

PRACTICE PROGRAM

Q1 Applying the concept of Method Overriding and runtime polymorphism to create a superclass called Figure tha stores the dim of 2-D object. It also defines a method called area() that computes the area of an object. The program derives 2 subclasses from Figure. The first is Rectangle and the second is Triangle. Use dynamic method dispatch to display output as;

Inside area of rec

Area is 45

Inside area of tri

Area is 40

Area of figure is undefined

Area is 0

```
A= class Figure {  
  
    double dim1; double dim2;  
  
    Figure(double a, double b) { dim1 = a; dim2 = b; }  
  
    double area() {  
  
        System.out.println("Area for Figure is undefined."); return 0;  
  
    }  
  
}  
  
class Rectangle extends Figure {  
  
    Rectangle(double a, double b)  
  
    {  
  
        super(a, b); } // override area for rectangle  
  
    double area()  
  
    {  
  
        System.out.println("Inside Area for Rectangle."); return dim1 * dim2;  
  
    }  
  
}
```

```

}

class Triangle extends Figure {

Triangle(double a, double b)

{

super(a, b); }

double area() // override area for right triangle

{

System.out.println("Inside Area for Triangle."); return dim1 * dim2 / 2;

}

}

class AbstractAreas {

public static void main(String args[])

{

Figure f = new Figure(10, 10);

Rectangle r = new Rectangle(9, 5); Triangle t = new Triangle(10, 8);

Figure figref; // this is OK, no object is created

figref = r;

System.out.println("Area is " + figref.area());

figref = t;

System.out.println("Area is " + figref.area());

figref = f;

System.out.println("Area is " + figref.area());

} }

```

Q2 Define a STUDENT class with USN, Name, Marks in 3 tests of a subject. Declare an array of n STUDENT objects. Using appropriate methods, find the average of 2 better marks for each student. Print the USN, Name and the average marks of all the students. Use constructors and destructors to initialize and destroy the object.

```
A= import java.util.Scanner;
```

```
class Student {
    private String usn;
    private String name;
    private int[] marks = new int[3];

    // Constructor
    public Student(String usn, String name, int[] marks) {
        this.usn = usn;
        this.name = name;
        this.marks = marks;
    }

    // Getter and Setter methods
    public String getUsn() {
        return usn;
    }

    public void setUsn(String usn) {
        this.usn = usn;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int[] getMarks() {
        return marks;
    }

    public void setMarks(int[] marks) {
        this.marks = marks;
    }

    // Method to find the average of 2 better marks
```

```

public double findAverage() {
    int sum = 0;
    int count = 0;

    for (int mark : marks) {
        sum += mark;
        count++;
    }

    return (double) sum / count;
}

// Destructor
@Override
protected void finalize() throws Throwable {
    super.finalize();
    System.out.println("Student object destroyed.");
}
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = scanner.nextInt();

        Student[] students = new Student[n];

        for (int i = 0; i < n; i++) {
            System.out.print("Enter the USN: ");
            String usn = scanner.next();
            System.out.print("Enter the Name: ");
            String name = scanner.next();
            System.out.print("Enter the Marks in 3 tests: ");
            int[] marks = new int[3];
            for (int j = 0; j < 3; j++) {
                marks[j] = scanner.nextInt();
            }
            students[i] = new Student(usn, name, marks);
        }

        System.out.println("\nUSN\tName\tAverage Marks");
        for (Student student : students) {
            System.out.println(student.getUsn() + "\t" + student.getName() + "\t" +
student.findAverage());
        }
    }
}

```

```

Enter the number of students: 2
Enter the USN: bms21
Enter the Name: shreya
Enter the Marks in 3 tests: 32
31
35
Enter the USN: bms22
Enter the Name: 40
< Enter the Marks in 3 tests: 40
35
33

USN      Name      Average Marks
bms21    shreya    32.666666666666664
bms22    40         36.0

```

Q3 Write a program to Implement the TIME class. Each object of this class will represent a specified time of the day, storing the hours, minutes and seconds as integer. Include a constructor, access methods, a method advance (int h, int m,int s) to advance the given time, method currenttime (int h, int m, int s) to update the current time of an existing object and a print method to print the details.

