West Bengal State University

Department of Computer Science

Assignment

Subject: Object Oriented Programming

Course Code: CMSPCOR10P

Name: Sayan Dutta

Semester: II

Session: 2024–2026

Submitted to:

Dr. Krishnendu Basuli

Department of Computer Science

West Bengal State University

Contents

1 Shipping database using multilevel inheritance

Write a program in Java for a shipping database using multilevel inheritance with at least 3 classes. All classes must have two member variables and two member methods.

1.1 Java Implementation

```
import java.util.Scanner;
import java.util.ArrayList;

class shipment{
    protected String shipmentId;
    protected String destination;

    public shipment(String shipmentId, String destination) {
        this.shipmentId = shipmentId;
        this.destination = destination;
    }

    public void trackShipment() {
        System.out.println("Tracking shipment ID: " + shipmentId + "
           heading to " + destination);
    }

    public void updateDestination(String newDestination) {
        this.destination = newDestination;
        System.out.println("Destination updated for shipment " +
           shipmentId);
    }
}
class cargo extends shipment {
    protected double cargoWeight;
    protected String cargoType;

    public cargo(String shipmentId, String destination, double cargoWeight
       , String cargoType) {
        super(shipmentId, destination);
        this.cargoWeight = cargoWeight;
        this.cargoType = cargoType;
    }

    public double calculateShippingCost() {
        return cargoWeight * 2.5;
    }

    public void displayCargoInfo() {
        System.out.println("Cargo Type: " + cargoType + ", Weight: " +
           cargoWeight + " kg");
    }
```

```
    public void updateCargoType(String newType) {
        this.cargoType = newType;
    }

    public void updateCargoWeight(double newWeight) {
        this.cargoWeight = newWeight;
    }
}
class container extends cargo {
    private String containerNumber;
    private String containerSize;

    public container(String shipmentId, String destination, double
       cargoWeight, String cargoType,
                     String containerNumber, String containerSize) {
        super(shipmentId, destination, cargoWeight, cargoType);
        this.containerNumber = containerNumber;
        this.containerSize = containerSize;
    }

    public void assignContainer(String number, String size) {
        this.containerNumber = number;
        this.containerSize = size;
    }

    public void displayContainerDetails() {
        System.out.println("Container: " + containerNumber + " (" +
           containerSize + ")");
    }

    public String getContainerNumber() {
        return containerNumber;
    }

    public String getContainerSize() {
        return containerSize;
    }
}
public class ShippingDatabase {
    private static final ArrayList<container> shipments = new ArrayList
       <>();
    private static final Scanner sc = new Scanner(System.in);

    public static void main(String[] args) {
        boolean running = true;

        while(running) {
            System.out.println("\n===== Shipping Database Menu =====");
            System.out.println("1. Add New Shipment");
            System.out.println("2. View All Shipments");
            System.out.println("3. Update destination");
            System.out.println("4. Update cargo details");
```

```
            System.out.println("5. Update container details");
            System.out.println("6. Exit");
            System.out.print("Enter your choice (1-6): ");

            int choice = getIntInput(6);

            switch (choice) {
                case 1 -> addNewShipment();
                case 2 -> viewAllShipments();
                case 3 -> updateShipmentDestination();
                case 4 -> updateCargoDetails();
                case 5 -> updateContainerDetails();
                case 6 -> running = confirmExit();
            }
        }
        System.out.println("Program exited. Thank you!");
        sc.close();
    }

    private static void addNewShipment() {
        System.out.println("\n=== Enter New Shipment Details ===");

        String shipmentId;
        while(true) {
            System.out.print("Shipment ID: ");
            shipmentId = sc.nextLine().trim();

            if(shipmentId.isEmpty()) {
                System.out.println("Error: Shipment ID cannot be empty!");
                continue;
            }

            if(isShipmentIdExists(shipmentId)) {
                System.out.println("Error: Shipment ID already exists!");
            } else {
                break;
            }
        }
        System.out.print("Destination: ");
        String destination = sc.nextLine();

        System.out.print("Cargo Weight (kg): ");
        double cargoWeight = getDoubleInput();

        System.out.print("Cargo Type: ");
        String cargoType = sc.nextLine();

        System.out.print("Container Number: ");
        String containerNumber = sc.nextLine();

        System.out.print("Container Size: ");
        String containerSize = sc.nextLine();
```

```
        container newContainer = new container(shipmentId, destination,
            cargoWeight,
                cargoType, containerNumber, containerSize);
        shipments.add(newContainer);

        System.out.println("\nShipment added successfully!");
    }

    private static boolean isShipmentIdExists(String id) {
        for(container container : shipments) {
            if(container.shipmentId.equalsIgnoreCase(id)) {
                return true;
            }
        }
        return false;
    }

    private static void viewAllShipments() {
        if(shipments.isEmpty()) {
            System.out.println("\nNo shipments found!");
            return;
        }

        System.out.println("\n=== All Shipments ===");
        for(int i = 0; i < shipments.size(); i++) {
            System.out.println("\nShipment #" + (i+1));
            container container = shipments.get(i);
            container.trackShipment();
            container.displayCargoInfo();
            System.out.printf("Shipping Cost: $%.2f%n", container.
                calculateShippingCost());
            container.displayContainerDetails();
            System.out.println("----------------------");
        }
    }

    private static void updateShipmentDestination() {
        if(shipments.isEmpty()) {
            System.out.println("\nNo shipments available to update!");
            return;
        }

        System.out.println("\n=== Update Destination ===");
        viewAllShipmentsBrief();
        System.out.print("Enter shipment number to update: ");
        int index = getIntInput(shipments.size()) - 1;

        System.out.print("Enter new destination: ");
        String newDest = sc.nextLine();

        shipments.get(index).updateDestination(newDest);
        System.out.println("Destination updated successfully!");
    }
```

```
    private static void updateCargoDetails() {
        if(shipments.isEmpty()) {
            System.out.println("\nNo shipments available to update!");
            return;
        }

        System.out.println("\n=== Update Cargo Details ===");
        viewAllShipmentsBrief();
        System.out.print("Enter shipment number to update: ");
        int index = getIntInput(shipments.size()) - 1;

        System.out.print("Enter new cargo weight (kg): ");
        double newWeight = getDoubleInput();
        sc.nextLine(); // Clear buffer

        System.out.print("Enter new cargo type: ");
        String newType = sc.nextLine();

        container container = shipments.get(index);
        container.updateCargoWeight(newWeight);
        container.updateCargoType(newType);
        System.out.println("Cargo details updated successfully!");
    }

    private static void updateContainerDetails() {
        if(shipments.isEmpty()) {
            System.out.println("\nNo shipments available to update!");
            return;
        }

        System.out.println("\n=== Update Container Details ===");
        viewAllShipmentsBrief();
        System.out.print("Enter shipment number to update: ");
        int index = getIntInput(shipments.size()) - 1;

        System.out.print("Enter new container number: ");
        String newNumber = sc.nextLine();

        System.out.print("Enter new container size: ");
        String newSize = sc.nextLine();

        shipments.get(index).assignContainer(newNumber, newSize);
        System.out.println("Container details updated successfully!");
    }

    private static boolean confirmExit() {
        System.out.print("\nAre you sure you want to exit? (yes/no): ");
        String answer = sc.nextLine();
        return answer.equalsIgnoreCase("yes");
    }

    private static int getIntInput(int max) {
        while(true) {
```

```
            try {
                int input = Integer.parseInt(sc.nextLine());
                if(input >= 1 && input <= max) return input;
                System.out.printf("Please enter a number between %d and %d
                    : ", 1, max);
            } catch(NumberFormatException e) {
                System.out.print("Invalid input. Please enter a number: ")
                    ;
            }
        }
    }

    private static double getDoubleInput() {
        while(true) {
            try {
                return Double.parseDouble(sc.nextLine());
            } catch(NumberFormatException e) {
                System.out.print("Invalid input. Please enter a number: ")
                    ;
            }
        }
    }

    private static void viewAllShipmentsBrief() {
        for(int i = 0; i < shipments.size(); i++) {
            container c = shipments.get(i);
            System.out.println("Shipment #" + (i+1) +
                    " - ID: " + c.shipmentId +
                    " | Container: " + c.getContainerNumber() +
                    " (" + c.getContainerSize() + ")");
        }
    }
}
```

1.2 Program Output

```
===== Shipping Database Menu =====
1. Add New Shipment
2. View All Shipments
3. Update destination
4. Update cargo details
5. Update container details
6. Exit
Enter your choice (1-6): 1

=== Enter New Shipment Details ===
Shipment ID: SHP-001
Destination: New York
Cargo Weight (kg): 1500
Cargo Type: Electronics
Container Number: CNTR-001
Container Size: 40ft
```

```
Shipment added successfully!

===== Shipping Database Menu =====
1. Add New Shipment
2. View All Shipments
3. Update destination
4. Update cargo details
5. Update container details
6. Exit
Enter your choice (1-6): 2

=== All Shipments ===

Shipment #1
Tracking shipment ID: SHP-001 heading to New York
Cargo Type: Electronics, Weight: 1500.0 kg
Shipping Cost: \$3750.00
Container: CNTR-001 (40ft)
----------------------

===== Shipping Database Menu =====
1. Add New Shipment
2. View All Shipments
3. Update destination
4. Update cargo details
5. Update container details
6. Exit
Enter your choice (1-6): 3

=== Update Destination ===
Shipment #1 - ID: SHP-001 | Container: CNTR-001 (40ft)
Enter shipment number to update: 1
Enter new destination: Chicago
Destination updated for shipment SHP-001
Destination updated successfully!

===== Shipping Database Menu =====
1. Add New Shipment
2. View All Shipments
3. Update destination
4. Update cargo details
5. Update container details
6. Exit
Enter your choice (1-6): 6

Are you sure you want to exit? (yes/no): yes
Program exited. Thank you!
```

2 Shape Area Calculation: Polymorphism & Inheritance

Write a program in Java to calculate the area of different shapes using polymorphism and inheritance.

2.1 Java Implementation

```
import java.util.Scanner;

abstract class Shape {
    public abstract double calculateArea();

    public void displayArea() {
        System.out.printf("Area: %.2f\n", calculateArea());
    }
}
class Circle extends Shape {
    private final double radius;

    public Circle(double radius) {
        this.radius = radius;
    }

    @Override
    public double calculateArea() {
        return Math.PI * radius * radius;
    }
}
class Rectangle extends Shape {
    private final double length;
    private final double width;

    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    @Override
    public double calculateArea() {
        return length * width;
    }
}
class Triangle extends Shape {
    private final double base;
    private final double height;

    public Triangle(double base, double height) {
        this.base = base;
        this.height = height;
    }

    @Override
    public double calculateArea() {
        return (base * height) / 2;
```

```
    }
}
class Rhombus extends Shape {
    private final double diagonal1;
    private final double diagonal2;

    public Rhombus(double diagonal1, double diagonal2) {
        this.diagonal1 = diagonal1;
        this.diagonal2 = diagonal2;
    }

    @Override
    public double calculateArea() {
        return (diagonal1 * diagonal2) / 2;
    }
}
class Parallelogram extends Shape {
    private final double base;
    private final double height;

    public Parallelogram(double base, double height) {
        this.base = base;
        this.height = height;
    }

    @Override
    public double calculateArea() {
        return base * height;
    }
}
class Trapezium extends Shape {
    private final double base1;
    private final double base2;
    private final double height;

    public Trapezium(double base1, double base2, double height) {
        this.base1 = base1;
        this.base2 = base2;
        this.height = height;
    }

    @Override
    public double calculateArea() {
        return ((base1 + base2) / 2) * height;
    }
}
class EquilateralTriangle extends Shape {
    private final double side;

    public EquilateralTriangle(double side) {
        this.side = side;
    }

    @Override
```

```
    public double calculateArea() {
        return (Math.sqrt(3) / 4) * side * side;
    }
}
class Sector extends Shape {
    private final double radius;
    private final double angle; // in degrees

    public Sector(double radius, double angle) {
        this.radius = radius;
        this.angle = angle;
    }

    @Override
    public double calculateArea() {
        return (Math.PI * radius * radius) * (angle / 360);
    }
}

public class ShapeAreaCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int choice;

        do {
            System.out.println("\nShape Area Calculator");
            System.out.println("1. Circle");
            System.out.println("2. Rectangle");
            System.out.println("3. Triangle");
            System.out.println("4. Rhombus");
            System.out.println("5. Parallelogram");
            System.out.println("6. Trapezium");
            System.out.println("7. Equilateral Triangle");
            System.out.println("8. Sector");
            System.out.println("9. Exit");
            System.out.print("Enter your choice: ");

            choice = scanner.nextInt();
            Shape shape = null;

            switch (choice) {
                case 1 -> {
                    System.out.print("Enter radius: ");
                    shape = new Circle(scanner.nextDouble());
                }
                case 2 -> {
                    System.out.print("Enter length: ");
                    double length = scanner.nextDouble();
                    System.out.print("Enter width: ");
                    shape = new Rectangle(length, scanner.nextDouble());
                }
                case 3 -> {
                    System.out.print("Enter base: ");
                    double base = scanner.nextDouble();
```

```
                    System.out.print("Enter height: ");
                    shape = new Triangle(base, scanner.nextDouble());
                }
                case 4 -> {
                    System.out.print("Enter first diagonal: ");
                    double d1 = scanner.nextDouble();
                    System.out.print("Enter second diagonal: ");
                    shape = new Rhombus(d1, scanner.nextDouble());
                }
                case 5 -> {
                    System.out.print("Enter base: ");
                    double pBase = scanner.nextDouble();
                    System.out.print("Enter height: ");
                    shape = new Parallelogram(pBase, scanner.nextDouble())
                        ;
                }
                case 6 -> {
                    System.out.print("Enter first base: ");
                    double b1 = scanner.nextDouble();
                    System.out.print("Enter second base: ");
                    double b2 = scanner.nextDouble();
                    System.out.print("Enter height: ");
                    shape = new Trapezium(b1, b2, scanner.nextDouble());
                }
                case 7 -> {
                    System.out.print("Enter side length: ");
                    shape = new EquilateralTriangle(scanner.nextDouble());
                }
                case 8 -> {
                    System.out.print("Enter radius: ");
                    double radius = scanner.nextDouble();
                    System.out.print("Enter angle (degrees): ");
                    shape = new Sector(radius, scanner.nextDouble());
                }
                case 9 -> System.out.println("Exiting program...");
                default -> System.out.println("Invalid choice! Please try
                    again.");
            }

            if(shape != null) {
                shape.displayArea();
            }

        } while(choice != 9);

        scanner.close();
    }
}
```

2.2 Program Output

```
Shape Area Calculator
1. Circle
2. Rectangle
3. Triangle
4. Rhombus
5. Parallelogram
6. Trapezium
7. Equilateral Triangle
8. Sector
9. Exit
Enter your choice: 1
Enter radius: 5
Area: 78.54

Shape Area Calculator
1. Circle
2. Rectangle
3. Triangle
4. Rhombus
5. Parallelogram
6. Trapezium
7. Equilateral Triangle
8. Sector
9. Exit
Enter your choice: 2
Enter length: 8
Enter width: 4.5
Area: 36.00

Shape Area Calculator
1. Circle
2. Rectangle
3. Triangle
4. Rhombus
5. Parallelogram
6. Trapezium
7. Equilateral Triangle
8. Sector
9. Exit
Enter your choice: 3
Enter base: 10
Enter height: 6
Area: 30.00

Shape Area Calculator
1. Circle
2. Rectangle
3. Triangle
4. Rhombus
5. Parallelogram
6. Trapezium
7. Equilateral Triangle
8. Sector
9. Exit
Enter your choice: 6
Enter first base: 5
```

```
Enter second base: 7
Enter height: 4
Area: 24.00

Shape Area Calculator
1. Circle
2. Rectangle
3. Triangle
4. Rhombus
5. Parallelogram
6. Trapezium
7. Equilateral Triangle
8. Sector
9. Exit
Enter your choice: 7
Enter side length: 6
Area: 15.59

Shape Area Calculator
1. Circle
2. Rectangle
3. Triangle
4. Rhombus
5. Parallelogram
6. Trapezium
7. Equilateral Triangle
8. Sector
9. Exit
Enter your choice: 8
Enter radius: 10
Enter angle (degrees): 60
Area: 52.36

Shape Area Calculator
1. Circle
2. Rectangle
3. Triangle
4. Rhombus
5. Parallelogram
6. Trapezium
7. Equilateral Triangle
8. Sector
9. Exit
Enter your choice: 12
Invalid choice! Please try again.

Shape Area Calculator
1. Circle
2. Rectangle
3. Triangle
4. Rhombus
5. Parallelogram
6. Trapezium
7. Equilateral Triangle
8. Sector
9. Exit
Enter your choice: 9
Exiting program...
```

3 Hospital management using multilevel inheritance

Write a program in Java for Hospital management using Multilevel inheritance with at least 3 classes all classes must have two-member variable and two -member method.

