = new Student();
      Student s2 = new Student("John Doe", 12345, 3.75);
      s1.displayInfo();
      s2.displayInfo();
      s1.setName("Jane Smith");
      s1.setStudentId(12346);
      s1.setGpa(3.90);
      s1.displayInfo();
   }
}
```

Solution

```
class Student{
      // Task 1
      private String name;
      private int studentId;
      private double gpa;
```

```
      public Student(){
            this.name = "Unknown";
            this.studentId = 0;
            this.gpa = 0.0;
      }
      public Student(String name,int studentId,double gpa){
            this.name = name;
            this.studentId = studentId;
            this.gpa = gpa;
      }
      String getName(){
            return this.name;
      }
      int getStudentId(){
            return this.studentId;
      }
      double getGpa(){
            return this.gpa;
      }
      void setName(String name){
            this.name= name;
      }
      void setStudentId(int studentId){
            this.studentId= studentId;
      }
      void setGpa(double gpa){
            this.gpa = gpa;
      }
      void displayInfo(){
            System.out.println("Student Information:");
            System.out.println("Name: " + name);
            System.out.println("Student Id: " + studentId);
            System.out.println("GPA: " + gpa +"\n");
      }

}
class Main {
    public static void main(String[] args) {
      //Task 1
         Student s1 = new Student();
         Student s2 = new Student("John Doe", 12345, 3.75);

         s1.displayInfo();
         s2.displayInfo();

         s1.setName("Jane Smith");
         s1.setStudentId(12346);
         s1.setGpa(3.90);
         s1.displayInfo();

    }

    }
```

Output

```
Student Information:
Name: Unknown
Student Id: 0
GPA: 0.0

Student Information:
Name: kamran
Student Id: 1
GPA: 3.2

Student Information:
Name: Jane Smith
Student Id: 12346
GPA: 3.9
```

Task 2

Add Control Flow Methods

Modify your Student class from Task 1 by adding the following methods:

Requirements:

1. Method calculateGrade() that returns a char based on GPA using if/else:
   - 'A' if gpa >= 3.7
   - 'B' if gpa >= 3.0
   - 'C' if gpa >= 2.0
   - 'D' if gpa >= 1.0
   - 'F' otherwise
1. Method getGradeLetter() that returns the grade letter using switch statement
2. Method isHonorStudent() that returns true if GPA >= 3.5 (use ternary operator)
3. Method updateGPA(double newGPA) that updates the GPA and validates it's between 0.0 and 4.0 using if/else
4. Method getStatus() that returns "Excellent", "Good", "Average", or "Needs Improvement" based on GPA using if/else

Update your Main class:

```
public class Main {
   public static void main(String[] args) {
      Student s1 = new Student("John Doe", 12345, 3.75);
      Student s2 = new Student("Jane Smith", 12346, 2.50);
      System.out.println("Grade: " + s1.calculateGrade());
      System.out.println("Is Honor Student: " + s1.isHonorStudent());
      System.out.println("Status: " + s1.getStatus());

      s2.updateGPA(3.90);
      s2.displayInfo();
   }
}
```

Solution

```
class Student{


	private String name;
	private int studentId;
	private double gpa;

	public Student(){
		this.name = "Unknown";
		this.studentId = 0;
		this.gpa = 0.0;
	}
	public Student(String name,int studentId,double gpa){
		this.name = name;
		this.studentId = studentId;
		this.gpa = gpa;
	}
	void displayInfo(){
		System.out.println("Student Information:");
		System.out.println("Name: " + name);
		System.out.println("Student Id: " + studentId);
		System.out.println("GPA: " + gpa +"\n");
	}

	// Task 2
	char calculateGrade(){
		if(this.gpa>=3.7) return 'A';
		else if(this.gpa>=3) return 'B';
		else if(this.gpa>=2) return 'C';
		else if(this.gpa>=1) return 'D';
		else return 'F';
	}
	boolean isHonorStudent(){
		 return (this.gpa>=3.5) ? true : false;
	}
	void updateGPA(double newGPA){
		if(newGPA>=0.0 && newGPA<=4.0){
			this.gpa = newGPA;
		}
	}
	String getStatus(){
		if(this.gpa>=3.7)
			return "Excellent";
		else if(this.gpa>=3)
			return "Good";
		else if(this.gpa>=2)
			return "Average";
		else if(this.gpa>=1)
			return "Need improment";

		return "Fail";

}
}
class Main {
    public static void main(String[] args) {
```

```
        Student s1 = new Student("John Doe", 12345, 3.75);
        Student s2 = new Student("Jane Smith", 12346, 2.50);