A=

```

class Time {
    private int hours;
    private int minutes;
    private int seconds;

    // Constructor
    public Time(int hours, int minutes, int seconds) {
        this.hours = hours;
        this.minutes = minutes;
        this.seconds = seconds;
    }
}

```

```

// Getter and Setter methods
public int getHours() {
    return hours;
}

public void setHours(int hours) {
    this.hours = hours;
}

public int getMinutes() {
    return minutes;
}

public void setMinutes(int minutes) {
    this.minutes = minutes;
}

public int getSeconds() {
    return seconds;
}

public void setSeconds(int seconds) {
    this.seconds = seconds;
}

// Method to advance the time
public void advance(int h, int m, int s) {
    this.seconds += s;
    this.minutes += m + this.seconds / 60;
    this.hours += h + this.minutes / 60;
    this.seconds %= 60;
    this.minutes %= 60;
    this.hours %= 24;
}

// Method to update the current time
public void currentTime(int h, int m, int s) {
    this.hours = h;
    this.minutes = m;
    this.seconds = s;
}

// Method to print the time details
public void print() {
    System.out.println("Hours: " + hours + " Minutes: " + minutes + " Seconds: " + seconds);
}
}

```

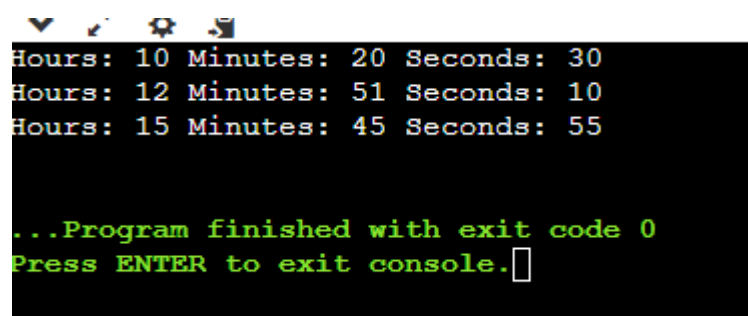
```

public class Main {
    public static void main(String[] args) {
        Time time = new Time(10, 20, 30);
        time.print();

        time.advance(2, 30, 40);
        time.print();

        time.currentTime(15, 45, 55);
        time.print();
    }
}

```



```

Hours: 10 Minutes: 20 Seconds: 30
Hours: 12 Minutes: 51 Seconds: 10
Hours: 15 Minutes: 45 Seconds: 55

...Program finished with exit code 0
Press ENTER to exit console.

```

Q4 Write a program to implement a class MATRIX for 2 X 2 Matrix. Include a default constructor, an inverse() method that returns the inverse of the matrix, a determinant() method that returns the determinant of the matrix, a Boolean method isSingular() that returns 1 or 0 according to whether the determinant is 0 and a print() method to print the details.

A=

```

class Matrix {
    private double[][] matrix;

    // Default constructor
    public Matrix() {
        matrix = new double[2][2];
    }

    // Access methods
    public double get(int i, int j) {
        return matrix[i][j];
    }

    public void set(int i, int j, double value) {
        matrix[i][j] = value;
    }
}

```

```

    }

    // Method to find the determinant
    public double determinant() {
        return matrix[0][0] * matrix[1][1] - matrix[0][1] * matrix[1][0];
    }

    // Method to check if the matrix is singular
    public boolean isSingular() {
        return determinant() == 0;
    }

    // Method to find the inverse of the matrix
    public Matrix inverse() {
        double det = determinant();
        if (det == 0) {
            throw new ArithmeticException("The matrix is singular and cannot be inverted.");
        }

        Matrix inverse = new Matrix();
        inverse.set(0, 0, matrix[1][1] / det);
        inverse.set(0, 1, -matrix[0][1] / det);
        inverse.set(1, 0, -matrix[1][0] / det);
        inverse.set(1, 1, matrix[0][0] / det);

        return inverse;
    }

    // Method to print the matrix details
    public void print() {
        System.out.println("Matrix: ");
        for (int i = 0; i < 2; i++) {
            for (int j = 0; j < 2; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Matrix matrix = new Matrix();
        matrix.set(0, 0, 1);
        matrix.set(0, 1, 2);
        matrix.set(1, 0, 3);
        matrix.set(1, 1, 4);
        matrix.print();
    }
}

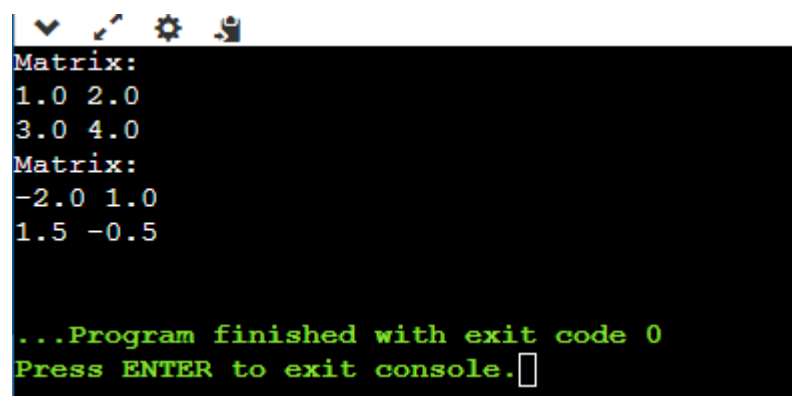
```



```

    if (!matrix.isSingular()) {
        Matrix inverse = matrix.inverse();
        inverse.print();
    } else {
        System.out.println("The matrix is singular and cannot be inverted.");
    }
}
}
}