3.1 Java Implementation

```
import java.util.ArrayList;
import java.util.Scanner;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.format.DateTimeParseException;

class Person {
    protected String name;
    protected int age;

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public void displayDetails() {
        System.out.println("Name: " + name + ", Age: " + age);
    }
}

class Patient extends Person {
    protected String diagnosis;
    protected String attendingDoctor;

    public Patient(String name, int age, String diagnosis, String
       attendingDoctor) {
        super(name, age);
        this.diagnosis = diagnosis;
        this.attendingDoctor = attendingDoctor;
    }

    public void updateDiagnosis(String newDiagnosis) {
        this.diagnosis = newDiagnosis;
        System.out.println("Diagnosis updated for " + name);
    }

    public void displayMedicalInformation() {
        System.out.println("Diagnosis: " + diagnosis + ", Doctor: " +
           attendingDoctor);
    }
}

class InPatient extends Patient {
    private final String admissionDate;
    private int roomNumber;
```

```
    public InPatient(String name, int age, String diagnosis, String
        attendingDoctor,
                     String admissionDate, int roomNumber) {
        super(name, age, diagnosis, attendingDoctor);
        this.admissionDate = admissionDate;
        this.roomNumber = roomNumber;
    }

    public void assignRoom(int newRoom) {
        this.roomNumber = newRoom;
        System.out.println("Room assigned for " + name);
    }

    public void displayAdmissionDetails() {
        System.out.println("Admission Date: " + admissionDate + ", Room: "
            + roomNumber);
    }
}

public class HospitalManagement {
    private static final ArrayList<InPatient> patients = new ArrayList<>()
        ;
    private static final Scanner sc = new Scanner(System.in);

    public static void main(String[] args) {
        boolean running = true;

        while(running) {
            System.out.println("\n===== Hospital Management System =====")
                ;
            System.out.println("1. Admit New Patient");
            System.out.println("2. View All Patients");
            System.out.println("3. Update Diagnosis");
            System.out.println("4. Assign New Room");
            System.out.println("5. Discharge Patient");
            System.out.println("6. Exit");
            System.out.print("Enter your choice (1-6): ");

            int choice = getIntInput(1, 6);

            switch(choice) {
                case 1 -> admitPatient();
                case 2 -> viewAllPatients();
                case 3 -> updateDiagnosis();
                case 4 -> assignNewRoom();
                case 5 -> dischargePatient();
                case 6 -> running = confirmExit();
            }
        }
        System.out.println("Program exited. Thank you!");
        sc.close();
    }
```

```
    private static void admitPatient() {
        System.out.println("\n=== New Patient Admission ===");

        System.out.print("Patient Name: ");
        String name = sc.nextLine();

        System.out.print("Age: ");
        int age = getIntInput(0, 120);

        System.out.print("Diagnosis: ");
        String diagnosis = sc.nextLine();

        System.out.print("Attending Doctor (Dr.): ");
        String doctor = sc.nextLine();

        System.out.print("Admission Date (DD/MM/YYYY): ");
        String date = getValidDate();

        System.out.print("Room Number: ");
        int room = getIntInput(1, 1000);

        patients.add(new InPatient(name, age, diagnosis, doctor, date,
            room));
        System.out.println("Patient admitted successfully!");
    }

    private static void viewAllPatients() {
        if(patients.isEmpty()) {
            System.out.println("\nNo patients in records!");
            return;
        }

        System.out.println("\n=== Patient Records ===");
        for(int i = 0; i < patients.size(); i++) {
            System.out.println("\nPatient #" + (i+1));
            InPatient patient = patients.get(i);
            patient.displayDetails();
            patient.displayMedicalInformation();
            patient.displayAdmissionDetails();
            System.out.println("----------------------");
        }
    }

    private static void updateDiagnosis() {
        if(patients.isEmpty()) {
            System.out.println("\nNo patients available!");
            return;
        }

        System.out.println("\n=== Update Diagnosis ===");
        viewBriefList();
        System.out.print("Enter patient number to update: ");
        int index = getIntInput(1, patients.size()) - 1;
```

```
        System.out.print("Enter new diagnosis: ");
        String newDiagnosis = sc.nextLine();

        patients.get(index).updateDiagnosis(newDiagnosis);
    }

    private static void assignNewRoom() {
        if(patients.isEmpty()) {
            System.out.println("\nNo patients available!");
            return;
        }

        System.out.println("\n=== Assign New Room ===");
        viewBriefList();
        System.out.print("Enter patient number to update: ");
        int index = getIntInput(1, patients.size()) - 1;

        System.out.print("Enter new room number: ");
        int newRoom = getIntInput(1, 1000);

        patients.get(index).assignRoom(newRoom);
    }

    private static void dischargePatient() {
        if(patients.isEmpty()) {
            System.out.println("\nNo patients available!");
            return;
        }

        System.out.println("\n=== Discharge Patient ===");
        viewBriefList();
        System.out.print("Enter patient number to discharge: ");
        int index = getIntInput(1, patients.size()) - 1;

        InPatient dischargedPatient = patients.remove(index);
        System.out.println("\nPatient " + dischargedPatient.name + "
           discharged successfully!");
        System.out.println("Discharge Summary:");
        dischargedPatient.displayDetails();
        dischargedPatient.displayMedicalInformation();
        dischargedPatient.displayAdmissionDetails();
    }

    private static boolean confirmExit() {
        System.out.print("\nAre you sure you want to exit? (yes/no): ");
        return !sc.nextLine().equalsIgnoreCase("yes");
    }
    private static int getIntInput(int min, int max) {
        while(true) {
            try {
                int input = Integer.parseInt(sc.nextLine());
                if(input >= min && input <= max) return input;
```

```
                System.out.printf("Please enter between %d-%d: ", min, max
                    );
            } catch(NumberFormatException e) {
                System.out.print("Invalid input. Enter a number: ");
            }
        }
    }
    private static void viewBriefList() {
        for(int i = 0; i < patients.size(); i++) {
            System.out.println("Patient #" + (i+1) + ": " + patients.get(i
                ).name);
        }
    }
    private static String getValidDate() {
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd/MM/
            yyyy");
        while(true) {
            try {
                String dateInput = sc.nextLine();
                LocalDate.parse(dateInput, formatter);
                return dateInput;
            } catch (DateTimeParseException e) {
                System.out.println("Invalid date format! Please use DD/MM/
                    YYYY format.");
            }
        }
    }
}
```

3.2 Program Output

```
===== Hospital Management System =====
1. Admit New Patient
2. View All Patients
3. Update Diagnosis
4. Assign New Room
5. Discharge Patient
6. Exit
Enter your choice (1-6): 1

=== New Patient Admission ===
Patient Name: John Doe
Age: 45
Diagnosis: Pneumonia
Attending Doctor (Dr.): Smith
Admission Date (DD/MM/YYYY): 15/05/2024
Room Number: 205
Patient admitted successfully!

===== Hospital Management System =====
1. Admit New Patient
2. View All Patients
3. Update Diagnosis
4. Assign New Room
```

```
5. Discharge Patient
6. Exit
Enter your choice (1-6): 1

=== New Patient Admission ===
Patient Name: Jane Smith
Age: 32
Diagnosis: Fractured Arm
Attending Doctor (Dr.): Johnson
Admission Date (DD/MM/YYYY): 16/05/2024
Room Number: 312
Patient admitted successfully!

===== Hospital Management System =====
1. Admit New Patient
2. View All Patients
3. Update Diagnosis
4. Assign New Room
5. Discharge Patient
6. Exit
Enter your choice (1-6): 2

=== Patient Records ===

Patient #1
Name: John Doe, Age: 45
Diagnosis: Pneumonia, Doctor: Smith
Admission Date: 15/05/2024, Room: 205
----------------------

Patient #2
Name: Jane Smith, Age: 32
Diagnosis: Fractured Arm, Doctor: Johnson
Admission Date: 16/05/2024, Room: 312
----------------------

===== Hospital Management System =====
1. Admit New Patient
2. View All Patients
3. Update Diagnosis
4. Assign New Room
5. Discharge Patient
6. Exit
Enter your choice (1-6): 3

=== Update Diagnosis ===
Patient #1: John Doe
Patient #2: Jane Smith
Enter patient number to update: 1
Enter new diagnosis: Severe Bronchitis
Diagnosis updated for John Doe

===== Hospital Management System =====
1. Admit New Patient
2. View All Patients
3. Update Diagnosis
4. Assign New Room
5. Discharge Patient
6. Exit
```

```
Enter your choice (1-6): 4

=== Assign New Room ===
Patient #1: John Doe
Patient #2: Jane Smith
Enter patient number to update: 2
Enter new room number: 410
Room assigned for Jane Smith

===== Hospital Management System =====
1. Admit New Patient
2. View All Patients
3. Update Diagnosis
4. Assign New Room
5. Discharge Patient
6. Exit
Enter your choice (1-6): 5

=== Discharge Patient ===
Patient #1: John Doe
Patient #2: Jane Smith
Enter patient number to discharge: 1

Patient John Doe discharged successfully!
Discharge Summary:
Name: John Doe, Age: 45
Diagnosis: Severe Bronchitis, Doctor: Smith
Admission Date: 15/05/2024, Room: 205

===== Hospital Management System =====
1. Admit New Patient
2. View All Patients
3. Update Diagnosis
4. Assign New Room
5. Discharge Patient
6. Exit
Enter your choice (1-6): 2

=== Patient Records ===

Patient #1
Name: Jane Smith, Age: 32
Diagnosis: Fractured Arm, Doctor: Johnson
Admission Date: 16/05/2024, Room: 410
----------------------

===== Hospital Management System =====
1. Admit New Patient
2. View All Patients
3. Update Diagnosis
4. Assign New Room
5. Discharge Patient
6. Exit
Enter your choice (1-6): 6

Are you sure you want to exit? (yes/no): yes
Program exited. Thank you!
```

4 Library management using multilevel inheritance

Write a program in Java for Library management using Multilevel inheritance with at least 3 classes all classes must have two-member variable and two -member method.

4.1 Java Implementation

```
import java.util.ArrayList;
import java.util.Scanner;

class LibraryItem {
    protected String title;
    protected boolean isAvailable;

    public LibraryItem(String title) {
        this.title = title;
        this.isAvailable = true;
    }

    public void checkOut() {
        if (isAvailable) {
            isAvailable = false;
            System.out.println(title + " checked out successfully");
        } else {
            System.out.println(title + " is already checked out");
        }
    }

    public void returnItem() {
        if (!isAvailable) {
            isAvailable = true;
            System.out.println(title + " returned successfully");
        }
        else {
            System.out.println(title + " was not checked out");
        }
    }
}

class Book extends LibraryItem {
    protected String author;
    public Book(String title, String author) {
        super(title);
        this.author = author;
    }
    public void displayInfo() {
        System.out.println(title);
        System.out.println("Author: " + author);
    }
}

class ComputerScienceBook extends Book {
    private final String specialization;
```

```
    private final int edition;
    private final String isbn;
    public ComputerScienceBook(String title, String author, String isbn,
                               String specialization, int edition) {
        super(title, author);
        this.specialization = specialization;
        this.edition = edition;
        this.isbn = isbn;
    }
    @Override
    public void displayInfo() {
        super.displayInfo();
        System.out.println("Specialization: " + specialization);
        System.out.println("Edition: " + edition);
        System.out.println("ISBN: " + isbn);
    }
}

public class LibraryManagementSystem {
    private static final ArrayList<LibraryItem> library = new ArrayList
       <>();
    private static final Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {

        library.add(new ComputerScienceBook(
                "Hands-On Machine Learning with Scikit-Learn, Keras, and
                   TensorFlow: Concepts, Tools, and Techniques to Build
                   Intelligent Systems, Third Edition",
                "Aurélien Géron",
                "1098125975",
                "Machine Learning",
                3
        ));

        library.add(new ComputerScienceBook(
                "Data Science From Scratch",
                "Joel Grus",
                "1492041130",
                "Data Science",
                2
        ));

        library.add(new ComputerScienceBook(
                "Python Data Science Handbook",
                "Jake VanderPlas",
                "1098121228",
                "Data Science",
                2
        ));

        library.add(new ComputerScienceBook(
                "SQL, PL/SQL, The programming language of Oracle",
                "Ivan Bayross",
```

```
                "8176560724",
                "Databases",
                1
        ));

        library.add(new ComputerScienceBook(
                "Digital Image Processing",
                "Rafael C. Gonzalez, Richard E. Woods",
                "9353062983",
                "Image Processing",
                4
        ));

        library.add(new ComputerScienceBook(
                "Let Us C",
                "Yashavant Kanetkar",
                "8183331637",
                "Programming",
                16
        ));

        library.add(new ComputerScienceBook(
                "Data Structures using C",
                "Reema Thareja",
                "0198099304",
                "Data Structures",
                2
        ));

        library.add(new ComputerScienceBook(
                "Computer Organization and Architecture",
                "William Stallings",
                "1292096853",
                "Computer Architecture",
                10
        ));

        library.add(new ComputerScienceBook(
                "An Introduction to Formal Languages and Automata",
                "Peter Linz",
                "1284077241",
                "Formal Languages",
                6
        ));

        library.add(new ComputerScienceBook(
                "Operating System Concepts",
                "Abraham Silberschatz, Peter B. Galvin, Greg Gagne",
                "1119800366",
                "Operating Systems",
                10
        ));

        library.add(new ComputerScienceBook(
```

```
                "Data Communication and Networking",
                "Behrouz A. Forouzan",
                "0073376221",
                "Networking",
                5
        ));

        library.add(new ComputerScienceBook(
                "Discrete Mathematics",
                "Kenneth H. Rosen",
                "125967651X",
                "Mathematics",
                7
        ));

        library.add(new ComputerScienceBook(
                "Java: The Complete Reference",
                "Herbert Schildt",
                "1260440230",
                "Programming",
                11
        ));

        while (true) {
            System.out.println("\n===== Computer Science Library =====");
            System.out.println("1. Show All Books");
            System.out.println("2. Check Out Book");
            System.out.println("3. Return Book");
            System.out.println("4. Exit");
            System.out.print("Enter choice (1-4): ");

            int choice = getIntInput(1, 4);

            System.out.println();

            switch (choice) {
                case 1 -> displayBooks();
                case 2 -> checkOutBook();
                case 3 -> returnBook();
                case 4 -> {
                    System.out.println("Exiting Program....");
                    scanner.close();
                    System.exit(0);
                }
            }
        }
    }

    private static void displayBooks() {
        System.out.println("\nAvailable Computer Science Books:\n");
        for (int i = 0; i < library.size(); i++) {
            LibraryItem item = library.get(i);
            System.out.print((i+1) + ". ");
```

```
            if (item instanceof ComputerScienceBook csb) {
                csb.displayInfo();
                System.out.println("Status: " + (csb.isAvailable ? "
                    Available" : "Checked Out"));
            }
            System.out.println("----------------------");
        }
    }

    private static void checkOutBook() {
        viewAllBooksInBrief();
        System.out.print("\nEnter book number to check out: ");
        int index = getIntInput(1, library.size()) - 1;
        library.get(index).checkOut();
    }

    private static void returnBook() {
        viewAllBooksInBrief();
        System.out.print("\nEnter book number to return: ");
        int index = getIntInput(1, library.size()) - 1;
        LibraryItem item = library.get(index);

        if (!item.isAvailable) {  // Additional check in UI
            item.returnItem();
        } else {
            System.out.println("This book wasn't checked out!");
        }
    }

    private static int getIntInput(int min, int max) {
        while (true) {
            try {
                int input = Integer.parseInt(scanner.nextLine());
                if (input >= min && input <= max) return input;
                System.out.printf("Please enter between %d-%d: ", min, max
                    );
            } catch (NumberFormatException e) {
                System.out.print("Invalid input. Enter a number: ");
            }
        }
    }

    private static void viewAllBooksInBrief() {
        for (int i = 0; i < library.size(); i++) {
            LibraryItem item = library.get(i);
            String status = item.isAvailable ? "Available" : "Unavailable"
                ;
            System.out.println((i+1) + ". " + item.title + "\n   Status: "
                + status);
        }
    }
}
```

4.2 Program Output

```
===== Computer Science Library =====
1. Show All Books
2. Check Out Book
3. Return Book
4. Exit
Enter choice (1-4): 1

Available Computer Science Books:

1. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow:
   Concepts, Tools, and Techniques to Build Intelligent Systems, Third Edition
Author: Aurélien  Géron
Specialization: Machine Learning
Edition: 3
ISBN: 1098125975
Status: Available
----------------------
2. Data Science From Scratch
Author: Joel Grus
Specialization: Data Science
Edition: 2
ISBN: 1492041130
Status: Available
----------------------
3. Python Data Science Handbook
Author: Jake VanderPlas
Specialization: Data Science
Edition: 2
ISBN: 1098121228
Status: Available
----------------------
... [Additional books displayed] ...
13. Java: The Complete Reference
Author: Herbert Schildt
Specialization: Programming
Edition: 11
ISBN: 1260440230
Status: Available
----------------------

===== Computer Science Library =====
1. Show All Books
2. Check Out Book
3. Return Book
4. Exit
Enter choice (1-4): 2

1. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow:
   Concepts, Tools, and Techniques to Build Intelligent Systems, Third Edition
   Status: Available
2. Data Science From Scratch
   Status: Available
3. Python Data Science Handbook
   Status: Available
... [Additional books listed] ...
13. Java: The Complete Reference
```

```
   Status: Available

Enter book number to check out: 1
Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts,
Tools, and Techniques to Build Intelligent Systems, Third Edition checked out
successfully

===== Computer Science Library =====
1. Show All Books
2. Check Out Book
3. Return Book
4. Exit
Enter choice (1-4): 1

Available Computer Science Books:

1. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow:
   Concepts, Tools, and Techniques to Build Intelligent Systems, Third Edition
Author: Aurélien Géron
Specialization: Machine Learning
Edition: 3
ISBN: 1098125975
Status: Checked Out
----------------------
2. Data Science From Scratch
Author: Joel Grus
Specialization: Data Science
Edition: 2
ISBN: 1492041130
Status: Available
----------------------
... [Additional books displayed] ...
13. Java: The Complete Reference
Author: Herbert Schildt
Specialization: Programming
Edition: 11
ISBN: 1260440230
Status: Available
----------------------

===== Computer Science Library =====
1. Show All Books
2. Check Out Book
3. Return Book
4. Exit
Enter choice (1-4): 3

1. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow:
   Concepts, Tools, and Techniques to Build Intelligent Systems, Third Edition
   Status: Checked Out
2. Data Science From Scratch
   Status: Available
3. Python Data Science Handbook
   Status: Available
... [Additional books listed] ...
13. Java: The Complete Reference
   Status: Available

Enter book number to return: 1
```

```
Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow:
Concepts, Tools, and Techniques to Build Intelligent Systems, Third Edition

===== Computer Science Library =====
1. Show All Books
2. Check Out Book
3. Return Book
4. Exit
Enter choice (1-4): 2

1. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow:
   Concepts, Tools, and Techniques to Build Intelligent Systems, Third Edition
   Status: Available
2. Data Science From Scratch
   Status: Available
... [Additional books listed] ...

Enter book number to check out: 13
Java: The Complete Reference checked out successfully

===== Computer Science Library =====
1. Show All Books
2. Check Out Book
3. Return Book
4. Exit
Enter choice (1-4): 3

Enter book number to return: 2
Data Science From Scratch was not checked out

===== Computer Science Library =====
1. Show All Books
2. Check Out Book
3. Return Book
4. Exit
Enter choice (1-4): 4

Exiting Program....
```

5 Railway management using multilevel inheritance

Write a program in Java for Railway management using Multilevel inheritance with at least 3 classes all classes must have two-member variable and two -member method.