        System.out.println("Grade: " + s1.calculateGrade());
        System.out.println("Is Honor Student: " + s1.isHonorStudent());
        System.out.println("Status: " + s1.getStatus());

        s2.updateGPA(3.90);
        s2.displayInfo();
    }
}
```

Output

```
Grade: A
Is Honor Student: true
Status: Excellent
Student Information:
Name: Jane Smith
Student Id: 12346
GPA: 3.9
```

Task 3

Task 3: Add String Handling and Array Methods

Modify your Student class from Task 2 by adding the following methods:

Requirements:

1. Method formatName() that returns the name in "Last, First" format (assume name is "First Last")
   - Use string methods: substring(), indexOf(), toUpperCase(), toLowerCase()
1. Method getInitials() that returns the initials of the name (e.g., "John Doe" → "JD")
2. Method validateName() that returns true if name contains only letters and spaces
3. Method getStudentInfoArray() that returns a String array with [name, studentId as String, gpa as String]
   - Use type conversion to convert int and double to String
1. Method displayFormattedInfo() that displays information using formatted strings

Update your Main class:

```
public class Main {
   public static void main(String[] args) {
      Student s1 = new Student("John Doe", 12345, 3.75);

      System.out.println("Formatted Name: " + s1.formatName());
      System.out.println("Initials: " + s1.getInitials());
      System.out.println("Valid Name: " + s1.validateName());

      String[] info = s1.getStudentInfoArray();
      for (String data : info) {
         System.out.println(data);
```

```
    }

    s1.displayFormattedInfo();
  }
}
```

Solution

```
class Student{

      private String name;
      private int studentId;
      private double gpa;

      public Student(){
            this.name = "Unknown";
            this.studentId = 0;
            this.gpa = 0.0;
      }
      public Student(String name,int studentId,double gpa){
            this.name = name;
            this.studentId = studentId;
            this.gpa = gpa;
      }
      // Task 3

      String formatName(){
         int ind = this.name.indexOf(" ");
         String first = name.substring(0,ind);
         String last = name.substring(ind+1,name.length());
         return last + ", " + first;
      }
      String getInitials(){
            int ind = this.name.indexOf(" ");
         String first = name.substring(0,ind);
         String last = name.substring(ind+1,name.length());

         String firstInitial = Character.toString(first.charAt(0));
         String lastInitial = Character.toString(last.charAt(0));

         return firstInitial+lastInitial;
      }
      boolean validateName(){
            int len = this.name.length();
            String n = name.toLowerCase();
            for(int i=0;i<len;i++){
                  if (!(((int)n.charAt(i) >= 97 && (int)n.charAt(i) <= 122) ||
(int)n.charAt(i) == 32)) return false;
            }
            return true;
      }
      String[] getStudentInfoArray(){
            String[] arr =
{this.name,Integer.toString(this.studentId),Double.toString(this.gpa)};
            return arr;
      }
      void displayFormattedInfo(){
            System.out.println("======== Formatted Student info ========");
```

```
            System.out.printf("Name: %s | Student Id: %d | GPA:
%.2f",name,studentId,gpa);
        }
}
class Main {
    public static void main(String[] args) {
        // Task 3
        Student s1 = new Student("Kamran Gul", 12345, 3.75);
        System.out.println("Formatted Name: " + s1.formatName());
        System.out.println("Initials: " + s1.getInitials());
        System.out.println("Valid Name: " + s1.validateName());
        String[] info = s1.getStudentInfoArray();

  for (String data : info) {
            System.out.println(data);
        }
        s1.displayFormattedInfo();
    }}
```

Output

```
Formatted Name: Gul, Kamran
Initials: KG
Valid Name: true
Kamran Gul
12345
3.75
======== Formatted Student info ========
Name: Kamran Gul | Student Id: 12345 | GPA: 3.75
```

Task 4

Task 4: Add Array Processing Methods

Modify your Student class from Task 3 by adding the following methods that work with arrays:

Requirements:

1. Method processGPAs(double[] gpas) that takes an array of GPAs and:
- Uses for-each loop to find the highest GPA
- Uses break when a GPA > 4.0 is found (invalid)
- Uses continue to skip GPAs < 1.0
- Returns the highest valid GPA

1. Method calculateAverage(double[] grades) that calculates and returns the average of an array
2. Method findGradeInArray(char[] grades, char target) that searches for a grade in an array:
- Uses break when found
- Returns true if found, false otherwise

1. Method getGradeArray(int count) that returns an array of grade characters (A, B, C, D, F) based on count
- Use switch or if/else to assign grades

1. Method modifyArray(int[] numbers) that doubles all values in the array (pass array as parameter)

Update your Main class:

public class Main {