```



The screenshot shows a Java IDE console window with a dark background. At the top, there are four icons: a checkmark, a pencil, a gear, and a trash can. The console output is as follows:

```

Matrix:
1.0 2.0
3.0 4.0
Matrix:
-2.0 1.0
1.5 -0.5

...Program finished with exit code 0
Press ENTER to exit console.

```

Q5 Make a class EMP with data members empno, empname and salary. Create two objects of EMP. Display the details of the employee who has got maximum salary using this pointer.

A=

```

class EMP {
    private int empno;
    private String empname;
    private float salary;

    public EMP(int empno, String empname, float salary) {
        this.empno = empno;
        this.empname = empname;
        this.salary = salary;
    }

    public float getSalary() {
        return salary;
    }

    public void displayDetails() {
        System.out.println("Employee Number: " + empno);
        System.out.println("Employee Name: " + empname);
    }
}

```

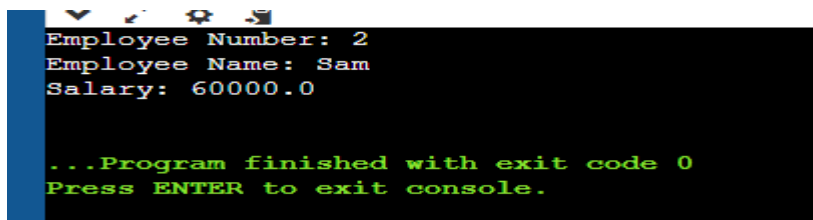
```

        System.out.println("Salary: " + salary);
    }
}

public class Main {
    public static void main(String[] args) {
        EMP emp1 = new EMP(1, "John", 50000);
        EMP emp2 = new EMP(2, "Sam", 60000);

        if (emp1.getSalary() > emp2.getSalary()) {
            emp1.displayDetails();
        } else {
            emp2.displayDetails();
        }
    }
}

```



```

Employee Number: 2
Employee Name: Sam
Salary: 60000.0

...Program finished with exit code 0
Press ENTER to exit console.

```

Q6 Create a class Student with data members-int usn, float cmarks, float smarks. Maximum marks of both cmarks and smarks are 50. Include methods to do the following:

- i. accept the details from the user**
- ii. display the grade according to the given slab along with their usn.**

Total	Grade	Total	Grade
>=75	S	Between 40 and 59	B
Between 60 and 74	A	<40	F

A= import java.util.Scanner;

```

class Student {
    private int usn;
    private float cmarks;
    private float smarks;

    public void acceptDetails() {
        Scanner sc = new Scanner(System.in);
    }
}