5.1 Java implementation

```
import java.util.ArrayList;
import java.util.Scanner;

class RailwayEntity {
    protected String name;
    protected String code;

    public RailwayEntity(String name, String code) {
        this.name = name;
        this.code = code;
    }

    public void displayInfo() {
        System.out.println("Entity: " + name + " (Code: " + code + ")");
    }

}

class Train extends RailwayEntity {
    protected String route;
    protected int capacity;

    public Train(String name, String code, String route, int capacity) {
        super(name, code);
        this.route = route;
        this.capacity = capacity;
    }

    public void displayCapacity() {
        System.out.println("Capacity: " + capacity + " passengers");
    }
}

class PassengerTrain extends Train {
    private final int numCoaches;
    private final String amenities;

    public PassengerTrain(String name, String code, String route, int
       capacity,
                          int numCoaches, String amenities) {
        super(name, code, route, capacity);
        this.numCoaches = numCoaches;
        this.amenities = amenities;
    }

    public void bookTicket() {
```

```
        System.out.println("Ticket booked for " + name + " (" + code + ")"
            );
    }

    public void displayAmenities() {
        System.out.println("Amenities: " + amenities);
    }

    public int getNumCoaches() {
        return numCoaches;
    }

}

public class RailwayManagementSystem {
    private static final ArrayList<PassengerTrain> trains = new ArrayList
        <>();
    private static final Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        trains.add(new PassengerTrain("Express 2023", "EXP-2023",
                "Mumbai-Delhi", 500, 20, "AC, WiFi, Pantry"));

        trains.add(new PassengerTrain("Superfast", "SF-456",
                "Chennai-Kolkata", 450, 18, "AC, Charging Points"));

        trains.add(new PassengerTrain("Local Commuter", "LC-789",
                "Bangalore Suburban", 800, 15, "Standard Seating"));

        while(true) {
            System.out.println("\n==== Railway Management System ====");
            System.out.println("1. View All Trains");
            System.out.println("2. Book Ticket");
            System.out.println("3. Show Train Details");
            System.out.println("4. Exit");
            System.out.print("Enter choice (1-4): ");

            int choice = getIntInput(1, 4);

            switch(choice) {
                case 1 -> viewAllTrains();
                case 2 -> bookTicket();
                case 3 -> showTrainDetails();
                case 4 -> {
                    System.out.println("Exiting system. Thank you!");
                    scanner.close();
                    System.exit(0);
                }
            }
        }
    }

    private static void viewAllTrains() {
        System.out.println("\nAvailable Trains:");
```

```
        for(int i = 0; i < trains.size(); i++) {
            System.out.println((i+1) + ". " + trains.get(i).name + " (" +
                trains.get(i).code + ")");
        }
    }

    private static void bookTicket() {
        viewAllTrains();
        System.out.print("Select train to book: ");
        int index = getIntInput(1, trains.size()) - 1;
        trains.get(index).bookTicket();
    }

    private static void showTrainDetails() {
        viewAllTrains();
        System.out.print("Select train to view details: ");
        int index = getIntInput(1, trains.size()) - 1;

        PassengerTrain train = trains.get(index);
        System.out.println("\n===== Train Details =====");
        train.displayInfo();
        train.displayCapacity();
        train.displayAmenities();
        System.out.println("Route: " + train.route);
        System.out.println("Coaches: " + train.getNumCoaches());
    }

    private static int getIntInput(int min, int max) {
        while(true) {
            try {
                int input = Integer.parseInt(scanner.nextLine());
                if(input >= min && input <= max) return input;
                System.out.printf("Please enter between %d-%d: ", min, max
                    );
            } catch(NumberFormatException e) {
                System.out.print("Invalid input. Enter a number: ");
            }
        }
    }
}
```

5.2 Program Output

```
==== Railway Management System ====
1. View All Trains
2. Book Ticket
3. Show Train Details
4. Exit
Enter choice (1-4): 1

Available Trains:
1. Express 2023 (EXP-2023)
```

```
2. Superfast (SF-456)
3. Local Commuter (LC-789)

==== Railway Management System ====
1. View All Trains
2. Book Ticket
3. Show Train Details
4. Exit
Enter choice (1-4): 2

Available Trains:
1. Express 2023 (EXP-2023)
2. Superfast (SF-456)
3. Local Commuter (LC-789)
Select train to book: 1
Ticket booked for Express 2023 (EXP-2023)

==== Railway Management System ====
1. View All Trains
2. Book Ticket
3. Show Train Details
4. Exit
Enter choice (1-4): 3

Available Trains:
1. Express 2023 (EXP-2023)
2. Superfast (SF-456)
3. Local Commuter (LC-789)
Select train to view details: 1

===== Train Details =====
Entity: Express 2023 (Code: EXP-2023)
Capacity: 500 passengers
Amenities: AC, WiFi, Pantry
Route: Mumbai-Delhi
Coaches: 20

==== Railway Management System ====
1. View All Trains
2. Book Ticket
3. Show Train Details
4. Exit
Enter choice (1-4): 4
Exiting system. Thank you!
```

6 Stack

Write a program in Java to implement stack data structure.

6.1 Java implementation

```
import java.util.EmptyStackException;
import java.util.Scanner;

public class Stack {

    private final int[] elements;
    private int top;
    private final int capacity;

    public Stack(int capacity) {
        if (capacity <= 0) {
            throw new IllegalArgumentException("Capacity must be positive!
               ");
        }
        this.capacity = capacity;
        elements = new int[capacity];
        top = -1;
    }

    public void push(int value) {
        if (isFull()) {
            throw new StackOverflowError("Stack is Full!");
        }
        elements[++top] = value;
    }

    public int pop() {
        if(isEmpty()) {
            throw new EmptyStackException();
        }
        return elements[top--];
    }

    public int peek() {
        if(isEmpty()) {
            throw new EmptyStackException();
        }
        return elements[top];
    }

    public boolean isEmpty() {
        return top == -1;
    }

    public boolean isFull() {
        return top == capacity - 1;
    }
```

```
    public int size() {
        return top + 1;
    }

    public void displayElements() {
        if (isEmpty()) {
            throw new EmptyStackException();
        }
        System.out.println("Stack elements (Top to bottom): ");
        for (int i = top; i >= 0; i--) {
            System.out.println(elements[i] + " ");
        }
        System.out.println();

    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter Stack Capacity: ");
        int capacity = sc.nextInt();
        Stack stack = new Stack(capacity);

        while (true) {
            System.out.println("\nStack Operations:");
            System.out.println("1. Push");
            System.out.println("2. Pop");
            System.out.println("3. Peek");
            System.out.println("4. Check if Empty");
            System.out.println("5. Check if Full");
            System.out.println("6. Get Size");
            System.out.println("7. Show all elements");
            System.out.println("8. Exit");
            System.out.println("Enter Choice: ");

            int choice = sc.nextInt();

            try {
                switch (choice) {
                    case 1 -> {
                        System.out.print("Enter a value to push: ");
                        int value = sc.nextInt();
                        stack.push(value);
                        System.out.println("Pushed " + value);
                    }
                    case 2 -> System.out.println("Popped: " + stack.pop())
                        ;
                    case 3 -> System.out.println("Top element: " + stack.
                        peek());
                    case 4 -> System.out.println("Is stack Empty?: " +
                        stack.isEmpty());
                    case 5 -> System.out.println("Is stack full?: " +
                        stack.isFull());
```

```
                    case 6 -> System.out.println("Current Size: " + stack.
                        size() + "\nCapacity: " + capacity);
                    case 7 -> stack.displayElements();
                    case 8 -> {
                        sc.close();
                        System.exit(0);
                    }
                    default -> System.out.println("Invalid Choice!");
                }
            } catch (EmptyStackException e) {
                System.out.println("Error: Stack is empty!");

            } catch (StackOverflowError e) {
                System.out.println("Error: Stack is full!");
            }
        }
    }
}
```

6.2 Program output

```
Enter Stack Capacity:
3

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
1
Enter a value to push: 10
Pushed 10

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
1
Enter a value to push: 20
Pushed 20

Stack Operations:
1. Push
2. Pop
```

```
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
1
Enter a value to push: 30
Pushed 30

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
1
Enter a value to push: 40
Error: Stack is full!

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
7
Stack elements (Top to bottom):
30
20
10


Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
3
Top element: 30

Stack Operations:
1. Push
2. Pop
3. Peek
```

```
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
2
Popped: 30

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
6
Current Size: 2
Capacity: 3

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
4
Is stack Empty?: false

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
5
Is stack full?: false

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Check if Full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
```

8

7 Queue

Write a program in Java to implement queue data structure.

7.1 Java implementation

```java
import java.util.Scanner;

public class Queue {
    private final int[] elements;
    private final int capacity;
    private int front;
    private int rear;
    private int size;

    public Queue(int capacity) {
        this.capacity = capacity;
        elements = new int[capacity];
        front = 0;
        rear = -1;
        size = 0;
    }

    public void enqueue(int value) {
        if(isFull()) {
            throw new IllegalStateException("Queue is full!");
        }
        rear = (rear + 1) % capacity;
        elements[rear] = value;
        size++;
    }

    public int dequeue() throws NoSuchFieldException {
        if(isEmpty()) {
            throw new NoSuchFieldException("Queue is Empty");
        }
        int value = elements[front];
        front = (front + 1) % capacity;
        size--;
        return value;
    }

    public int peek() throws NoSuchFieldException {
        if (isEmpty()) {
            throw new NoSuchFieldException("Queue is Empty");
        }
        return elements[front];
    }

    public boolean isFull() {
        return size == capacity;
    }
```

```
    public boolean isEmpty() {
        return size == 0;
    }

    public void display() {
        if(isEmpty()) {
            System.out.println("Queue is Empty");
            return;
        }
        System.out.println("Queue elements: ");
        for (int i = 0; i < size; i++) {
            int index = (front + i) % capacity;
            System.out.print(elements[index] + " ");
        }
        System.out.println();
    }

    public int getSize() {
        return size;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter queue Capacity: ");
        int capacity = sc.nextInt();
        Queue queue = new Queue(capacity);

        while (true) {
            System.out.println("\nQueue Operations:");
            System.out.println("1. Enqueue");
            System.out.println("2. Dequeue");
            System.out.println("3. Peek");
            System.out.println("4. Check if empty");
            System.out.println("5. Check if full");
            System.out.println("6. Display elements");
            System.out.println("7. Get size");
            System.out.println("8. Exit");
            System.out.print("Enter choice: ");

            int choice = sc.nextInt();

            try {
                switch (choice) {
                    case 1 -> {
                        System.out.print("Enter value to enqueue: ");
                        int value = sc.nextInt();
                        queue.enqueue(value);
                        System.out.println("Enqueued " + value);
                    }
                    case 2 -> System.out.println("Dequeued: " + queue.
                       dequeue());
                    case 3 -> System.out.println("Front element: " + queue
                       .peek());
```

```
                    case 4 -> System.out.println("Queue is empty: " +
                        queue.isEmpty());
                    case 5 -> System.out.println("Queue is full: " + queue
                        .isFull());
                    case 6 -> queue.display();
                    case 7 -> System.out.println("Current size: " + queue.
                        getSize() + "\nCapacity: " + capacity);
                    case 8 -> {
                        sc.close();
                        System.exit(0);
                    }
                    default -> System.out.println("Invalid choice!");
                }
            }catch (Exception e) {
                System.out.println("Error: " + e.getMessage());
            }
        }
    }

}
```

7.2 Program output

```
Enter queue Capacity:
3

Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 1
Enter value to enqueue: 10
Enqueued 10

Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 1
Enter value to enqueue: 20
Enqueued 20

Queue Operations:
1. Enqueue
2. Dequeue
```

```
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 1
Enter value to enqueue: 30
Enqueued 30

Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 1
Enter value to enqueue: 40
Error: Queue is full!

Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 6
Queue elements:
10 20 30

Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 3
Front element: 10

Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 2
Dequeued: 10
```

```
Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 7
Current size: 2
Capacity: 3

Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 4
Queue is empty: false

Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 5
Queue is full: false

Queue Operations:
1. Enqueue
2. Dequeue
3. Peek
4. Check if empty
5. Check if full
6. Display elements
7. Get size
8. Exit
Enter choice: 8
```

8 University admission system using packages

Write a program in Java for University admission system using packages.

8.1 File directory structure

```
com/
└── University/
    ├── UniversityAdmissionSystem.java
    ├── admission/
    │   └── AdmissionProcessor.java
    ├── applicants
    │   └── Applicant.java
    └── courses
        └── Course.java
```

8.2 Java implementation

8.2.1 com/University/UniversityAdmissionSystem.java

```
package com.University;

import com.University.courses.Course;
import com.University.applicants.Applicant;
import com.University.admission.AdmissionProcessor;
import java.util.ArrayList;
import java.util.Scanner;

public class UniversityAdmissionSystem {
    private static final ArrayList<Course> courses = new ArrayList<>();
    private static final ArrayList<Applicant> applicants = new ArrayList
        <>();
    private static final Scanner scanner = new Scanner(System.in);

    // Preloaded courses with fixed 60% minimum grade
    static {
        courses.add(new Course("CS101", "Computer Science", 50, 60.0));
        courses.add(new Course("EE201", "Electrical Engineering", 40,
            60.0));
        courses.add(new Course("MA301", "Mathematics", 35, 60.0));
        courses.add(new Course("PHY401", "Physics", 30, 60.0));
        courses.add(new Course("CHEM501", "Chemistry", 25, 60.0));
    }

    public static void main(String[] args) {
        while(true) {
            System.out.println("\n==== University Admission System ====");
            System.out.println("1. View All Courses");
            System.out.println("2. Register Applicant");
            System.out.println("3. Process Admissions");
            System.out.println("4. View Applicants");
            System.out.println("5. Exit");
```

```
            System.out.print("Enter choice (1-5): ");

            int choice = getIntInput(1, 5);

            switch(choice) {
                case 1 -> viewCourses();
                case 2 -> registerApplicant();
                case 3 -> processAdmissions();
                case 4 -> viewApplicants();
                case 5 -> {
                    System.out.println("Exiting system...");
                    scanner.close();
                    System.exit(0);
                }
            }
        }
    }

    private static void viewCourses() {
        System.out.println("\nAvailable Courses:");
        for(Course c : courses) {
            System.out.printf("%s - %s (Capacity: %d/%d, Minimum Grade:
                60%%)\n",
                    c.getCourseCode(), c.getCourseName(),
                    c.getEnrolled(), c.getCapacity());
        }
    }

    private static void registerApplicant() {
        System.out.print("\nEnter applicant ID: ");
        String id = scanner.nextLine();
        System.out.print("Enter name: ");
        String name = scanner.nextLine();
        System.out.print("Enter grade: ");
        double grade = getDoubleInput(0, 100);

        if(grade < 60) {
            System.out.println("Applicant does not meet minimum grade
                requirement (60%)");
            return;
        }

        Applicant applicant = new Applicant(id, name, grade);
        selectCourseForApplicant(applicant);
        applicants.add(applicant);
        System.out.println("Applicant registered!");
    }

    private static void selectCourseForApplicant(Applicant applicant) {
        viewCourses();
        System.out.print("Select course (1-" + courses.size() + "): ");
        int choice = getIntInput(1, courses.size());
        applicant.selectCourse(courses.get(choice-1));
    }
```

```
    private static void processAdmissions() {
        if(applicants.isEmpty()) {
            System.out.println("\nNo applicants to process!");
            return;
        }

        AdmissionProcessor processor = new AdmissionProcessor();
        int successful = 0;

        for(Applicant a : applicants) {
            if(processor.processApplication(a)) successful++;
        }

        System.out.println("\nProcessed " + applicants.size() + "
           applications");
        System.out.println("Successful admissions: " + successful);
    }

    private static void viewApplicants() {
        if(applicants.isEmpty()) {
            System.out.println("\nNo applicants registered!");
            return;
        }

        System.out.println("\nRegistered Applicants:");
        for(Applicant a : applicants) {
            System.out.printf("%s - %s (Grade: %.1f%%) - Applied to: %s\n"
               ,
                    a.getApplicantId(), a.getName(),
                    a.getGrade(), a.getSelectedCourse().getCourseName());
        }
    }

    private static int getIntInput(int min, int max) {
        while(true) {
            try {
                int input = Integer.parseInt(scanner.nextLine());
                if(input >= min && input <= max) return input;
                System.out.printf("Enter between %d-%d: ", min, max);
            } catch(NumberFormatException e) {
                System.out.print("Invalid input. Enter a number: ");
            }
        }
    }

    private static double getDoubleInput(double min, double max) {
        while(true) {
            try {
                double input = Double.parseDouble(scanner.nextLine());
                if(input >= min && input <= max) return input;
                System.out.printf("Enter between %.1f-%.1f: ", min, max);
            } catch(NumberFormatException e) {
                System.out.print("Invalid input. Enter a number: ");
```

```
            }
        }
    }
}
```

8.2.2 com/Unversity/admission/AdmissionProcessor.java

```
package com.University.admission;

import com.University.courses.Course;
import com.University.applicants.Applicant;

public class AdmissionProcessor {
    public boolean processApplication(Applicant applicant) {
        Course course = applicant.getSelectedCourse();
        if(course != null &&
                applicant.getGrade() >= course.getMinGrade() &&
                course.canEnroll()) {
            course.enrollStudent();
            return true;
        }
        return false;
    }
}
```

8.2.3 com/Unversity/applicants/Applicant.java

```
package com.University.applicants;

import com.University.courses.Course;

public class Applicant {
    private final String applicantId;
    private final String name;
    private final double grade;
    private Course selectedCourse;

    public Applicant(String id, String name, double grade) {
        this.applicantId = id;
        this.name = name;
        this.grade = grade;
    }

    // Getters
    public String getApplicantId() { return applicantId; }
    public String getName() { return name; }
    public double getGrade() { return grade; }
    public Course getSelectedCourse() { return selectedCourse; }

    public void selectCourse(Course course) {
```

```
        this.selectedCourse = course;
    }
}
```

8.2.4 com/Unversity/courses/Course.java

```
package com.University.courses;

public class Course {
    private final String courseCode;
    private final String courseName;
    private final int capacity;
    private int enrolled;
    private final double minGrade;

    public Course(String code, String name, int cap, double minGrade) {
        this.courseCode = code;
        this.courseName = name;
        this.capacity = cap;
        this.minGrade = minGrade;
        this.enrolled = 0;
    }


    public String getCourseCode() { return courseCode; }
    public String getCourseName() { return courseName; }
    public int getCapacity() { return capacity; }
    public int getEnrolled() { return enrolled; }
    public double getMinGrade() { return minGrade; }

    public boolean canEnroll() {
        return enrolled < capacity;
    }

    public void enrollStudent() {
        if(canEnroll()) enrolled++;
    }
}
```

8.3 Program output

```
==== University Admission System ====
1. View All Courses
2. Register Applicant
3. Process Admissions
4. View Applicants
5. Exit
Enter choice (1-5): 1

Available Courses:
CS101 - Computer Science (Capacity: 0/50, Minimum Grade: 60%)
```

```
EE201 - Electrical Engineering (Capacity: 0/40, Minimum Grade: 60%)
MA301 - Mathematics (Capacity: 0/35, Minimum Grade: 60%)
PHY401 - Physics (Capacity: 0/30, Minimum Grade: 60%)
CHEM501 - Chemistry (Capacity: 0/25, Minimum Grade: 60%)

==== University Admission System ====
1. View All Courses
2. Register Applicant
3. Process Admissions
4. View Applicants
5. Exit
Enter choice (1-5): 2

Enter applicant ID: A001
Enter name: Alice
Enter grade: 72.5
Available Courses:
CS101 - Computer Science (Capacity: 0/50, Minimum Grade: 60%)
EE201 - Electrical Engineering (Capacity: 0/40, Minimum Grade: 60%)
MA301 - Mathematics (Capacity: 0/35, Minimum Grade: 60%)
PHY401 - Physics (Capacity: 0/30, Minimum Grade: 60%)
CHEM501 - Chemistry (Capacity: 0/25, Minimum Grade: 60%)
Select course (1-5): 1
Applicant registered!

==== University Admission System ====
1. View All Courses
2. Register Applicant
3. Process Admissions
4. View Applicants
5. Exit
Enter choice (1-5): 2

Enter applicant ID: A002
Enter name: Bob
Enter grade: 58.0
Applicant does not meet minimum grade requirement (60%)

==== University Admission System ====
1. View All Courses
2. Register Applicant
3. Process Admissions
4. View Applicants
5. Exit
Enter choice (1-5): 4

Registered Applicants:
A001 - Alice (Grade: 72.5%) - Applied to: Computer Science

==== University Admission System ====
1. View All Courses
2. Register Applicant
3. Process Admissions
4. View Applicants
5. Exit
Enter choice (1-5): 3

Processed 1 applications
Successful admissions: 1
```

```
==== University Admission System ====
1. View All Courses
2. Register Applicant
3. Process Admissions
4. View Applicants
5. Exit
Enter choice (1-5): 1

Available Courses:
CS101 - Computer Science (Capacity: 1/50, Minimum Grade: 60%)
EE201 - Electrical Engineering (Capacity: 0/40, Minimum Grade: 60%)
MA301 - Mathematics (Capacity: 0/35, Minimum Grade: 60%)
PHY401 - Physics (Capacity: 0/30, Minimum Grade: 60%)
CHEM501 - Chemistry (Capacity: 0/25, Minimum Grade: 60%)

==== University Admission System ====
1. View All Courses
2. Register Applicant
3. Process Admissions
4. View Applicants
5. Exit
Enter choice (1-5): 5
Exiting system...
```

9 Stack using a package and interface

Write a program in Java to implement stack data structure using package and interface.