```
    public static void main(String[] args) {
        Student s1 = new Student("John Doe", 12345, 3.75);

        double[] gpas = {3.5, 2.8, 4.0, 3.9, 0.5};
        double highest = s1.processGPAs(gpas);
        System.out.println("Highest GPA: " + highest);

        double[] grades = {85.0, 90.0, 78.0, 92.0};
        double avg = s1.calculateAverage(grades);
        System.out.println("Average: " + avg);

        char[] gradeList = {'A', 'B', 'C', 'A', 'B'};
        boolean found = s1.findGradeInArray(gradeList, 'A');
        System.out.println("Grade A found: " + found);

        int[] nums = {1, 2, 3, 4, 5};
        s1.modifyArray(nums);
        for (int num : nums) {
            System.out.print(num + " ");
        }
    }
}
```

Solution

```
class Student{

	private String name;
	private int studentId;
	private double gpa;

	public Student(){
		this.name = "Unknown";
		this.studentId = 0;
		this.gpa = 0.0;
	}
	public Student(String name,int studentId,double gpa){
		this.name = name;
		this.studentId = studentId;
		this.gpa = gpa;
	}

	// Task 4

	double processGPAs(double[] gpas){
		double highestGpa = 1;
		for(double i : gpas){
			if(i>4.0){
				break;
			}
			else if(i<1.0){
				continue;
```

```
                }
                else if(i>highestGpa){
                        highestGpa = i;
                }
        }
        return highestGpa;
    }
    double calculateAverage(double[] grades){
        double sum = 0;
        int length = grades.length;
        for(double i : grades){
                sum+=i;
        }
        return sum/length;
    }
    boolean findGradeInArray(char[] grades, char target){
        for(char i : grades){
                if(i==target){
                        return true;
                }
        }
    return false;
    }
    int[] modifyArray(int[] numbers){
        for(int i=0;i<numbers.length;i++){
                numbers[i] = numbers[i]*2;
        }
    return numbers;
    }

}
class Main {
    public static void main(String[] args) {
        // Task 4

     Student s1 = new Student("John Doe", 12345, 3.75);

        double[] gpas = {3.5, 2.8, 3.9, 0.5};

        double highest = s1.processGPAs(gpas);
        System.out.println("Highest GPA: " + highest);

        double[] grades = {85.0, 90.0, 78.0, 92.0};
        double avg = s1.calculateAverage(grades);
        System.out.println("Average: " + avg);

        char[] gradeList = {'A', 'B', 'C', 'A', 'B'};
        boolean found = s1.findGradeInArray(gradeList, 'A');
        System.out.println("Grade A found: " + found);

        int[] nums = {1, 2, 3, 4, 5};
        s1.modifyArray(nums);
        for (int num : nums) {
            System.out.print(num + " ");
        }
    }

    }
```

Output

```
Highest GPA: 3.9
Average: 86.25
Grade A found: true
2 4 6 8 10
```

Task 5

Task 5: Add Method Overloading and Multi-dimensional Arrays

Modify your Student class from Task 4 by adding the following methods:

Requirements:

1. **Method Overloading**: Create multiple versions of calculateTotal():

- calculateTotal(int a, int b) - returns int
- calculateTotal(double a, double b) - returns double
- calculateTotal(int a, int b, int c) - returns int

1. Method process2DArray(int[][] matrix) that:

- Takes a 2D array and finds the maximum value
- Returns the maximum value
- Uses nested loops

1. Method getStudentGrades2D() that returns a 2D array representing grades for multiple assignments:

- Returns a 3x4 array (3 students, 4 assignments) with sample data

1. Method displayGradeTable(int[][] grades) that displays grades in table format
2. Method calculateRowAverage(int[][] grades, int row) that returns average of a specific row
3. Method calculateColumnAverage(int[][] grades, int col) that returns average of a specific column

Update your Main class:

```
public class Main {
   public static void main(String[] args) {
      Student s1 = new Student("John Doe", 12345, 3.75);

      // Method overloading
      System.out.println("Sum (int): " + s1.calculateTotal(5, 3));
      System.out.println("Sum (double): " + s1.calculateTotal(5.5, 3.2));
      System.out.println("Sum (3 ints): " + s1.calculateTotal(5, 3, 2));