```

```

        System.out.println("Enter USN: ");
        usn = sc.nextInt();
        System.out.println("Enter marks in computer applications: ");
        cmarks = sc.nextFloat();
        System.out.println("Enter marks in soft skills: ");
        smarks = sc.nextFloat();
    }

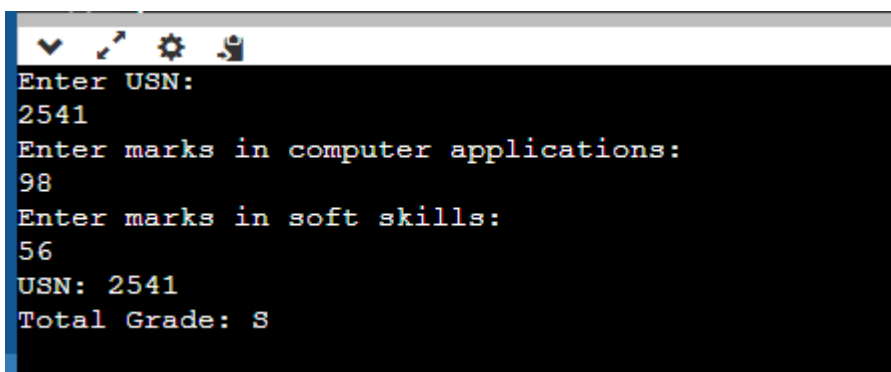
    public void calculateGrade() {
        float totalMarks = cmarks + smarks;
        String totalGrade = "";

        if (totalMarks >= 75) {
            totalGrade = "S";
        } else if (totalMarks >= 40 && totalMarks < 59) {
            totalGrade = "B";
        } else if (totalMarks >= 60 && totalMarks < 74) {
            totalGrade = "A";
        } else {
            totalGrade = "F";
        }

        System.out.println("USN: " + usn);
        System.out.println("Total Grade: " + totalGrade);
    }
}

public class Main {
    public static void main(String[] args) {
        Student s = new Student();
        s.acceptDetails();
        s.calculateGrade();
    }
}

```



```

Enter USN:
2541
Enter marks in computer applications:
98
Enter marks in soft skills:
56
USN: 2541
Total Grade: S

```

Q7 Write a program to create a base class STUDENT(Name, Regno, Age) and using inheritance create classes UG_Student and PG_Student having fields as semester, fees and stipend. Enter the data for atleast 5 students. Find the average age semester wise for all UG and PG students separately.

A=

```
import java.util.Scanner;

class STUDENT {
    String name;
    int regNo;
    int age;

    // Constructor for STUDENT class
    public STUDENT(String name, int regNo, int age) {
        this.name = name;
        this.regNo = regNo;
        this.age = age;
    }
}

class UG_Student extends STUDENT {
    int semester;
    double fees;

    // Constructor for UG_Student class
    public UG_Student(String name, int regNo, int age, int semester, double fees) {
        super(name, regNo, age);
        this.semester = semester;
        this.fees = fees;
    }
}

class PG_Student extends STUDENT {
    int semester;
    double stipend;

    // Constructor for PG_Student class
    public PG_Student(String name, int regNo, int age, int semester, double stipend) {
        super(name, regNo, age);
        this.semester = semester;
        this.stipend = stipend;
    }
}
```

```
}
```

```
public class Main {  
    public static void main(String[] args) {  
        Scanner scanner = new Scanner(System.in);  
  
        // Creating arrays to store UG and PG students  
        UG_Student[] ugStudents = new UG_Student[5];  
        PG_Student[] pgStudents = new PG_Student[5];  
  
        // Input data for UG students  
        System.out.println("Enter details for UG students:");  
        for (int i = 0; i < 5; i++) {  
            System.out.println("UG Student " + (i + 1) + ":");  
            System.out.print("Name: ");  
            String name = scanner.next();  
            System.out.print("Reg No: ");  
            int regNo = scanner.nextInt();  
            System.out.print("Age: ");  
            int age = scanner.nextInt();  
            System.out.print("Semester: ");  
            int semester = scanner.nextInt();  
            System.out.print("Fees: ");  
            double fees = scanner.nextDouble();  
  
            ugStudents[i] = new UG_Student(name, regNo, age, semester, fees);  
        }  
  
        // Input data for PG students  
        System.out.println("\nEnter details for PG students:");  
        for (int i = 0; i < 5; i++) {  
            System.out.println("PG Student " + (i + 1) + ":");  
            System.out.print("Name: ");  
            String name = scanner.next();  
            System.out.print("Reg No: ");  
            int regNo = scanner.nextInt();  
            System.out.print("Age: ");  
            int age = scanner.nextInt();  
            System.out.print("Semester: ");  
            int semester = scanner.nextInt();  
            System.out.print("Stipend: ");  
            double stipend = scanner.nextDouble();  
  
            pgStudents[i] = new PG_Student(name, regNo, age, semester, stipend);  
        }  
  
        // Calculate average age semester-wise for UG students  
        int[] ugSemesterTotalAge = new int[8];  
    }  
}
```

```

int[] ugSemesterStudentCount = new int[8];

for (UG_Student student : ugStudents) {
    ugSemesterTotalAge[student.semester] += student.age;
    ugSemesterStudentCount[student.semester]++;
}

System.out.println("\nAverage age semester-wise for UG students:");
for (int i = 1; i <= 7; i++) {
    if (ugSemesterStudentCount[i] > 0) {
        double averageAge = (double) ugSemesterTotalAge[i] / ugSemesterStudentCount[i];
        System.out.println("Semester " + i + ": " + averageAge);
    }
}

// Calculate average age semester-wise for PG students
int[] pgSemesterTotalAge = new int[8];
int[] pgSemesterStudentCount = new int[8];

for (PG_Student student : pgStudents) {
    pgSemesterTotalAge[student.semester] += student.age;
    pgSemesterStudentCount[student.semester]++;
}

System.out.println("\nAverage age semester-wise for PG students:");
for (int i = 1; i <= 7; i++) {
    if (pgSemesterStudentCount[i] > 0) {
        double averageAge = (double) pgSemesterTotalAge[i] / pgSemesterStudentCount[i];
        System.out.println("Semester " + i + ": " + averageAge);
    }
}

scanner.close();
}
}