9.1 File directory structure

```
.
├── MainStackArray.java
└── StackPackage/
    ├── ArrayStack.java
    └── StackInterface.java
```

9.2 Java implementation

9.2.1 MainStackArray.java

```
import StackPackage.StackInterface;
import StackPackage.ArrayStack;
import java.util.EmptyStackException;
import java.util.Scanner;

public class MainStackArray {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter Stack Capacity: ");
        int capacity = sc.nextInt();
        StackInterface stack = new ArrayStack(capacity);

        while (true) {
            System.out.println("\nStack Operations:");
            System.out.println("1. Push");
            System.out.println("2. Pop");
            System.out.println("3. Peek");
            System.out.println("4. Check if empty");
            System.out.println("5. Check if full");
            System.out.println("6. Get Size");
            System.out.println("7. Show all elements");
            System.out.println("8. Exit");
            System.out.println("Enter Choice: ");

            int choice = sc.nextInt();

            try {
                switch (choice) {
                    case 1 -> {
                        System.out.println("Enter value to Push: ");
                        int value = sc.nextInt();
                        stack.push(value);
                        System.out.println("Pushed " + value);
                    }
                    case 2 -> System.out.println("Popped: " + stack.pop())
                        ;
```

```
                    case 3 -> System.out.println("Top element: " + stack.
                        peek());
                    case 4 -> System.out.println("Stack is empty: " +
                        stack.isEmpty());
                    case 5 -> System.out.println("Stack is full: " + stack
                        .isFull());
                    case 6 -> System.out.println("Current Size: " + stack.
                        size() + "\nCapacity: " + capacity);
                    case 7 -> stack.displayElements();
                    case 8 -> {
                        sc.close();
                        System.exit(0);
                    }
                    default -> System.out.println("Invalid choice!");
                }
            }catch (EmptyStackException | StackOverflowError e) {
                System.out.println("Error: " + e.getMessage());
            }
        }
    }
}
```

9.2.2 StackPackage/ArrayStack.java

```
package StackPackage;

import java.util.EmptyStackException;

public class ArrayStack implements StackInterface{

    private final int[] stackArray;
    private int top;
    private final int capacity;

    public ArrayStack(int capacity) {
        if (capacity <= 0) {
            throw new IllegalStateException("Capacity must be Positive!");
        }
        this.capacity = capacity;
        stackArray = new int[capacity];
        top = -1;
    }

    @Override
    public void push(int value) {
        if (isFull()) {
            throw new StackOverflowError("Stack is Full");
        }
        stackArray[++top] = value;
    }

    @Override
```

```
    public int pop() {
        if (isEmpty()) {
            throw new EmptyStackException();
        }
        return stackArray[top--];
    }

    @Override
    public int peek() {
        if (isEmpty()) {
            throw new EmptyStackException();
        }
        return stackArray[top];
    }

    @Override
    public boolean isEmpty() {
        return top == -1;
    }

    @Override
    public boolean isFull() {
        return capacity - 1 == top;
    }

    @Override
    public void displayElements() {
        if (isEmpty()) {
            throw new EmptyStackException();
        }
        System.out.println("Stack elements (Top to bottom): ");
        for (int i = top; i >= 0; i--) {
            System.out.println(stackArray[i] + " ");
        }
        System.out.println();
    }

    @Override
    public int size() {
        return top + 1;
    }
}
```

9.2.3 StackPackage/StackInterface.java

```
package StackPackage;
public interface StackInterface {
    void push(int value);
    int pop();
    int peek();
    boolean isEmpty();
    boolean isFull();
```

```
    void displayElements();
    int size();
}
```

9.3 Program output

```
Enter Stack Capacity: 3

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if empty
5. Check if full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
1
Enter value to Push:
10
Pushed 10

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if empty
5. Check if full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
1
Enter value to Push:
20
Pushed 20

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if empty
5. Check if full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
1
Enter value to Push:
30
Pushed 30

Stack Operations:
1. Push
2. Pop
```

```
3. Peek
4. Check if empty
5. Check if full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
1
Enter value to Push:
40
Error: Stack is Full

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if empty
5. Check if full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
3
Top element: 30

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if empty
5. Check if full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
7
Stack elements (Top to bottom):
30
20
10

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if empty
5. Check if full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
2
Popped: 30

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if empty
```

```
5. Check if full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
4
Stack is empty: false

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if empty
5. Check if full
6. Get Size
7. Show all elements
8. Exit
Enter Choice:
8
```

10 Package and interface with inheritance

Write a program in Java to implement package and interface with inheritance.

10.1 File directory structure

```
.
├── MainAnimalKingdom.java
└── animalKingdom/
    ├── Animal.java
    ├── Bird.java
    ├── Mammal.java
    ├── Reptile.java
    └── Vertebrate.java
```

10.2 Java implementation

10.2.1 `MainAnimalKingdom.java`

```
import animalKingdom.Animal;
import animalKingdom.Mammal;
import animalKingdom.Reptile;
import animalKingdom.Bird;

import java.util.Scanner;

public class MainAnimalKingdom {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        while(true) {
            System.out.println("\nAnimal Category Selection:");
            System.out.println("1. Mammals");
            System.out.println("2. Birds");
            System.out.println("3. Reptiles");
            System.out.println("4. Exit");
            System.out.print("Enter your choice: ");

            int choice = scanner.nextInt();
            scanner.nextLine();

            if(choice == 4) {
                System.out.println("Exiting program...");
                break;
            }

            if(choice < 1 || choice > 3) {
                System.out.println("Invalid choice!");
                continue;
            }

            String prompt = switch (choice) {
```

```
                case 1 -> "Enter the mammal name (e.g., Tiger, Dog): ";
                case 2 -> "Enter the bird name (e.g., Peacock, Eagle): ";
                case 3 -> "Enter the reptile name (e.g., Snake, Lizard): "
                    ;
                default -> "";
            };

            System.out.print(prompt);
            String name = scanner.nextLine();

            Animal animal = createAnimal(choice, name);
            displayAnimalInfo(animal, name);
        }

        scanner.close();
    }

    private static Animal createAnimal(int category, String name) {
        return switch(category) {
            case 1 -> new Mammal(name);
            case 2 -> new Bird(name);
            case 3 -> new Reptile(name);
            default -> null;
        };
    }

    private static void displayAnimalInfo(Animal animal, String name) {
        System.out.println("\n" + name + " Characteristics:");
        System.out.println("Warm Blooded: " + animal.warmBlooded());
        System.out.println("Feeds Milk: " + animal.feedMilk());
        System.out.println("Has Fur/Feathers: " + animal.hasFur());
        System.out.println("Has Lungs: " + animal.hasLungs());
        System.out.println("Four Limb Body Plan: " + animal.
            fourLimbBodyPlan());
        System.out.print("Movement: ");
        animal.move();
        System.out.print("Sound: ");
        animal.makeSound();
        System.out.print("Eating: ");
        animal.eat();
        System.out.print("Sleeping: ");
        animal.sleep();
    }
}
```

10.2.2 animalKingdom/`Animal.java`

```
package animalKingdom;

public interface Animal {
    void eat();
    void sleep();
```

```
    void move();
    void makeSound();
    boolean warmBlooded();
    boolean feedMilk();
    boolean hasFur();
    boolean hasLungs();
    boolean fourLimbBodyPlan();
}
```

10.2.3 animalKingdom/Bird.java

```
package animalKingdom;

import java.util.Set;

public class Bird extends Vertebrate{
    public Bird(String name) {
        super(name);
    }

    @Override
    public void eat() {
        System.out.println(name + " is pecking food");
    }
    @Override
    public void sleep() {
        System.out.println(name + " is roosting");
    }
    @Override
    public void move() {
        final Set<String> FLIGHTLESS_BIRDS = Set.of(
                "ostriches", "emus", "cassowaries", "rheas", "kiwis"
        );
        if (FLIGHTLESS_BIRDS.contains(name.toLowerCase())) {
            System.out.println(name + " is walking/running");
        }
        else if(name.equalsIgnoreCase("penguin")) {
            System.out.println(name + " is swimming/walking/running");
        }
        else {
            System.out.println(name + " is flying");
        }
    }
    @Override public void makeSound() {
        System.out.println(name + " says: Chirp");
    }
    @Override public boolean warmBlooded() {
        return true;
    }
    @Override public boolean feedMilk() {
        return false;
    }
```

```
    @Override public boolean hasFur() {
        return false;
    }
}
```

10.2.4 animalKingdom/Mammal.java

```
package animalKingdom;

import java.util.Set;

public class Mammal extends Vertebrate {
    public Mammal(String name) {
        super(name);
    }

    @Override
    public void eat() {
        System.out.println(name + " is chewing food");
    }

    @Override
    public void sleep() {
        System.out.println(name + " is sleeping");
    }

    @Override
    public void move() {
        System.out.println(name + " is walking/running");
    }

    @Override
    public void makeSound() {
        final Set<String> CANINE = Set.of(
                "dog", "wolf", "jackal", "fox"
        );
        if (CANINE.contains(name.toLowerCase())) {
            System.out.println(name + " says: Growl/Bark/Howl");
        }
        else {
            System.out.println(name + " says: Growl");
        }

    }

    @Override
    public boolean warmBlooded() {
        return true;
    }

    @Override
    public boolean feedMilk() {
```

```
        return !name.equalsIgnoreCase("platypus") && !name.
            equalsIgnoreCase("anteater");
    }

    @Override
    public boolean hasFur() {
        return true;
    }
}
```

10.2.5 animalKingdom/Reptile.java

```
package animalKingdom;

public class Reptile extends Vertebrate{
    public Reptile(String name) {
        super(name);
    }

    @Override
    public void eat() {
        System.out.println(name + " is swallowing food");
    }
    @Override
    public void sleep() {
        System.out.println(name + " is basking");
    }
    @Override
    public void move() {
        System.out.println(name + " is crawling");
    }
    @Override
    public void makeSound() {
        System.out.println(name + " hisses");
    }
    @Override
    public boolean warmBlooded() {
        return false;
    }
    @Override
    public boolean feedMilk() {
        return false;
    }
    @Override
    public boolean hasFur() {
        return false;
    }
}
```

10.2.6 animalKingdom/Vertebrate.java

```java
package animalKingdom;

public abstract class Vertebrate implements Animal{
    protected String name;

    public Vertebrate(String name) {
        this.name = name;
    }

    @Override
    public boolean hasLungs() {
        return true;
    }

    @Override
    public boolean fourLimbBodyPlan() {
        return true;
    }
}
```

10.3 Program output

```
Animal Category Selection:
1. Mammals
2. Birds
3. Reptiles
4. Exit
Enter your choice: 1
Enter the mammal name (e.g., Tiger, Dog): Dog

Dog Characteristics:
Warm Blooded: true
Feeds Milk: true
Has Fur/Feathers: true
Has Lungs: true
Four Limb Body Plan: true
Movement: Dog is walking/running
Sound: Dog says: Growl/Bark/Howl
Eating: Dog is chewing food
Sleeping: Dog is sleeping

Animal Category Selection:
1. Mammals
2. Birds
3. Reptiles
4. Exit
Enter your choice: 2
Enter the bird name (e.g., Peacock, Eagle): Penguin

Penguin Characteristics:
Warm Blooded: true
Feeds Milk: false
Has Fur/Feathers: false
Has Lungs: true
```

```
Four Limb Body Plan: true
Movement: Penguin is swimming/walking/running
Sound: Penguin says: Chirp
Eating: Penguin is pecking food
Sleeping: Penguin is roosting

Animal Category Selection:
1. Mammals
2. Birds
3. Reptiles
4. Exit
Enter your choice: 3
Enter the reptile name (e.g., Snake, Lizard): Snake

Snake Characteristics:
Warm Blooded: false
Feeds Milk: false
Has Fur/Feathers: false
Has Lungs: true
Four Limb Body Plan: true
Movement: Snake is crawling
Sound: Snake hisses
Eating: Snake is swallowing food
Sleeping: Snake is basking

Animal Category Selection:
1. Mammals
2. Birds
3. Reptiles
4. Exit
Enter your choice: 4
Exiting program...
```

11 Create your own exception

Write a program in Java to create your own exception.

11.1 Java implementation

```
import java.util.InputMismatchException;
import java.util.Scanner;

class InvalidAgeException extends Exception {
    public InvalidAgeException(String message) {
        super(message);
    }
}

public class VotingEligibility {
    public static void validateAge(int age) throws InvalidAgeException {
        if (age < 0) {
            throw new InvalidAgeException("Age cannot be Negative!");
        } else if (age < 18) {
            throw new InvalidAgeException("You must be at least 18 years
               old to vote!");
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (true) {
            System.out.println("\n1. Check your voting eligibility!");
            System.out.println("2. Exit");
            System.out.print("Enter your choice: ");

            try {
                int choice = sc.nextInt();

                switch (choice) {
                    case 1:
                        try {
                            System.out.print("Enter your age: ");
                            int age = sc.nextInt();
                            validateAge(age);
                            System.out.println("\nVoting eligibility
                               confirmed!");
                        } catch (InvalidAgeException e) {
                            System.out.println("Error: " + e.getMessage())
                               ;
                        } catch (InputMismatchException e) {
                            System.out.println("Invalid input! Please
                               enter numbers only.");
                            sc.nextLine();
                        }
                        break;
```

```
                    case 2:
                        sc.close();
                        System.out.println("Exiting program...");
                        System.exit(0);

                    default:
                        System.out.println("Invalid choice! Please enter 1
                            or 2.");
                }
            } catch (InputMismatchException e) {
                System.out.println("Invalid input! Please enter numbers
                   only.");
                sc.nextLine();
            }
        }
    }
}
```

11.2 Program output

```
1. Check your voting eligibility!
2. Exit
Enter your choice: 1
Enter your age: 15
Error: You must be at least 18 years old to vote!

1. Check your voting eligibility!
2. Exit
Enter your choice: 1
Enter your age: -5
Error: Age cannot be Negative!

1. Check your voting eligibility!
2. Exit
Enter your choice: 1
Enter your age: abc
Invalid input! Please enter numbers only.

1. Check your voting eligibility!
2. Exit
Enter your choice: 1
Enter your age: 25

Voting eligibility confirmed!

1. Check your voting eligibility!
2. Exit
Enter your choice: 2
Exiting program...
```

12 Nested try-catch

Write a program in Java for nested try catch.

12.1 Java implementation

```
import java.util.InputMismatchException;
import java.util.Scanner;

public class NestedTryCatch {
    public static void main(String[] args) {
        int[] marks = {100, 200, 300};
        Scanner sc = new Scanner(System.in);
        boolean running = true;

        while (running) {
            System.out.println("\n===== Menu =====");
            System.out.println("1. Access Array Element");
            System.out.println("2. Exit");
            System.out.print("Enter your choice: ");

            try {
                int choice = sc.nextInt();
                try {
                    switch (choice) {
                        case 1 -> {
                            boolean validIndex = false;
                            while (!validIndex) {
                                try {
                                    System.out.print("Enter array index
                                       (0-2): ");
                                    int index = sc.nextInt();
                                    try {
                                        System.out.println("Value at index
                                            " + index + ": " + marks[index
                                           ]);
                                        validIndex = true;
                                    } catch (
                                        ArrayIndexOutOfBoundsException e) {
                                        System.out.println("Level 3 Inner
                                           Catch: Invalid index! Maximum
                                           index is 2");
                                    }
                                } catch (InputMismatchException e) {
                                    System.out.println("Level 2 Middle
                                       Catch: Please enter numbers only
                                       for index!");
                                    sc.nextLine();
                                }
                            }
                        }
                        case 2 -> {
                            running = false;
```

```
                            System.out.println("Exiting program...");
                        }
                        default -> System.out.println("Invalid menu choice
                            !");
                    }
                } catch (Exception e) {
                    System.out.println("Level 1 Outer Catch: General error
                        in menu operation");
                }

            } catch (InputMismatchException e) {
                System.out.println("Top-level Catch: Invalid input! Please
                    enter numbers only for menu choice.");
                sc.nextLine(); // Clear invalid input
            }
        }
        sc.close();
    }
}
```

12.2 Program output

```
===== Menu =====
1. Access Array Element
2. Exit
Enter your choice: 1
Enter array index (0-2): 5
Level 3 Inner Catch: Invalid index! Maximum index is 2
Enter array index (0-2): -1
Level 3 Inner Catch: Invalid index! Maximum index is 2
Enter array index (0-2): abc
Level 2 Middle Catch: Please enter numbers only for index!
Enter array index (0-2): 1
Value at index 1: 200

===== Menu =====
1. Access Array Element
2. Exit
Enter your choice: xyz
Top-level Catch: Invalid input! Please enter numbers only for menu choice.

===== Menu =====
1. Access Array Element
2. Exit
Enter your choice: 3
Invalid menu choice!

===== Menu =====
1. Access Array Element
2. Exit
Enter your choice: 2
Exiting program...
```

13 Create multiple threads

Write a program in Java to create multiple threads.

13.1 Java implementation

```
import java.util.Scanner;

public class MultipleThreads {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("How many threads you want run?: ");
        int total = sc.nextInt();
        for (int i = 1; i <= total; i++) {
            Thread thread = new Thread(new MyRunnable(i));
                thread.start();
        }
    }
}
class MyRunnable implements Runnable {
    private final int threadId;
    public MyRunnable(int threadId) {
        this.threadId = threadId;
    }
    @Override
    public void run() {
        System.out.println("Thread " + threadId + " is running");
        try {
            Thread.sleep(1000L * threadId);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        System.out.println("Thread " + threadId + " completed");
    }
}
```

13.2 Program input

```
How many threads you want run?: 3
```

13.3 Program output

```
Thread 1 is running
Thread 2 is running
Thread 3 is running
Thread 1 completed
Thread 2 completed
Thread 3 completed
```

14 Demonstrate priority assignment to multiple threads

Write a program in Java to demonstrate priority assigning to multiple threads.