      // 2D Array
      int[][] matrix = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
      int max = s1.process2DArray(matrix);
      System.out.println("Max value: " + max);

      int[][] grades = s1.getStudentGrades2D();
      s1.displayGradeTable(grades);

      double rowAvg = s1.calculateRowAverage(grades, 0);
      System.out.println("Row 0 average: " + rowAvg);
   }
}
```

Solution

```
class Student{

	private String name;
	private int studentId;
	private double gpa;

	public Student(){
		this.name = "Unknown";
		this.studentId = 0;
		this.gpa = 0.0;
	}
	public Student(String name,int studentId,double gpa){
		this.name = name;
		this.studentId = studentId;
		this.gpa = gpa;
	}

	// Task 5

	int calculateTotal(int a, int b){
		return a+b;
	}
	double calculateTotal(double a,double b){
		return a+b;
	}
	int calculateTotal(int a, int b, int c){
		return a+b+c;
	}
	int process2DArray(int[][] matrix){
		int max = matrix[0][0];
		for(int i=0;i<3;i++){
			for(int j=0;j<3;j++){
				if(matrix[i][j]>max){
					max = matrix[i][j];
				}
			}
		}
		return max;
	}
	int[][] getStudentGrades2D(){
		int[][] grades = {{10,20,30,40},{50,60,70,80},{90,100,10,20}};
		return grades;
	}
	void displayGradeTable(int[][] grades){
		System.out.println("Grade Table: ");
		for(int i=0;i<3;i++){
			System.out.printf("Student %d: ",i+1);
			for(int j=0;j<4;j++){
				System.out.print(grades[i][j] + " ");
			}
			System.out.print("\n");
		}
	}
	double calculateRowAverage(int[][] grades, int row){
		int sum = 0;
```

```
            for(int i=0;i<4;i++){
                  sum += grades[row][i];
            }
            return sum/4.0;
      }
}
class Main {
    public static void main(String[] args) {
            // Task 5

        Student s1 = new Student("John Doe", 12345, 3.75);

        // Method overloading
        System.out.println("Sum (int): " + s1.calculateTotal(5, 3));
        System.out.println("Sum (double): " + s1.calculateTotal(5.5, 3.2));
        System.out.println("Sum (3 ints): " + s1.calculateTotal(5, 3, 2));

        // 2D Array
        int[][] matrix = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
        int max = s1.process2DArray(matrix);
        System.out.println("Max value: " + max);

        int[][] grades = s1.getStudentGrades2D();
        s1.displayGradeTable(grades);

        double rowAvg = s1.calculateRowAverage(grades, 0);
        System.out.println("Row 0 average: " + rowAvg);
    }

}
```

Output

```
Sum (int): 8
Sum (double): 8.7
Sum (3 ints): 10
Max value: 9
Grade Table:
Student 1: 10 20 30 40
Student 2: 50 60 70 80
Student 3: 90 100 10 20
Row 0 average: 25.0
```

Task 6

Task 6: Add Uneven Arrays, Type Conversion, and Complete Integration

Modify your Student class from Task 5 by adding the following final methods:

Requirements:

1. Method createUnevenArray() that returns an uneven 2D array (jagged array):
   - First row: 2 elements
   - Second row: 4 elements
   - Third row: 3 elements
   - Initialize with sample data
1. Method displayUnevenArray(int[][] arr) that displays the uneven array
2. Method convertTypes() that demonstrates type conversion:
   - Convert int to double (automatic)
   - Convert double to int (explicit casting)
   - Convert char to int (explicit casting)
   - Return a String array with conversion results
1. Method demonstrateTypePromotion() that shows automatic type promotion:
   - Add byte + short + int, store in long
   - Add int + float, store in double
   - Return results as a String
1. Method completeStudentReport() that creates a comprehensive report:
   - Uses all previous methods
   - Returns a formatted String with all student information
   - Uses string handling, arrays, and all control structures

Update your Main class to demonstrate everything:

```
public class Main {
   public static void main(String[] args) {
      Student s1 = new Student("John Doe", 12345, 3.75);
      Student s2 = new Student("Jane Smith", 12346, 3.90);

      // Display complete information
      System.out.println("=== Student 1 ===");
      System.out.println(s1.completeStudentReport());

      System.out.println("\n=== Student 2 ===");
      System.out.println(s2.completeStudentReport());

      // Demonstrate uneven arrays
      int[][] uneven = s1.createUnevenArray();
      s1.displayUnevenArray(uneven);