```

Enter details for UG students:

UG Student 1:

Name: SHRE M

Reg No: 266

Age: 12

Semester: 7

Fees: 7986

UG Student 2:

Name: SHREP

Reg No: 546

Age: 21

Semester: 3

Fees: 7689

UG Student 3:

Name: SHRE S

Reg No: 568

Age: 31

Semester: 3

Fees: 7898

UG Student 4:

Name: SHECY

Reg No: 785

Age: 22

Semester: 3

Fees: 9809

UG Student 5:

Name: SINDH

Reg No: 879

Age: 4

Semester: 4

Fees: 98071

Enter details for PG students:

PG Student 1:

Name: FVGJHJ

Reg No: 8098

Age: 23

Semester: 4

Stipend: 7687

PG Student 2:

Name: HGFDH

Reg No: 768

Age: 23

Semester: 4

Stipend: 7898

PG Student 3:

Name: HBKBGK

Reg No: 9800

Age: 22

Semester: 4

Stipend: 8798

PG Student 4:

Name: GHJG

Reg No: 7898

Q8 Implement a class called PERSON having data members as name, dob and address. Derive a class STUDENT from PERSON having data members rollno and sem. Derive another class EXAM from student which has data members marks1, marks2 and computes the average and displays the topper of the class. Use suitable methods to accept and display data in these classes.