14.1 Java implementation

```
import java.util.Scanner;

class MyThread extends Thread {
    private final int iterations;

    public MyThread(String name, int iterations) {
        super(name);
        this.iterations = iterations;
    }

    @Override
    public void run() {
        for (int i = 0; i < iterations; i++) {
            System.out.println(this.getName());
        }
    }
}

public class ThreadPriority {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of iterations for all threads: ");
        int iterations = sc.nextInt();

        MyThread t1 = new MyThread("Server (Most Important)", iterations);
        MyThread t2 = new MyThread("Text Editor", iterations);
        MyThread t3 = new MyThread("Background Music", iterations);
        MyThread t4 = new MyThread("Printer", iterations);
        MyThread t5 = new MyThread("Video Game", iterations);

        t1.setPriority(Thread.MAX_PRIORITY);
        t2.setPriority(Thread.NORM_PRIORITY);
        t3.setPriority(Thread.MIN_PRIORITY);
        t4.setPriority(Thread.NORM_PRIORITY);
        t5.setPriority(Thread.MIN_PRIORITY);

        t1.start();
        t2.start();
        t3.start();
        t4.start();
        t5.start();

        sc.close();
    }
}
```

14.2 Program input

```
Enter number of iterations for all threads: 5
```

14.3 Program output

```
Server (Most Important)
Server (Most Important)
Text Editor
Printer
Background Music
Video Game
Server (Most Important)
Printer
Text Editor
Server (Most Important)
Video Game
Background Music
Text Editor
Printer
Server (Most Important)
Background Music
Video Game
Text Editor
Printer
```

15 Producer–consumer via inter-thread communication

Write a program in Java to implement producer-consumer program using inter-thread communication.

15.1 Java implementation

```
import java.util.concurrent.ArrayBlockingQueue;
import java.util.concurrent.BlockingQueue;

class Producer implements Runnable {
    private final BlockingQueue<Integer> sharedQueue;

    public Producer(BlockingQueue<Integer> sharedQueue) {
        this.sharedQueue = sharedQueue;
    }

    @Override
    public void run() {
        try {
            for (int i = 1; i <= 10; i++) {
                System.out.println("Produced: " + i);
                sharedQueue.put(i);
                Thread.sleep(100);
            }
        }catch (InterruptedException e) {
            Thread.currentThread().interrupt();
            System.out.println("Producer Interrupted");
        }
        System.out.println("Producer finished");

    }
}

class Consumer implements Runnable {
    private final BlockingQueue<Integer> sharedQueue;

    public Consumer(BlockingQueue<Integer> sharedQueue) {
        this.sharedQueue = sharedQueue;
    }

    @Override
    public void run() {
        try {
            for(int i = 1; i <= 10; i++) {
                Integer item = sharedQueue.take();
                System.out.println("Consumed: " + item);
                Thread.sleep(100);
            }
        }catch (InterruptedException e) {
            Thread.currentThread().interrupt();
            System.out.println("Consumer interrupted");
        }
```

```
        System.out.println("Consumer finished");
    }
}

public class ProducerConsumer {
    public static void main(String[] args) throws InterruptedException {
        BlockingQueue<Integer> sharedQueue = new ArrayBlockingQueue<>(5);

        Thread producerThread = new Thread(new Producer(sharedQueue));
        Thread consumerThread = new Thread(new Consumer(sharedQueue));

        producerThread.start();
        consumerThread.start();

        // Wait for both threads to finish
        producerThread.join();
        consumerThread.join();

        System.out.println("All tasks completed");
    }
}
```

15.2 Program output

```
Produced: 1
Consumed: 1
Produced: 2
Consumed: 2
Produced: 3
Consumed: 3
Produced: 4
Consumed: 4
Produced: 5
Consumed: 5
Produced: 6
Consumed: 6
Produced: 7
Consumed: 7
Produced: 8
Consumed: 8
Produced: 9
Consumed: 9
Produced: 10
Consumed: 10
Producer finished
Consumer finished
All tasks completed
```

16 Banker's algorithm

Write a program in Java to implement banker-algorithm in order to prevent deadlock.

16.1 Java implementation

```
import java.util.Scanner;

public class BankersAlgorithm {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Get system configuration
        System.out.print("Enter number of processes: ");
        int numProcesses = sc.nextInt();
        System.out.print("Enter number of resources: ");
        int numResources = sc.nextInt();

        // Initialize matrices and vectors
        int[][] max = new int[numProcesses][numResources];
        int[][] allocation = new int[numProcesses][numResources];
        int[] totalResources = new int[numResources];
        int[][] need = new int[numProcesses][numResources];
        boolean[] finish = new boolean[numProcesses];

        // Read maximum matrix
        System.out.println("\nEnter MAX matrix:");
        for(int i=0; i<numProcesses; i++) {
            System.out.print("Process P" + i + ": ");
            for(int j=0; j<numResources; j++) {
                max[i][j] = sc.nextInt();
            }
        }

        // Read allocation matrix
        System.out.println("\nEnter ALLOCATION matrix:");
        for(int i=0; i<numProcesses; i++) {
            System.out.print("Process P" + i + ": ");
            for(int j=0; j<numResources; j++) {
                allocation[i][j] = sc.nextInt();
                if(allocation[i][j] > max[i][j]) {
                    System.out.println("Error: Allocation exceeds maximum
                        claim!");
                    return;
                }
            }
        }

        // Read total system resources
        System.out.print("\nEnter TOTAL system resources: ");
        for(int j=0; j<numResources; j++) {
            totalResources[j] = sc.nextInt();
```

```
        }

        // Calculate available resources
        int[] available = new int[numResources];
        for(int j=0; j<numResources; j++) {
            int allocatedSum = 0;
            for(int i=0; i<numProcesses; i++) {
                allocatedSum += allocation[i][j];
            }
            available[j] = totalResources[j] - allocatedSum;
            if(available[j] < 0) {
                System.out.println("Error: Total resources exceeded by
                   allocations!");
                return;
            }
        }

        // Calculate need matrix
        for(int i=0; i<numProcesses; i++) {
            for(int j=0; j<numResources; j++) {
                need[i][j] = max[i][j] - allocation[i][j];
            }
        }

        // Safety algorithm
        int[] work = available.clone();
        int[] safeSequence = new int[numProcesses];
        int count = 0;

        while(count < numProcesses) {
            boolean found = false;
            for(int i=0; i<numProcesses; i++) {
                if(!finish[i] && isNeedLessThanWork(need[i], work)) {
                    // Add allocation to work
                    for(int j=0; j<numResources; j++) {
                        work[j] += allocation[i][j];
                    }
                    safeSequence[count] = i;
                    finish[i] = true;
                    count++;
                    found = true;
                    break;
                }
            }
            if(!found) {
                System.out.println("\nSystem is in unsafe state!");
                return;
            }
        }

        // Print safe sequence
        System.out.println("\nSystem is in safe state!");
        System.out.print("Safe sequence: ");
        for(int i=0; i<numProcesses; i++) {
```

```
            System.out.print("P" + safeSequence[i]);
            if(i != numProcesses-1) System.out.print(" -> ");
        }
        System.out.println();
    }

    private static boolean isNeedLessThanWork(int[] need, int[] work) {
        for(int j=0; j<need.length; j++) {
            if(need[j] > work[j]) {
                return false;
            }
        }
        return true;
    }
}
```

16.2 Program input

```
Enter number of processes: 5
Enter number of resources: 3

Enter MAX matrix:
Process P0: 7 5 3
Process P1: 3 2 2
Process P2: 9 0 2
Process P3: 2 2 2
Process P4: 4 3 3

Enter ALLOCATION matrix:
Process P0: 0 1 0
Process P1: 2 0 0
Process P2: 3 0 2
Process P3: 2 1 1
Process P4: 0 0 2

Enter TOTAL system resources: 10 5 7
```

16.3 Program output

```
System is in safe state!
Safe sequence: P1 -> P3 -> P4 -> P0 -> P2
```

17 Count vowels in a 2–3 line text

Write a program in Java to count the vowels of a 2-to-3-line text.

17.1 Java implementation

```
import java.util.Scanner;

public class vowelCounter {
    private static final String END_MARKER = ":end";

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        while (true) {
            System.out.println("\n==== Text Analyzer =====");
            System.out.println("1. Start New Analysis");
            System.out.println("2. Exit");
            System.out.print("Enter choice: ");

            String choice = sc.nextLine();

            if (choice.equals("2")) {
                System.out.println("Exiting program...");
                break;
            } else if (choice.equals("1")) {
                analyzeText(sc);
            }
            else {
                System.out.println("Invalid choice! Please enter 1 or 2");
            }
        }
        sc.close();
    }
    private static void analyzeText(Scanner sc) {
        System.out.println("\nEnter your text (type '" + END_MARKER + "'
           on a new line to finish):");

        int lineCount = 0;
        int vowelCount = 0;
        int characterCount = 0;

        while (true) {
            String line = sc.nextLine();

            if (line.equalsIgnoreCase(END_MARKER)) {
                break;
            }
            lineCount++;
            characterCount += line.length();

            // Count vowels in current line
```

```
            for (int i = 0; i < line.length(); i++) {
                char c = Character.toLowerCase(line.charAt(i));
                if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == '
                    u') {
                    vowelCount++;
                }
            }

        }

        System.out.println("\n=== Analysis Results ===");
        System.out.println("Total lines: " + lineCount);
        System.out.println("Total characters: " + characterCount);
        System.out.println("Total vowels: " + vowelCount);
    }
}
```

17.2 Program output

```
==== Text Analyzer =====
1. Start New Analysis
2. Exit
Enter choice: 1

Enter your text (type ':end' on a new line to finish):
This is a simple example.
Java is fun!
Let's write clean code.
:end

=== Analysis Results ===
Total lines: 3
Total characters: 68
Total vowels: 20
```

18 Copy contents of `A.txt` to `B.txt`

Create a file A.txt and copy the content of A.txt to B.txt.

18.1 Java implementation

```
import java.io.*;

public class TextFileCopy {
    public static void main(String[] args) {
        // Create and write to A.txt
        try (BufferedWriter writer = new BufferedWriter(new FileWriter("A.
            txt"))){
             writer.write("This is the content of file A.txt\n");
             writer.write("Second line of text\n");
             writer.write("Third line of text");
             System.out.println("File A.txt created successfully!");
        } catch (IOException e) {
             System.out.println("Error creating A.txt: " + e.getMessage());
        }

        // Copy the contents of A.txt to B.txt
        try (BufferedReader reader = new BufferedReader(new FileReader("A.
            txt"));
              BufferedWriter writer = new BufferedWriter(new FileWriter("B.
                 txt"))) {

             String line;
             while ((line = reader.readLine()) != null) {
                 writer.write(line);
                 writer.newLine(); // maintain line breaks
             }
             System.out.println("Content copied to B.txt successfully!");

        }catch (FileNotFoundException e) {
             System.out.println("File A.txt not found: " + e.getMessage());
        } catch (IOException e) {
             System.out.println("Error creating B.txt: " + e.getMessage());
        }
    }
}
```

18.2 Program output

```
File A.txt created successfully!
Content copied to B.txt successfully!
```

Contents of `A.txt`

```
This is the content of file A.txt
Second line of text
Third line of text
```

Contents of `B.txt`

```
This is the content of file A.txt
Second line of text
Third line of text
```

19 Applet program 1

Write a program in Java to create a banner using Applet.

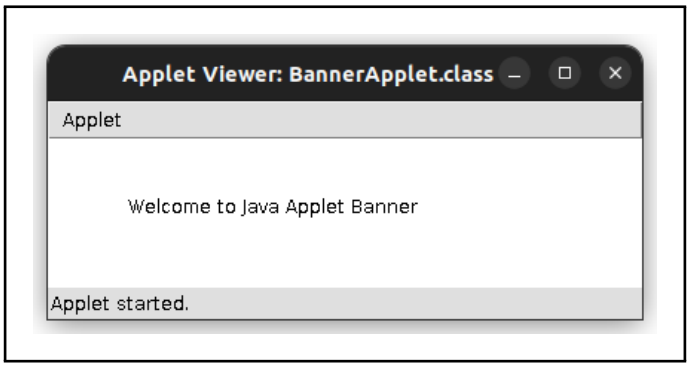
19.1 Java implementation

```
import java.applet.Applet;
import java.awt.Graphics;

public class BannerApplet extends Applet implements Runnable {
    String message = " Welcome to Java Applet Banner ";
    int x = 0;
    Thread t;
    boolean stopFlag;

    public void init() {
        setSize(400, 100); // Set the applet size
        setBackground(java.awt.Color.BLACK);
    }

    public void start() {
        t = new Thread(this);
        stopFlag = false;
        t.start();
    }

    public void run() {
        while (!stopFlag) {
            x -= 5;
            if (x < -message.length() * 10) {
                x = getWidth(); // Reset position to right edge
            }
            repaint();
            try {
                Thread.sleep(100); // Speed of scrolling
            } catch (InterruptedException e) {}
        }
    }

    public void paint(Graphics g) {
        g.setColor(java.awt.Color.GREEN);
        g.drawString(message, x, 50);
    }

    public void stop() {
        stopFlag = true;
        t = null;
    }
}
```

19.2 HTML implementation

```
<html>
<body>
<applet code="BannerApplet.class" width="400" height="100">
</applet>
</body>
</html>
```

19.3 Program output



20 Native Method

Write a Java Program to write a native method.

20.1 File directory structure

```
.
└── NativeMethod/
    ├── NativeExample.java
    └── NativeExample.c
```

20.2 Java implementaion

20.2.1 NativeMethod/NativeExample.java

```java
package NativeMethod;

public class NativeExample {
    // Declare a native method
    public native void printMessage();
    // Static block to load native library
    static {
        System.loadLibrary("NativeExample");
    }
    public static void main(String[] args) {
        new NativeExample().printMessage();
    }
}
```

20.3 C implementation

20.3.1 NativeMethod/NativeExample.c

```c
#include <stdio.h>
#include <jni.h>
#include "NativeMethod_NativeExample.h"  // Generated header

// JNI function name MUST match: Java_[Package]_[Class]_[Method]
JNIEXPORT void JNICALL Java_NativeMethod_NativeExample_printMessage(JNIEnv
    *env, jobject obj) {
    printf("Hello from native code!\n");
}
```

20.4 Native Method Execution

This section demonstrates how to integrate a native C function with a Java program using the Java Native Interface (JNI). The program involves creating a native method in Java, generating a header file, implementing the method in C, compiling it into a shared library, and invoking it from Java.

Step 1: Java Class Definition

```
[NativeMethod/NativeExample.java
```

Step 2: Generating the Header File

Compile the Java file and generate the corresponding C header using the `-h` flag:

```
javac -h . NativeExample.java
```

This generates the file `NativeMethod_NativeExample.h`, which defines the native function signature expected in C.

Step 3: Implementing Native Code in C

```
NativeMethod/NativeExample.c
```

Step 4: Compiling the Native Code into a Shared Library

Compile the C file into a shared library that the JVM can load.

For Windows (with gcc installed):

```
gcc -I"%JAVA_HOME%\include" -I"%JAVA_HOME%\include\win32" -shared -o
   NativeExample.dll NativeExample.c
```

For Linux:

```
gcc -I"$JAVA_HOME/include" -I"$JAVA_HOME/include/linux" -fPIC -shared -o
   libNativeExample.so NativeExample.c
```

Step 5: Running the Java Program

Use the fully-qualified class name and specify the location of the compiled native library:

```
java -Djava.library.path=. NativeMethod.NativeExample
```

20.5 Program output

```
Hello from native code!
```

This confirms the successful invocation of a native method from Java using JNI

21 Implement any five functions of String

21.1 Java implementation

```
import java.util.Scanner;

public class StringOperations {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int choice;
        do {
            System.out.println("\n===== String Operations Menu =====");
            System.out.println("1. Get String Length");
            System.out.println("2. Concatenate Two Strings");
            System.out.println("3. Compare Two Strings");
            System.out.println("4. Check Substring Existence");
            System.out.println("5. Replace characters in String");
            System.out.println("6. Exit");
            System.out.print("Enter your choice (1-6): ");
            choice = sc.nextInt();
            sc.nextLine();
            switch (choice) {
                case 1 -> {
                    System.out.print("Enter a String: ");
                    String str = sc.nextLine();
                    System.out.println("Length: " + str.length());
                }
                case 2 -> {
                    System.out.print("Enter First String: ");
                    String str1 = sc.nextLine();
                    System.out.print("Enter Second String: ");
                    String str2 = sc.nextLine();
                    System.out.println("Concatenated String: " + str1.
                        concat(str2));
                }
                case 3 -> {
                    System.out.print("Enter First String: ");
                    String str1 = sc.nextLine();
                    System.out.print("Enter Second String: ");
                    String str2 = sc.nextLine();
                    if (str1.equals(str2)) {
                        System.out.println("Strings are Equal");
                    } else {
                        System.out.println("Strings are not Equal");
                    }
                }
                case 4 -> {
                    System.out.print("Enter main String: ");
                    String mainStr = sc.nextLine();
                    System.out.print("Enter substring to search: ");
                    String subStr = sc.nextLine();
                    System.out.println("Substring exists: " + mainStr.
                        contains(subStr));
```

```
                }
                case 5 -> {
                    System.out.print("Enter Original String: ");
                    String originalString = sc.nextLine();
                    System.out.print("Enter characters to replace: ");
                    String oldChar = sc.nextLine();
                    System.out.print("Enter replacement character: ");
                    String newChar = sc.nextLine();
                    System.out.println("Modified String: " +
                       originalString.replace(oldChar,newChar));
                }
                case 6 -> System.out.println("Exiting Program...");
                default -> System.out.println("Invalid choice! Please try
                   again.");
            }
        }while (choice != 6);
        sc.close();
    }
}
```

21.2 Program output

```
===== String Operations Menu =====
1. Get String Length
2. Concatenate Two Strings
3. Compare Two Strings
4. Check Substring Existence
5. Replace characters in String
6. Exit
Enter your choice (1-6): 1
Enter a String: Hello World
Length: 11

===== String Operations Menu =====
Enter your choice (1-6): 2
Enter First String: Hello
Enter Second String: World
Concatenated String: HelloWorld

===== String Operations Menu =====
Enter your choice (1-6): 3
Enter First String: Java
Enter Second String: Java
Strings are Equal

===== String Operations Menu =====
Enter your choice (1-6): 4
Enter main String: Artificial Intelligence
Enter substring to search: Intelligence
Substring exists: true

===== String Operations Menu =====
Enter your choice (1-6): 5
Enter Original String: Banana
Enter characters to replace: a
```

```
Enter replacement character: @
Modified String: B@n@n@

===== String Operations Menu =====
Enter your choice (1-6): 6
Exiting Program...
```

22 Demonstrate `InetAddress`

Write a Java Program to demonstrate `InetAddress`.