      // Demonstrate type conversion
      String[] conversions = s1.convertTypes();
      for (String conv : conversions) {
         System.out.println(conv);
      }

      // Demonstrate type promotion
      String promotion = s1.demonstrateTypePromotion();
```

```
        System.out.println(promotion);

        // Array operations
        double[] gpas = {3.5, 3.8, 2.9, 3.95};
        double highest = s1.processGPAs(gpas);
        System.out.println("Highest GPA from array: " + highest);

        // Method overloading demonstration
        System.out.println("Overloaded methods:");
        System.out.println(s1.calculateTotal(10, 20));
        System.out.println(s1.calculateTotal(10.5, 20.5));
        System.out.println(s1.calculateTotal(10, 20, 30));
    }
}
```

Solution

```
class Student{

	private String name;
	private int studentId;
	private double gpa;

	public Student(){
		this.name = "Unknown";
		this.studentId = 0;
		this.gpa = 0.0;
	}
	public Student(String name,int studentId,double gpa){
		this.name = name;
		this.studentId = studentId;
		this.gpa = gpa;
	}

	// Task 4

	double processGPAs(double[] gpas){
		double highestGpa = 1;
		for(double i : gpas){
			if(i>4.0){
				break;
			}
			else if(i<1.0){
				continue;
			}
			else if(i>highestGpa){
				highestGpa = i;
			}
		}
		return highestGpa;
	}
	// Task 5
```

```
      int calculateTotal(int a, int b){
            return a+b;
      }
      double calculateTotal(double a,double b){
            return a+b;
      }
      int calculateTotal(int a, int b, int c){
            return a+b+c;
      }

      // Task 6
      int[][] createUnevenArray(){
            int[][] array;
            // array[0] = new int[2];
            // array[1] = new int [4];
            // array[2] = new int[3];
            array = new int[][]{
            {0,1},
            {2,3,4,5},
            {6,7,8}
      };
      return array;
      }
      void displayUnevenArray(int[][] arr){
            for(int i=0; i<arr.length;i++){
                  System.out.printf("Row %d: ",i+1);
                  for(int j=0; j<arr[i].length;j++){
                        System.out.print(arr[i][j] + " ");
                  }
                  System.out.print("\n");
            }
      }
      String[] convertTypes(){
            int a = 5;
            double b = 3.14;
            char c = 'K';

            double con_a = a;
            int con_b = (int)b;
            int con_c = (int)c;

            String[] str =
{String.valueOf(con_a),String.valueOf(con_b),String.valueOf(con_c)};
            return str;
      }
      String demonstrateTypePromotion(){
            byte a = 1;
            short b = 2;
            int c = 3;
            long res1 = a+b+c;

            float d = 3.14f;
            double res2 = c+d;

            return String.valueOf(res1) + String.valueOf(res2);
      }

}
class Main {
```

```
    public static void main(String[] args) {
           //Task 6

            Student s1 = new Student("John Doe", 12345, 3.75);
        Student s2 = new Student("Jane Smith", 12346, 3.90);

        // Display complete information
        // System.out.println("=== Student 1 ===");
        // System.out.println(s1.completeStudentReport());

        // System.out.println("\n=== Student 2 ===");
        // System.out.println(s2.completeStudentReport());

        // Demonstrate uneven arrays
        int[][] uneven = s1.createUnevenArray();
        System.out.println("Uneven Array: ");
        s1.displayUnevenArray(uneven);

        // // Demonstrate type conversion
        System.out.println("\nType Conversion results: ");
        String[] conversions = s1.convertTypes();
        for (String conv : conversions) {
            System.out.print(conv + " ");
        }

        // Demonstrate type promotion
        System.out.println("\n\nType Promotion results: ");
        String promotion = s1.demonstrateTypePromotion();
        System.out.println(promotion);

        // Array operations
        double[] gpas = {3.5, 3.8, 2.9, 3.95};
        double highest = s1.processGPAs(gpas);
        System.out.println("\nHighest GPA from array: " + highest);

        // // Method overloading demonstration
        System.out.println("\nOverloaded methods:");
        System.out.println(s1.calculateTotal(10, 20));
        System.out.println(s1.calculateTotal(10.5, 20.5));
        System.out.println(s1.calculateTotal(10, 20, 30));
    }

}
```

Output

```
Uneven Array:
Row 1: 0 1
Row 2: 2 3 4 5
Row 3: 6 7 8

Type Conversion results:
5.0 3 75

Type Promotion results:
66.1400003433227754

Highest GPA from array: 3.95

Overloaded methods:
30
31.0
60
```