A=

```
import java.util.Scanner;

class PERSON {
    String name;
    String dob;
    String address;

    // Constructor for PERSON class
    public PERSON(String name, String dob, String address) {
        this.name = name;
        this.dob = dob;
        this.address = address;
    }

    // Method to display PERSON data
    public void displayPersonDetails() {
        System.out.println("Name: " + name);
        System.out.println("Date of Birth: " + dob);
        System.out.println("Address: " + address);
    }
}

class STUDENT extends PERSON {
    int rollno;
    int sem;

    // Constructor for STUDENT class
    public STUDENT(String name, String dob, String address, int rollno, int sem) {
        super(name, dob, address);
        this.rollno = rollno;
        this.sem = sem;
    }
}
```



```

// Method to display STUDENT data
public void displayStudentDetails() {
    displayPersonDetails(); // Reusing displayPersonDetails() from the parent class
    System.out.println("Roll Number: " + rollno);
    System.out.println("Semester: " + sem);
}
}

class EXAM extends STUDENT {
    int marks1;
    int marks2;

    // Constructor for EXAM class
    public EXAM(String name, String dob, String address, int rollno, int sem, int marks1, int
marks2) {
        super(name, dob, address, rollno, sem);
        this.marks1 = marks1;
        this.marks2 = marks2;
    }

    // Method to calculate and display average marks
    public void displayAverageMarks() {
        int average = (marks1 + marks2) / 2;
        System.out.println("Average Marks: " + average);
    }

    // Method to display the topper of the class (assuming marks1 is considered for
comparison)
    public void displayClassTopper(EXAM[] students) {
        int maxMarks = -1;
        String topperName = "";

        for (EXAM student : students) {
            if (student.marks1 > maxMarks) {
                maxMarks = student.marks1;
                topperName = student.name;
            }
        }

        System.out.println("Class Topper: " + topperName);
    }
}

public class Main {
    public static void main(String[] args) {
        // Creating an array of EXAM objects for demonstration
        EXAM[] students = new EXAM[3];

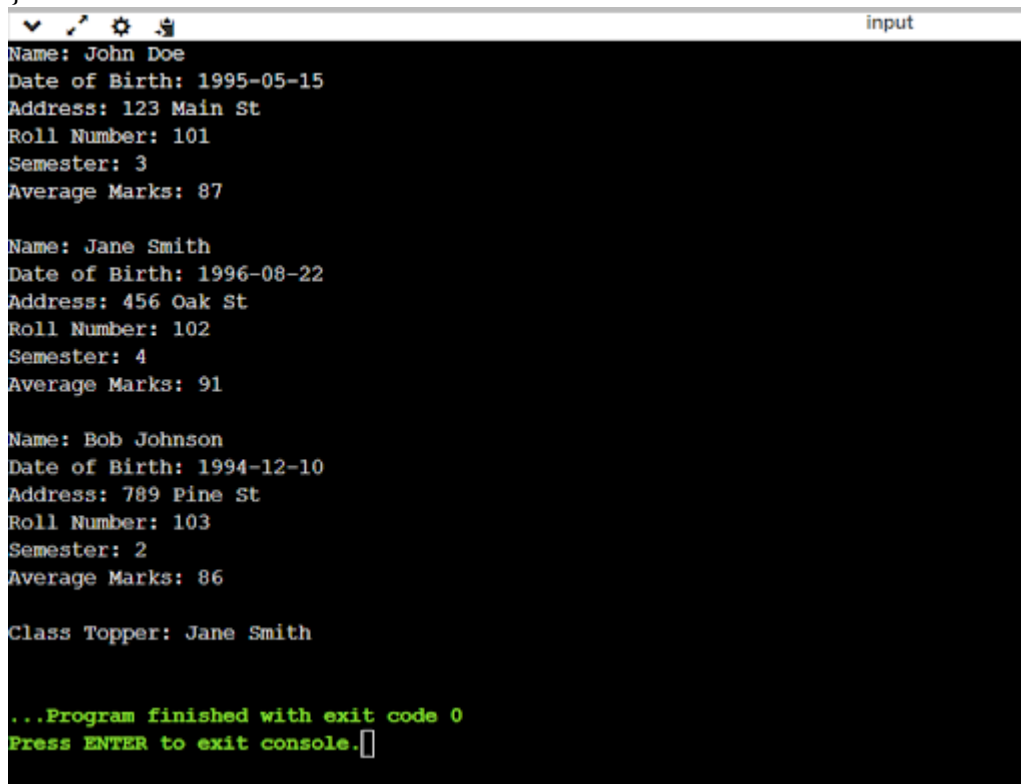
        // Adding sample data

```

```
students[0] = new EXAM("John Doe", "1995-05-15", "123 Main St", 101, 3, 90, 85);
students[1] = new EXAM("Jane Smith", "1996-08-22", "456 Oak St", 102, 4, 95, 88);
students[2] = new EXAM("Bob Johnson", "1994-12-10", "789 Pine St", 103, 2, 80, 92);
```

```
// Displaying student details
for (EXAM student : students) {
    student.displayStudentDetails();
    student.displayAverageMarks();
    System.out.println();
}

// Displaying the topper of the class
students[0].displayClassTopper(students);
}
```



```
input
Name: John Doe
Date of Birth: 1995-05-15
Address: 123 Main St
Roll Number: 101
Semester: 3
Average Marks: 87

Name: Jane Smith
Date of Birth: 1996-08-22
Address: 456 Oak St
Roll Number: 102
Semester: 4
Average Marks: 91

Name: Bob Johnson
Date of Birth: 1994-12-10
Address: 789 Pine St
Roll Number: 103
Semester: 2
Average Marks: 86

Class Topper: Jane Smith

...Program finished with exit code 0
Press ENTER to exit console.[]
```