22.1 Java implementation

```
import java.net.InetAddress;

public class InetAddressDemo {
    public static void main(String[] args) {
        try {
            // 1. Get Local Host information
            InetAddress localHost = InetAddress.getLocalHost();
            System.out.println("Local Host: ");
            System.out.println("Host Name: " + localHost.getHostName());
            System.out.println("IP Address: " + localHost.getHostAddress()
                );
            System.out.println("Canonical Name: " + localHost.
                getCanonicalHostName());
            System.out.println();

            // 2. Get information for a specific Host
            String hostName = "www.google.com";
            InetAddress googleHost = InetAddress.getByName(hostName);
            System.out.println("Information for " + hostName + ": ");
            System.out.println("IP Address: " + googleHost.getHostAddress
                ());
            System.out.println("Canonical Name: " + googleHost.
                getCanonicalHostName());
            System.out.println();

            // 3. Get all IP addresses for a Port
            String multiPHost = "www.facebook.com";
            System.out.println("All IP addresses for " + multiPHost + ":")
                ;
            InetAddress[] allIPs = InetAddress.getAllByName(multiPHost);
            for (InetAddress address: allIPs) {
                System.out.println("IP: " + address.getHostAddress());
            }
            System.out.println();

            // 4. Reverse Lookup (IP to hostname)
            String ipAddress = "8.8.8.8";
            InetAddress dnsServer = InetAddress.getByName(ipAddress);
            System.out.println("Reverse lookup for " + ipAddress + ":");
            System.out.println("Host Name: " + dnsServer.getHostName());
            System.out.println("Canonical Name: " + dnsServer.
                getCanonicalHostName());
        } catch (java.net.UnknownHostException e) {
            throw new RuntimeException(e);
        }
    }
}
```

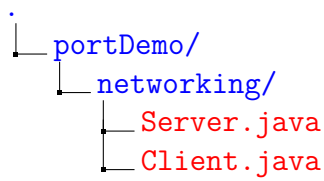
22.2 Program output

```
Local Host:
Host Name: DESKTOP-P2NC49C
IP Address: 192.168.145.1
Canonical Name: DESKTOP-P2NC49C

Information for www.google.com:
IP Address: 142.250.67.68
Canonical Name: tzdela-bf-in-f4.1e100.net

All IP addresses for www.facebook.com:
IP: 157.240.1.35

Reverse lookup for 8.8.8.8:
Host Name: dns.google
Canonical Name: dns.google
```

23 Demonstrate `getPort()`

Write a Java Program to demonstrate `getPort()`.

23.1 File directory structure



23.2 Java implementation

23.2.1 portDemo/networking/`Server.java`

```
package portDemo.networking;

import java.net.ServerSocket;
import java.net.Socket;

public class Server {
    public static void main(String[] args) {
        try {
            ServerSocket serverSocket = new ServerSocket(12345);
            System.out.println("Server started on port: " + serverSocket.
               getLocalPort());

            Socket clientSocket = serverSocket.accept();
            System.out.println("\nClient Connected from:");
            System.out.println("Remote Port: " + clientSocket.getPort());
            System.out.println("Local Port: " + clientSocket.getLocalPort
               ());

            clientSocket.close();
            serverSocket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

23.2.2 portDemo/networking/`Client.java`

```
 package portDemo.networking;

import java.net.Socket;

public class Client {
    public static void main(String[] args) {
        try {
```

```
            Socket socket = new Socket("localHost", 12345);
            System.out.println("Connected to Server:");
            System.out.println("Server Port: " + socket.getPort());
            System.out.println("Client Port: " + socket.getLocalPort());

            socket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

23.3 Socket Communication

The following terminal outputs demonstrate the successful execution of a server and client socket connection in Java. The server listens on port 12345, and the client connects to the same port from a randomly assigned local port.

23.3.1 Server Terminal Output (Before Client Connects)

```
C:\Users\Sayan Dutta\IdeaProjects\Java_Assignments_MSc\src\portDemo\
networking>java Server
Server started on port: 12345
```

23.3.2 Client Terminal Output

```
C:\Users\Sayan Dutta\IdeaProjects\Java_Assignments_MSc\src\portDemo\
networking>java Client
Connected to Server:
Server Port: 12345
Client Port: 52352
```

23.3.3 Updated Server Terminal Output (After Client Connects)

```
Client Connected from:
Remote Port: 52352
Local Port: 12345
```

24 Demonstrate URL

Write a Java Program to demonstrate URL.

24.1 Java implementation

```java
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Scanner;

public class URLDemo {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter URL: ");
        String inputURL = sc.nextLine();
        try {
            URL url = new URL(inputURL);
            System.out.println("=== URL Breakdown ===");
            System.out.println("Protocol: " + url.getProtocol());
            System.out.println("Host: " + url.getHost());
            System.out.println("Port: " + url.getPort());
            System.out.println("Default Port: " + url.getDefaultPort());
            System.out.println("Path: " + url.getPath());
            System.out.println("Query: " + url.getQuery());
            System.out.println("Reference: " + url.getRef());
            System.out.println("Authority: " + url.getAuthority());
            System.out.println("User Info: " + url.getUserInfo());
        } catch (MalformedURLException e) {
            System.out.println("Connection error: " + e.getMessage());
        }
    }
}
```

24.2 Program input

```
Enter URL: https://github.com/Sayan-Dutta-2003
```

24.3 Program output

```
=== URL Breakdown ===
Protocol: https
Host: github.com
Port: -1
Default Port: 443
Path: /Sayan-Dutta-2003
Query: null
Reference: null
Authority: github.com
User Info: null
```

25 Demonstrate Datagram

Write a Java Program to demonstrate datagram.

25.1 File directory structure

```
.
└── Datagram/
    ├── Server.java
    └── Client.java
```

25.2 Java implementation

25.2.1 Datagram/Server.java

```
package Datagram;

import java.io.IOException;
import java.net.*;

public class Server {
    public static void main(String[] args) throws Exception {
        DatagramSocket socket = null;
        try {
            socket = new DatagramSocket(9876);
            System.out.println("Server started on port 9876...");

            byte[] receiveBuffer = new byte[1024];

            while (true) {
                // Receive packet from client
                DatagramPacket receivePacket = new DatagramPacket(
                   receiveBuffer, receiveBuffer.length);
                socket.receive(receivePacket);

                // Process received Data
                String receivedData = new String(receivePacket.getData(),
                   0, receivePacket.getLength());
                System.out.println("Received from Client: " + receivedData
                   );

                // Prepare response
                InetAddress clientAddress = receivePacket.getAddress();

                int clientPort = receivePacket.getPort();
                String responseMessage = "Message received!";
                byte[] sendBuffer = responseMessage.getBytes();

                // Send response back to client
                DatagramPacket sendPacket = new DatagramPacket(sendBuffer,
                    sendBuffer.length, clientAddress,clientPort);
                socket.send(sendPacket);
                System.out.println("Sent response to client");
```

```
            }
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (socket != null && !socket.isClosed()) {
                socket.close();
            }
        }

    }
}
```

25.2.2 Datagram/Client.java

```
package Datagram;

import java.io.IOException;
import java.net.*;


public class Client {
    public static void main(String[] args) {
        DatagramSocket socket = null;
        try {
            socket = new DatagramSocket();
            InetAddress serverAddress = InetAddress.getByName("localhost")
                ;

            // Prepare message to send
            String message = "Hello Server!";
            byte[] sendBuffer = message.getBytes();

            // Send packet to server
            DatagramPacket sendPacket = new DatagramPacket(
                    sendBuffer,
                    sendBuffer.length,
                    serverAddress,
                    9876
            );
            socket.send(sendPacket);
            System.out.println("Sent message to server: " + message);

            // Receive response from server
            byte[] receiveBuffer = new byte[1024];
            DatagramPacket receivePacket = new DatagramPacket(
               receiveBuffer, receiveBuffer.length);
            socket.receive(receivePacket);

            // Process the response
            String receivedResponse = new String(
                    receivePacket.getData(),
                    0,
```

```
                    receivePacket.getLength()
            );
            System.out.println("Received from server: " + receivedResponse
                );
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (socket != null && !socket.isClosed()) {
                socket.close();
            }
        }
    }
}
```

25.3 UDP Socket Execution

The following terminal outputs demonstrate the successful execution of a UDP server and client socket communication in Java. The server listens on port 9876, and the client sends a message which the server receives and acknowledges.

25.3.1 Server Terminal Output (Before Client Connects)

```
C:\Users\Sayan Dutta\IdeaProjects\Java_Assignments_MSc\src\Datagram>
java Server.java
Server started on port 9876...
```

25.3.2 Client Terminal Output

```
C:\Users\Sayan Dutta\IdeaProjects\Java_Assignments_MSc\src\Datagram>
java Client.java
Sent message to server: Hello Server!
Received from server: Message received!
```

25.3.3 Server Terminal Output (After Client Sends Message)

```
C:\Users\Sayan Dutta\IdeaProjects\Java_Assignments_MSc\src\Datagram>
java Server.java
Server started on port 9876...
Received from Client: Hello Server!
Sent response to client
```

26 Keyboard event listener

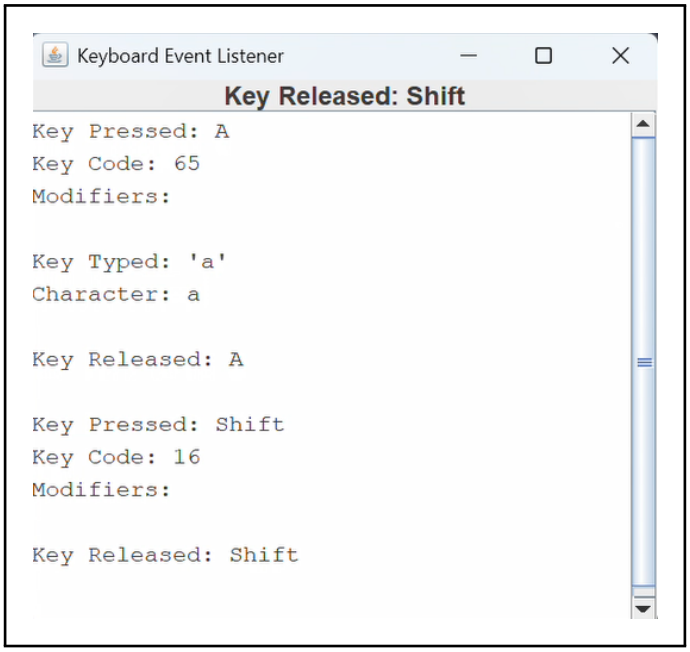
Write a Java programme to implement the keyboard event listener.

26.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class KeyboardEventListener extends JFrame implements KeyListener {
    private JLabel label;
    private JTextArea textArea;

    public KeyboardEventListener() {
        // Set up the frame
        setTitle("Keyboard Event Listener");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());

        // Create components
        label = new JLabel("Press any key...", JLabel.CENTER);
        label.setFont(new Font("Arial", Font.BOLD, 16));

        textArea = new JTextArea();
        textArea.setEditable(false);
        textArea.setFont(new Font("Monospaced", Font.PLAIN, 14));
        JScrollPane scrollPane = new JScrollPane(textArea);

        // Add components to frame
        add(label, BorderLayout.NORTH);
        add(scrollPane, BorderLayout.CENTER);

        // Add key listener to the text area
        textArea.addKeyListener(this);

        // Make sure the text area can receive focus
        textArea.setFocusable(true);
        textArea.requestFocusInWindow();
    }

    // Implement KeyListener methods
    @Override
    public void keyPressed(KeyEvent e) {
        String message = "Key Pressed: " + KeyEvent.getKeyText(e.
            getKeyCode());
        label.setText(message);
        textArea.append(message + "\n");
        textArea.append("Key Code: " + e.getKeyCode() + "\n");
        textArea.append("Modifiers: " + KeyEvent.getKeyModifiersText(e.
            getModifiersEx()) + "\n\n");
    }
```

```
    @Override
    public void keyReleased(KeyEvent e) {
        String message = "Key Released: " + KeyEvent.getKeyText(e.
            getKeyCode());
        label.setText(message);
        textArea.append(message + "\n\n");
    }

    @Override
    public void keyTyped(KeyEvent e) {
        String message = "Key Typed: '" + e.getKeyChar() + "'";
        label.setText(message);
        textArea.append(message + "\n");
        textArea.append("Character: " + e.getKeyChar() + "\n\n");
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            KeyboardEventListener listener = new KeyboardEventListener();
            listener.setVisible(true);
        });
    }
}
```

26.2 Program output



27 Mouse event listener

Write a Java program to implement mouse event listener (click, enter, exited, pressed, release).

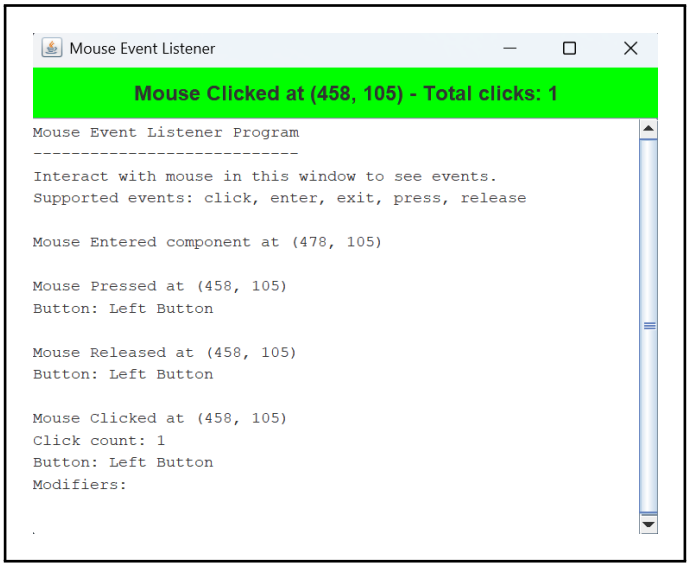
27.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class MouseEventListener extends JFrame implements MouseListener {
    private JLabel statusLabel;
    private JTextArea eventLog;
    private int clickCount = 0;

    public MouseEventListener() {
        setTitle("Mouse Event Listener");
        setSize(500, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());

        // Create components
        statusLabel = new JLabel("Mouse status: Waiting for events...",
            JLabel.CENTER);
        statusLabel.setFont(new Font("Arial", Font.BOLD, 16));
        statusLabel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10,
            10));
        statusLabel.setOpaque(true);
        statusLabel.setBackground(Color.LIGHT_GRAY);

        eventLog = new JTextArea();
        eventLog.setEditable(false);
        eventLog.setFont(new Font("Monospaced", Font.PLAIN, 12));
        JScrollPane scrollPane = new JScrollPane(eventLog);

        // Add components to frame
        add(statusLabel, BorderLayout.NORTH);
        add(scrollPane, BorderLayout.CENTER);

        // Add mouse listener to the frame
        addMouseListener(this);

        // Add mouse listener to the text area as well
        eventLog.addMouseListener(this);

        // Display instructions
        eventLog.append("Mouse Event Listener Program\n");
        eventLog.append("----------------------------\n");
        eventLog.append("Interact with mouse in this window to see events
            .\n");
        eventLog.append("Supported events: click, enter, exit, press,
            release\n\n");
    }
```

```
    // Implement MouseListener methods
    @Override
    public void mouseClicked(MouseEvent e) {
        clickCount++;
        String message = "Mouse Clicked at (" + e.getX() + ", " + e.getY()
            + ")";
        statusLabel.setText(message + " - Total clicks: " + clickCount);
        eventLog.append(message + "\n");
        eventLog.append("Click count: " + e.getClickCount() + "\n");
        eventLog.append("Button: " + getMouseButtonText(e.getButton()) + "
            \n");
        eventLog.append("Modifiers: " + KeyEvent.getKeyModifiersText(e.
            getModifiersEx()) + "\n\n");
    }

    @Override
    public void mouseEntered(MouseEvent e) {
        statusLabel.setText("Mouse Entered component");
        statusLabel.setBackground(Color.GREEN);
        eventLog.append("Mouse Entered component at (" + e.getX() + ", " +
            e.getY() + ")\n\n");
    }

    @Override
    public void mouseExited(MouseEvent e) {
        statusLabel.setText("Mouse Exited component");
        statusLabel.setBackground(Color.RED);
        eventLog.append("Mouse Exited component at (" + e.getX() + ", " +
            e.getY() + ")\n\n");
    }

    @Override
    public void mousePressed(MouseEvent e) {
        statusLabel.setText("Mouse Pressed at (" + e.getX() + ", " + e.
            getY() + ")");
        eventLog.append("Mouse Pressed at (" + e.getX() + ", " + e.getY()
            + ")\n");
        eventLog.append("Button: " + getMouseButtonText(e.getButton()) + "
            \n\n");
    }

    @Override
    public void mouseReleased(MouseEvent e) {
        statusLabel.setText("Mouse Released at (" + e.getX() + ", " + e.
            getY() + ")");
        eventLog.append("Mouse Released at (" + e.getX() + ", " + e.getY()
            + ")\n");
        eventLog.append("Button: " + getMouseButtonText(e.getButton()) + "
            \n\n");
    }

    // Helper method to get mouse button text
    private String getMouseButtonText(int button) {
```

```
        switch (button) {
            case MouseEvent.BUTTON1: return "Left Button";
            case MouseEvent.BUTTON2: return "Middle Button";
            case MouseEvent.BUTTON3: return "Right Button";
            default: return "Unknown Button";
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            MouseEventListener listener = new MouseEventListener();
            listener.setVisible(true);
        });
    }
}
```

27.2 Program output



28 Applet programme 2

Write a Java program to create a child frame window from within an applet.

28.1 Java implementation

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

public class ChildFrameApplet extends Applet implements ActionListener {
    Button openFrameButton;

    public void init() {
        openFrameButton = new Button("Open Child Frame");
        add(openFrameButton);
        openFrameButton.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e) {
        new ChildFrame("Child Frame Window");
    }
}

class ChildFrame extends Frame {
    ChildFrame(String title) {
        super(title);
        setSize(300, 150);
        setLayout(new FlowLayout());
        Label label = new Label("This is a child frame window.");
        add(label);
        setVisible(true);

        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                dispose();
            }
        });
    }
}
```

28.2 HTML implementation

```
<html>
<body>
<applet code="ChildFrameApplet.class" width="300" height="100">
</applet>
</body>
</html>
```

28.3 Program output

29 Applet programme 3

Write a Java program to create a night view of the sky using applet.

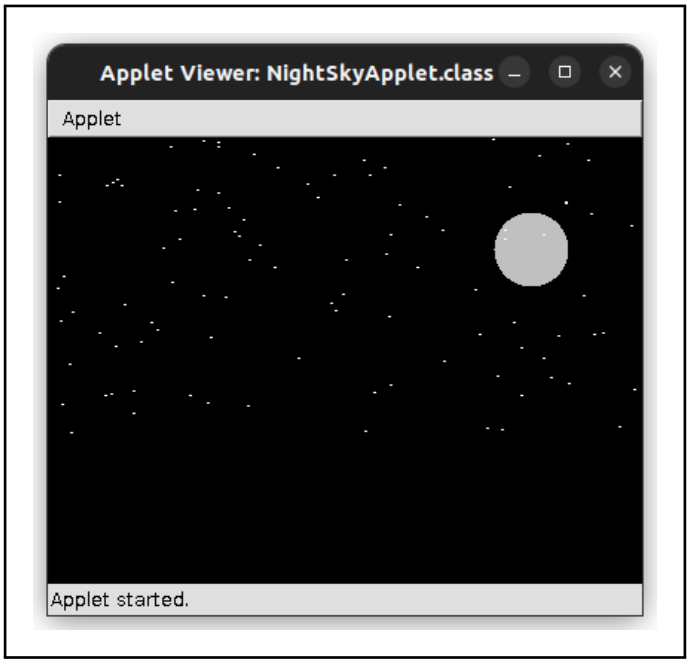
29.1 Java implementation

```
import java.applet.Applet;
import java.awt.*;
import java.util.Random;

public class NightSkyApplet extends Applet {
    public void init() {
        setSize(400, 300);
        setBackground(Color.BLACK);
    }
    public void paint(Graphics g) {
        drawMoon(g);
        drawStars(g, 100);
    }
    // Draw a moon
    private void drawMoon(Graphics g) {
        g.setColor(Color.LIGHT_GRAY);
        g.fillOval(300, 50, 50, 50); // Moon shape
    }
    // Draw stars randomly across the sky
    private void drawStars(Graphics g, int count) {
        g.setColor(Color.WHITE);
        Random rand = new Random();
        for (int i = 0; i < count; i++) {
            int x = rand.nextInt(getWidth());
            int y = rand.nextInt(getHeight() - 100); // Avoid lower part
            g.fillOval(x, y, 2, 2); // Tiny star
        }
    }
}
```

29.2 HTML implementation

```
<html>
<body>
<applet code="NightSkyApplet.class" width="400" height="300">
</applet>
</body>
</html>
```

29.3 Program output



30 Applet programme 4

Write a Java program to create a sand clock using applet.

30.1 Java implementation

```
import java.applet.Applet;
import java.awt.*;
import java.util.Random;

public class AnimatedSandClockApplet extends Applet implements Runnable {
    private static final int NUM_GRAINS = 50;
    private int[] grainX = new int[NUM_GRAINS];
    private int[] grainY = new int[NUM_GRAINS];
    private boolean[] hasFallen = new boolean[NUM_GRAINS];
    private int fallenCount = 0;
    private Thread animator;
    private Random rand = new Random();

    @Override
    public void init() {
        setSize(300, 500);
        // initialize grains at random positions inside the upper triangle
        for (int i = 0; i < NUM_GRAINS; i++) {
            spawnGrainAtTop(i);
            hasFallen[i] = false;
        }
    }

    private void spawnGrainAtTop(int i) {
        // y between top rim (50) and neck (250)
        int y = 50 + rand.nextInt(200);
        // compute x-range at this y within upper triangle
        float frac = (float)(y - 50) / 200; // 0 at y=50, 1 at y=250
        int xLeft = 100 + (int)(50 * frac);
        int xRight = 200 - (int)(50 * frac);
        int x = xLeft + rand.nextInt(xRight - xLeft + 1);
        grainX[i] = x;
        grainY[i] = y;
    }

    @Override
    public void start() {
        animator = new Thread(this);
        animator.start();
    }

    @Override
    public void stop() {
        animator = null;
    }

    @Override
```

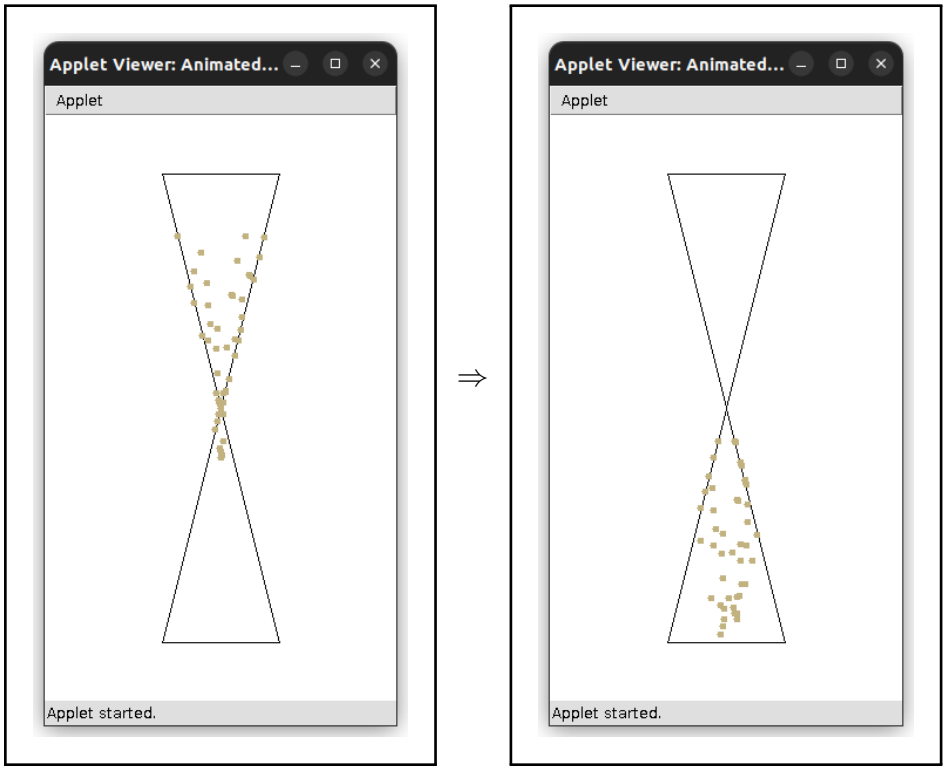
```
    public void run() {
        while (Thread.currentThread() == animator) {
            for (int i = 0; i < NUM_GRAINS; i++) {
                if (!hasFallen[i]) {
                    grainY[i]++;
                    if (grainY[i] > 450) {
                        hasFallen[i] = true;
                        fallenCount++;
                    }
                }
            }
            repaint();
            // stop when all grains have fallen
            if (fallenCount >= NUM_GRAINS) {
                stop();
                break;
            }
            try {
                Thread.sleep(50);
            } catch (InterruptedException e) {
                break;
            }
        }
    }

    @Override
    public void paint(Graphics g) {
        // Clear background
        g.setColor(Color.WHITE);
        g.fillRect(0, 0, getWidth(), getHeight());

        // Draw hourglass outline
        g.setColor(Color.BLACK);
        // Top and bottom rims
        g.drawLine(100, 50, 200, 50);
        g.drawLine(100, 450, 200, 450);
        // Sides down to the neck
        g.drawLine(100, 50, 150, 250);
        g.drawLine(200, 50, 150, 250);
        // Sides up from the neck
        g.drawLine(100, 450, 150, 250);
        g.drawLine(200, 450, 150, 250);

        // Draw all grains that haven't fallen past bottom
        g.setColor(new Color(194, 178, 128));
        for (int i = 0; i < NUM_GRAINS; i++) {
            if (!hasFallen[i]) {
                int x = grainX[i];
                int y = grainY[i];
                // clamp inside shape as before
                if (y < 250) {
                    float frac = (float)(y - 50) / 200;
                    int xLeft = 100 + (int)(50 * frac);
                    int xRight = 200 - (int)(50 * frac);
```

```
                    x = Math.max(xLeft, Math.min(x, xRight));
                } else {
                    float frac = (float)(450 - y) / 200;
                    int xLeft = 100 + (int)(50 * frac);
                    int xRight = 200 - (int)(50 * frac);
                    x = Math.max(xLeft, Math.min(x, xRight));
                }
                g.fillOval(x - 3, y - 3, 6, 6);
            }
        }
    }
}
```

30.2 HTML implementation

```
<html>
  <body>
    <applet code="AnimatedSandClockApplet.class" width="300" height="400">
    </applet>
  </body>
</html>
```

30.3 Program output



31 Bouncing balls

Write a Java program to bounce more than one ball.

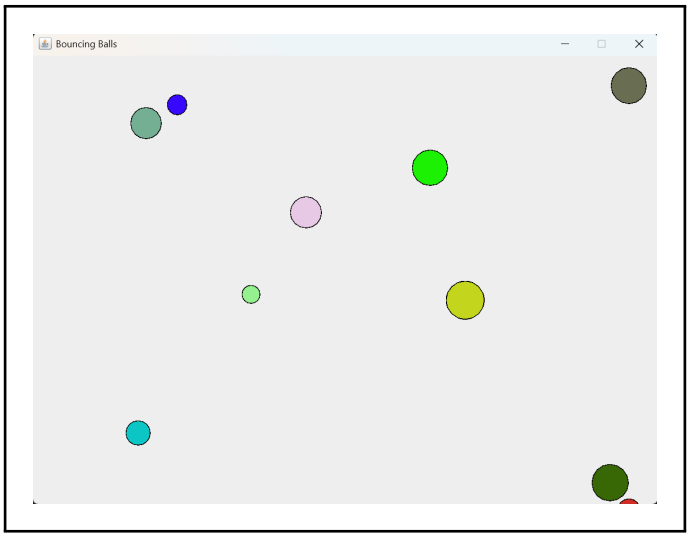
31.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;
import java.util.Random;

public class BouncingBalls extends JPanel {
    private static final int WIDTH = 800;
    private static final int HEIGHT = 600;
    private ArrayList<Ball> balls = new ArrayList<>();
    private Random random = new Random();

    public BouncingBalls() {
        // Create 10 random balls
        for (int i = 0; i < 10; i++) {
            balls.add(createRandomBall());
        }

        // Animation timer
        Timer timer = new Timer(20, new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                updateBalls();
                repaint();
            }
        });
        timer.start();

        // Add mouse listener to create new balls on click
        addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                balls.add(new Ball(e.getX(), e.getY(),
                        random.nextInt(30) + 10,
                        getRandomColor(),
                        random.nextInt(10) - 5,
                        random.nextInt(10) - 5));
            }
        });
    }

    private Ball createRandomBall() {
        int size = random.nextInt(30) + 20;
        return new Ball(
                random.nextInt(WIDTH - size),
                random.nextInt(HEIGHT - size),
                size,
```

```
                getRandomColor(),
                random.nextInt(10) - 5,
                random.nextInt(10) - 5);
    }

    private Color getRandomColor() {
        return new Color(
                random.nextInt(256),
                random.nextInt(256),
                random.nextInt(256));
    }

    private void updateBalls() {
        for (Ball ball : balls) {
            ball.move();

            // Wall collision detection
            if (ball.x < 0 || ball.x > WIDTH - ball.size) {
                ball.dx = -ball.dx;
            }
            if (ball.y < 0 || ball.y > HEIGHT - ball.size) {
                ball.dy = -ball.dy;
            }
        }
    }

    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        for (Ball ball : balls) {
            ball.draw(g);
        }
    }

    private class Ball {
        int x, y;        // Position
        int size;        // Diameter
        Color color;      // Color
        int dx, dy;      // Velocity

        public Ball(int x, int y, int size, Color color, int dx, int dy) {
            this.x = x;
            this.y = y;
            this.size = size;
            this.color = color;
            this.dx = dx;
            this.dy = dy;
        }

        public void move() {
            x += dx;
            y += dy;
        }
```

```
        public void draw(Graphics g) {
            g.setColor(color);
            g.fillOval(x, y, size, size);
            g.setColor(Color.BLACK);
            g.drawOval(x, y, size, size);
        }
    }

    public static void main(String[] args) {
        JFrame frame = new JFrame("Bouncing Balls");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(WIDTH, HEIGHT);
        frame.setResizable(false);
        frame.add(new BouncingBalls());
        frame.setVisible(true);
    }
}
```

31.2 Program output



32 Digital clock

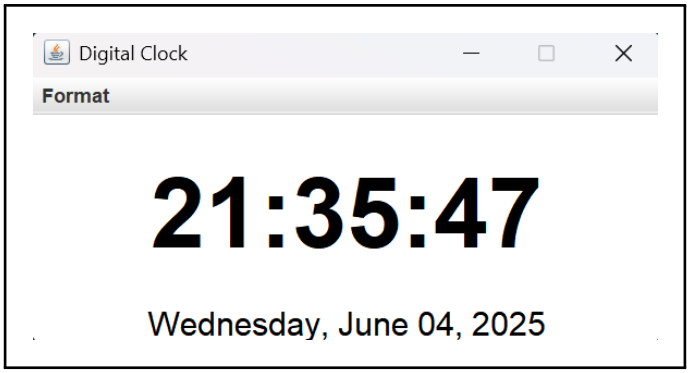
Write a Java program to create a digital clock.

32.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

public class DigitalClock extends JFrame {
    private JLabel timeLabel;
    private JLabel dateLabel;
    private String timeFormat = "HH:mm:ss";
    private boolean is24HourFormat = true;

    public DigitalClock() {
        // Set up the frame
        setTitle("Digital Clock");
        setSize(400, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
        setResizable(false);

        // Create time label
        timeLabel = new JLabel("", JLabel.CENTER);
        timeLabel.setFont(new Font("Digital-7", Font.BOLD, 60));
        timeLabel.setForeground(Color.BLACK);
        timeLabel.setBackground(Color.WHITE);
        timeLabel.setOpaque(true);

        // Create date label
        dateLabel = new JLabel("", JLabel.CENTER);
        dateLabel.setFont(new Font("Arial", Font.PLAIN, 20));
        dateLabel.setForeground(Color.BLACK);
        dateLabel.setBackground(Color.WHITE);
        dateLabel.setOpaque(true);

        // Add components to frame
        add(timeLabel, BorderLayout.CENTER);
        add(dateLabel, BorderLayout.SOUTH);

        // Add menu bar for format switching
        JMenuBar menuBar = new JMenuBar();
        JMenu formatMenu = new JMenu("Format");
        JMenuItem toggleFormat = new JMenuItem("Toggle 12/24 Hour");
        toggleFormat.addActionListener(e -> toggleTimeFormat());
        formatMenu.add(toggleFormat);
        menuBar.add(formatMenu);
        setJMenuBar(menuBar);

        // Timer to update clock every second
```

```
        Timer timer = new Timer(1000, e -> updateTime());
        timer.start();

        // Initial time update
        updateTime();
    }

    private void updateTime() {
        LocalTime now = LocalTime.now();
        DateTimeFormatter timeFormatter = DateTimeFormatter.ofPattern(
            timeFormat);
        String time = now.format(timeFormatter);
        timeLabel.setText(time);

        // Update date
        String date = java.time.LocalDate.now().format(DateTimeFormatter.
            ofPattern("EEEE, MMMM dd, yyyy"));
        dateLabel.setText(date);
    }

    private void toggleTimeFormat() {
        is24HourFormat = !is24HourFormat;
        timeFormat = is24HourFormat ? "HH:mm:ss" : "h:mm:ss a";
        updateTime();
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            DigitalClock clock = new DigitalClock();
            clock.setVisible(true);
        });
    }
}
```

32.2 Program output



33 Analog clock

Write a Java program to create a digital clock.

33.1 Java implementation

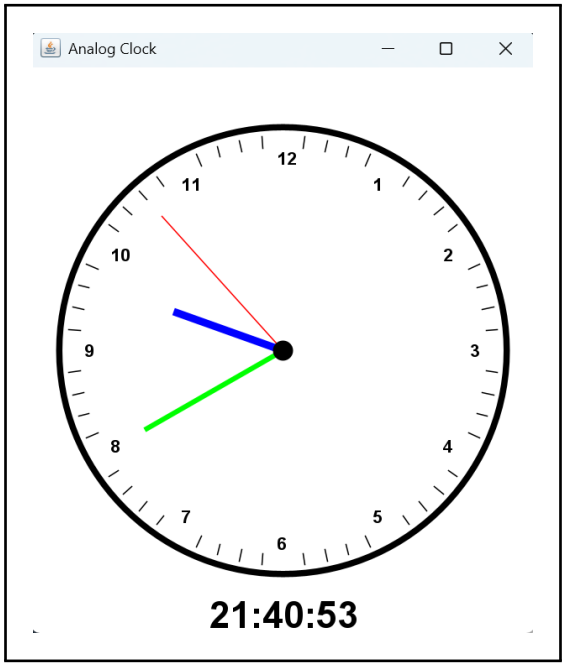
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.time.LocalTime;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class AnalogClock extends JPanel {
    private int centerX, centerY;
    private int clockRadius;
    private LocalTime currentTime;
    private Timer timer;
    private boolean showDigitalTime = true;
    private boolean showDate = true;

    public AnalogClock() {
        // Set up the panel
        setPreferredSize(new Dimension(400, 450));
        setBackground(Color.WHITE);

        // Timer to update clock every second
        timer = new Timer(1000, e -> {
            currentTime = LocalTime.now();
            repaint();
        });
        timer.start();

        // Add mouse listener to toggle digital display
        addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                showDigitalTime = !showDigitalTime;
                repaint();
            }
        });
    }

    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        Graphics2D g2d = (Graphics2D) g;
        g2d.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
           RenderingHints.VALUE_ANTIALIAS_ON);

        // Calculate clock dimensions
        centerX = getWidth() / 2;
        centerY = getHeight() / 2;
```

```
        clockRadius = Math.min(centerX, centerY) - 20;

        // Draw clock face
        drawClockFace(g2d);

        // Draw clock hands
        if (currentTime != null) {
            drawClockHands(g2d);
        }

        // Draw digital time and date
        if (showDigitalTime) {
            drawDigitalTime(g2d);
        }
    }

    private void drawClockFace(Graphics2D g2d) {
        // Draw clock border
        g2d.setColor(Color.BLACK);
        g2d.fillOval(centerX - clockRadius, centerY - clockRadius,
                clockRadius * 2, clockRadius * 2);
        g2d.setColor(Color.WHITE);
        g2d.fillOval(centerX - clockRadius + 5, centerY - clockRadius + 5,
                clockRadius * 2 - 10, clockRadius * 2 - 10);

        // Draw hour markers
        g2d.setColor(Color.BLACK);
        g2d.setFont(new Font("Arial", Font.BOLD, 14));
        for (int i = 1; i <= 12; i++) {
            double angle = Math.toRadians(90 - i * 30);
            int x = centerX + (int) (0.85 * clockRadius * Math.cos(angle))
                ;
            int y = centerY - (int) (0.85 * clockRadius * Math.sin(angle))
                ;
            g2d.drawString(Integer.toString(i), x - 5, y + 5);
        }

        // Draw minute markers
        g2d.setStroke(new BasicStroke(1f));
        for (int i = 0; i < 60; i++) {
            if (i % 5 != 0) { // Skip hour markers
                double angle = Math.toRadians(90 - i * 6);
                int x1 = centerX + (int) (0.90 * clockRadius * Math.cos(
                    angle));
                int y1 = centerY - (int) (0.90 * clockRadius * Math.sin(
                    angle));
                int x2 = centerX + (int) (0.95 * clockRadius * Math.cos(
                    angle));
                int y2 = centerY - (int) (0.95 * clockRadius * Math.sin(
                    angle));
                g2d.drawLine(x1, y1, x2, y2);
            }
        }
    }
```

```
    private void drawClockHands(Graphics2D g2d) {
        int hours = currentTime.getHour() % 12;
        int minutes = currentTime.getMinute();
        int seconds = currentTime.getSecond();

        // Draw hour hand
        double hourAngle = Math.toRadians(90 - (hours * 30 + minutes *
            0.5));
        int hourHandLength = (int) (0.5 * clockRadius);
        int hourX = centerX + (int) (hourHandLength * Math.cos(hourAngle))
            ;
        int hourY = centerY - (int) (hourHandLength * Math.sin(hourAngle))
            ;
        g2d.setStroke(new BasicStroke(6f));
        g2d.setColor(Color.BLUE);
        g2d.drawLine(centerX, centerY, hourX, hourY);

        // Draw minute hand
        double minuteAngle = Math.toRadians(90 - minutes * 6);
        int minuteHandLength = (int) (0.7 * clockRadius);
        int minuteX = centerX + (int) (minuteHandLength * Math.cos(
            minuteAngle));
        int minuteY = centerY - (int) (minuteHandLength * Math.sin(
            minuteAngle));
        g2d.setStroke(new BasicStroke(4f));
        g2d.setColor(Color.GREEN);
        g2d.drawLine(centerX, centerY, minuteX, minuteY);

        // Draw second hand
        double secondAngle = Math.toRadians(90 - seconds * 6);
        int secondHandLength = (int) (0.8 * clockRadius);
        int secondX = centerX + (int) (secondHandLength * Math.cos(
            secondAngle));
        int secondY = centerY - (int) (secondHandLength * Math.sin(
            secondAngle));
        g2d.setStroke(new BasicStroke(1f));
        g2d.setColor(Color.RED);
        g2d.drawLine(centerX, centerY, secondX, secondY);

        // Draw center cap
        g2d.setColor(Color.BLACK);
        g2d.fillOval(centerX - 8, centerY - 8, 16, 16);
    }

    private void drawDigitalTime(Graphics2D g2d) {
        // Draw digital time
        String time = currentTime.format(DateTimeFormatter.ofPattern("HH:
            mm:ss"));
        g2d.setFont(new Font("Digital-7", Font.BOLD, 30));
        FontMetrics fm = g2d.getFontMetrics();
        int timeWidth = fm.stringWidth(time);
        g2d.setColor(Color.BLACK);
```

```
        g2d.drawString(time, centerX - timeWidth/2, centerY + clockRadius
            + 40);

        // Draw date if enabled
        if (showDate) {
            String date = LocalDate.now().format(DateTimeFormatter.
                ofPattern("EEEE, MMMM d, yyyy"));
            g2d.setFont(new Font("Arial", Font.PLAIN, 16));
            fm = g2d.getFontMetrics();
            int dateWidth = fm.stringWidth(date);
            g2d.drawString(date, centerX - dateWidth/2, centerY +
                clockRadius + 70);
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            JFrame frame = new JFrame("Analog Clock");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.add(new AnalogClock());
            frame.pack();
            frame.setLocationRelativeTo(null);
            frame.setVisible(true);
        });
    }
}
```

33.2 Program output



34 India's Flag

Write a Java programme to create India's flag.

34.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.awt.geom.*;

public class IndianFlag extends JFrame {

    public IndianFlag() {
        setTitle("Indian National Flag");
        setSize(600, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        add(new FlagPanel());
    }

    static class FlagPanel extends JPanel {
        @Override
        protected void paintComponent(Graphics g) {
            super.paintComponent(g);
            Graphics2D g2d = (Graphics2D) g;
            g2d.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
                RenderingHints.VALUE_ANTIALIAS_ON);

            // Draw background (sky)
            GradientPaint skyGradient = new GradientPaint(0, 0, new Color
                (135, 206, 235),
                    0, getHeight(), new Color(176, 226, 255));
            g2d.setPaint(skyGradient);
            g2d.fillRect(0, 0, getWidth(), getHeight());

            // Draw flag pole
            drawFlagPole(g2d);

            // Draw the flag
            drawIndianFlag(g2d);
        }

        private void drawFlagPole(Graphics2D g2d) {
            // Draw pole
            g2d.setColor(new Color(101, 67, 33));
            g2d.fillRect(100, 50, 10, 300);

            // Draw pole top
            g2d.setColor(new Color(218, 165, 32));
            g2d.fillOval(95, 45, 20, 20);

            // Draw pole base
            g2d.setColor(new Color(139, 69, 19));
```

```
            g2d.fillRect(90, 350, 30, 20);
        }

        private void drawIndianFlag(Graphics2D g2d) {
            int flagX = 110;
            int flagY = 70;
            int flagWidth = 300;
            int flagHeight = 200;

            // Draw flag border
            g2d.setColor(Color.BLACK);
            g2d.drawRect(flagX, flagY, flagWidth, flagHeight);

            // Draw saffron stripe (top)
            g2d.setColor(new Color(255, 128, 0));
            g2d.fillRect(flagX, flagY, flagWidth, flagHeight / 3);

            // Draw white stripe (middle)
            g2d.setColor(Color.WHITE);
            g2d.fillRect(flagX, flagY + flagHeight / 3, flagWidth,
               flagHeight / 3);

            // Draw green stripe (bottom)
            g2d.setColor(new Color(0, 128, 0));
            g2d.fillRect(flagX, flagY + 2 * flagHeight / 3, flagWidth,
               flagHeight / 3);

            // Calculate center of the Ashoka Chakra
            int whiteBandHeight = flagHeight / 3;
            int centerY = flagY + flagHeight / 3 + whiteBandHeight / 2;

            // Draw Ashoka Chakra (wheel)
            drawAshokaChakra(g2d, flagX + flagWidth / 2, centerY,
               whiteBandHeight);
        }

        private void drawAshokaChakra(Graphics2D g2d, int centerX, int
           centerY, int whiteBandHeight) {
            int chakraDiameter = (int) (whiteBandHeight * 0.9); //
               Diameter of the wheel
            int outerRadius = chakraDiameter / 2;

            // Draw blue circle
            g2d.setColor(new Color(0, 0, 128));
            g2d.setStroke(new BasicStroke(2));
            g2d.draw(new Ellipse2D.Double(centerX - outerRadius, centerY -
                outerRadius,
                    chakraDiameter, chakraDiameter));

            // Draw 24 spokes (all connected at the center)
            int spokes = 24;
            double angleIncrement = 360.0 / spokes;

            for (int i = 0; i < spokes; i++) {
```

```
                double angle = Math.toRadians(i * angleIncrement - 90); //
                    -90 to start at top

                // Calculate spoke endpoint on outer circle
                int x2 = centerX + (int) (outerRadius * Math.cos(angle));
                int y2 = centerY + (int) (outerRadius * Math.sin(angle));

                // Draw line from center to outer circle
                g2d.drawLine(centerX, centerY, x2, y2);
            }

            // Draw the small inner circle
            g2d.fill(new Ellipse2D.Double(centerX - 4, centerY - 4, 8, 8))
                ;
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            new IndianFlag().setVisible(true);
        });
    }
}
```

34.2 Program output

35 Ashok Chakra

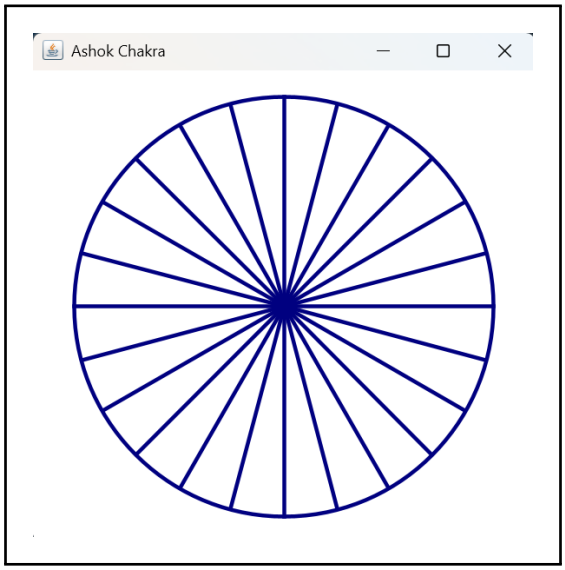
Write a Java programme to create Ashok Chakra.

35.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.awt.geom.*;

public class AshokChakra extends JFrame {

    public AshokChakra() {
        setTitle("Ashok Chakra");
        setSize(400, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        add(new ChakraPanel());
    }

    static class ChakraPanel extends JPanel {
        @Override
        protected void paintComponent(Graphics g) {
            super.paintComponent(g);
            Graphics2D g2d = (Graphics2D) g;
            g2d.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
                RenderingHints.VALUE_ANTIALIAS_ON);

            // Set background to white (like the flag's middle stripe)
            g2d.setColor(Color.WHITE);
            g2d.fillRect(0, 0, getWidth(), getHeight());

            // Draw the Ashok Chakra
            drawAshokChakra(g2d, getWidth()/2, getHeight()/2, Math.min(
                getWidth(), getHeight())/2 - 20);
        }

        private void drawAshokChakra(Graphics2D g2d, int centerX, int
            centerY, int radius) {
            // Draw the navy blue outer circle
            g2d.setColor(new Color(0, 0, 128)); // Navy blue
            g2d.setStroke(new BasicStroke(3));
            g2d.draw(new Ellipse2D.Double(centerX - radius, centerY -
                radius,
                    radius * 2, radius * 2));

            // Draw the 24 spokes (all connected at the center)
            int spokes = 24;
            double angleIncrement = 2 * Math.PI / spokes; // Angle between
                spokes in radians

            for (int i = 0; i < spokes; i++) {
                double angle = i * angleIncrement;
```

```
                // Calculate endpoint on outer circle
                int x = centerX + (int)(radius * Math.cos(angle));
                int y = centerY + (int)(radius * Math.sin(angle));

                // Draw line from center to outer circle
                g2d.drawLine(centerX, centerY, x, y);
            }

            // Draw the small navy blue circle at center
            g2d.fill(new Ellipse2D.Double(centerX - 5, centerY - 5, 10,
                10));

            // Add label
            g2d.setColor(Color.BLACK);
            g2d.setFont(new Font("Arial", Font.BOLD, 16));
            String label = "Ashok Chakra (24 Spokes)";
            int labelWidth = g2d.getFontMetrics().stringWidth(label);
            g2d.drawString(label, centerX - labelWidth/2, centerY + radius
                + 30);
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            new AshokChakra().setVisible(true);
        });
    }
}
```

35.2 Program output



36 Concentric circles

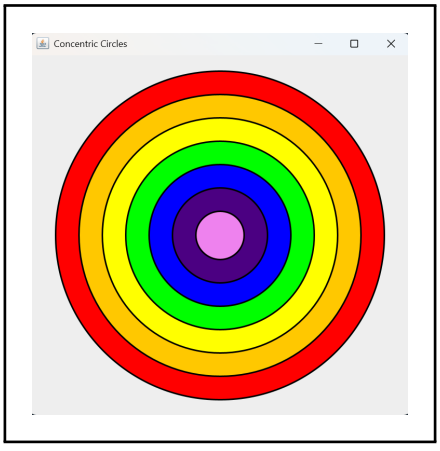
Write a Java programme to create concentric circles in different colours.

36.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.awt.geom.*;

public class ConcentricCircles extends JFrame {

    public ConcentricCircles() {
        setTitle("Concentric Circles");
        setSize(500, 500);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        add(new CirclesPanel());
    }

    static class CirclesPanel extends JPanel {
        // Colors for the concentric circles
        private final Color[] colors = {
                Color.RED,
                Color.ORANGE,
                Color.YELLOW,
                Color.GREEN,
                Color.BLUE,
                new Color(75, 0, 130), // INDIGO
                new Color(238, 130, 238)  // VIOLET
        };

        @Override
        protected void paintComponent(Graphics g) {
            super.paintComponent(g);
            Graphics2D g2d = (Graphics2D) g;
            g2d.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
                RenderingHints.VALUE_ANTIALIAS_ON);

            // Get center of the panel
            int centerX = getWidth() / 2;
            int centerY = getHeight() / 2;

            // Initial radius (will decrease for each circle)
            int radius = Math.min(getWidth(), getHeight()) / 2 - 20;

            // Draw concentric circles
            for (int i = 0; i < colors.length; i++) {
                // Set circle color
                g2d.setColor(colors[i]);

                // Calculate diameter for this circle
                int diameter = radius * 2;
```

```
                // Draw filled circle
                g2d.fill(new Ellipse2D.Double(
                        centerX - radius,
                        centerY - radius,
                        diameter,
                        diameter));

                // Draw outline
                g2d.setColor(Color.BLACK);
                g2d.setStroke(new BasicStroke(2));
                g2d.draw(new Ellipse2D.Double(
                        centerX - radius,
                        centerY - radius,
                        diameter,
                        diameter));

                // Reduce radius for next circle
                radius -= 30;

                // Stop if radius becomes too small
                if (radius < 10) break;
            }
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            new ConcentricCircles().setVisible(true);
        });
    }
}
```

36.2 Program output



37 Shapes

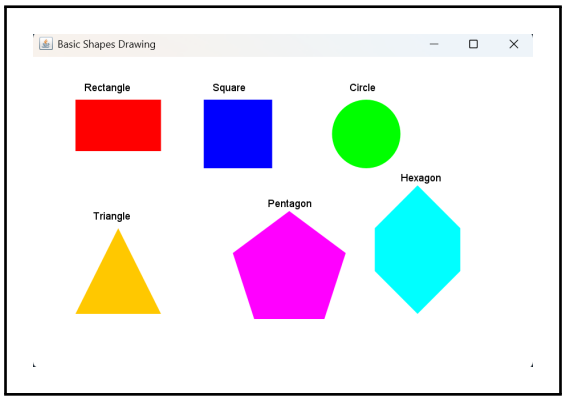
Write a Java programme to create different shapes.

37.1 Java implementation

```
import javax.swing.*;
import java.awt.*;

public class ShapeDrawing extends JFrame {

    public ShapeDrawing() {
        setTitle("Basic Shapes Drawing");
        setSize(600, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        add(new ShapesPanel());
    }

    static class ShapesPanel extends JPanel {
        @Override
        protected void paintComponent(Graphics g) {
            super.paintComponent(g);
            Graphics2D g2d = (Graphics2D) g;
            g2d.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
                RenderingHints.VALUE_ANTIALIAS_ON);
            // Set background
            g2d.setColor(Color.WHITE);
            g2d.fillRect(0, 0, getWidth(), getHeight());
            // Draw rectangle
            g2d.setColor(Color.RED);
            g2d.fillRect(50, 50, 100, 60);
            g2d.setColor(Color.BLACK);
            g2d.drawString("Rectangle", 60, 40);
            // Draw square
            g2d.setColor(Color.BLUE);
            g2d.fillRect(200, 50, 80, 80);
            g2d.setColor(Color.BLACK);
            g2d.drawString("Square", 210, 40);
            // Draw circle
            g2d.setColor(Color.GREEN);
            g2d.fillOval(350, 50, 80, 80);
            g2d.setColor(Color.BLACK);
            g2d.drawString("Circle", 370, 40);
            // Draw triangle
            g2d.setColor(Color.ORANGE);
            int[] xTriangle = {100, 50, 150};
            int[] yTriangle = {200, 300, 300};
            g2d.fillPolygon(xTriangle, yTriangle, 3);
            g2d.setColor(Color.BLACK);
            g2d.drawString("Triangle", 70, 190);
            // Draw pentagon
            g2d.setColor(Color.MAGENTA);
```

```
            int centerX = 300;
            int centerY = 250;
            int radius = 70;

            int[] xPentagon = new int[5];
            int[] yPentagon = new int[5];
            for (int i = 0; i < 5; i++) {
                double angle = Math.toRadians(-90 + i * 72);  // start at
                    top 90°, step 72° deg each
                xPentagon[i] = centerX + (int)(radius * Math.cos(angle));
                yPentagon[i] = centerY + (int)(radius * Math.sin(angle));
            }
            g2d.fillPolygon(xPentagon, yPentagon, 5);
            g2d.setColor(Color.BLACK);
            g2d.drawString("Pentagon", 275, 175);
            // Draw hexagon
            g2d.setColor(Color.CYAN);
            int[] xHexagon = {450, 500, 500, 450, 400, 400};
            int[] yHexagon = {150, 200, 250, 300, 250, 200};
            g2d.fillPolygon(xHexagon, yHexagon, 6);
            g2d.setColor(Color.BLACK);
            g2d.drawString("Hexagon", 430, 145);
        }
    }
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            new ShapeDrawing().setVisible(true);
        });
    }
}
```

37.2 Program output



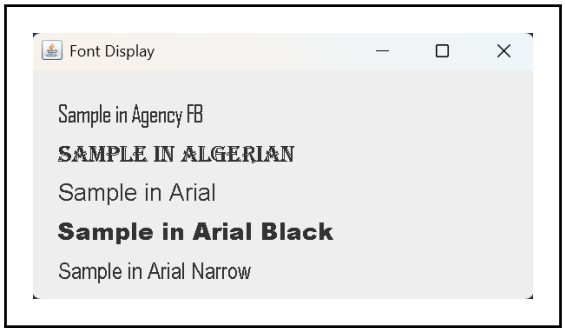
38 Text fonts

Write a Java program to display the fonts of a text.

38.1 Java implementation

```
import javax.swing.*;
import java.awt.*;

public class FontDisplay extends JFrame {
    public FontDisplay() {
        setTitle("Font Display");
        setSize(400, 300);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        add(new JPanel() {
            protected void paintComponent(Graphics g) {
                super.paintComponent(g);
                Graphics2D g2d = (Graphics2D) g;
                String[] fonts = GraphicsEnvironment.
                   getLocalGraphicsEnvironment().
                   getAvailableFontFamilyNames();
                int y = 40;
                for (int i = 0; i < 5; i++) {
                    g2d.setFont(new Font(fonts[i], Font.PLAIN, 18));
                    g2d.drawString("Sample in " + fonts[i], 20, y);
                    y += 30;
                }
            }
        });
    }
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new FontDisplay().setVisible(true
           ));
    }
}
```

38.2 Program output



39 Positioned Text

Write a Java program to display a text in a specific location on window.

39.1 Java implememtation

```
import javax.swing.JFrame;
import javax.swing.JPanel;
import java.awt.Graphics;
import java.awt.Font;

public class TextAtLocation extends JPanel {
    private final String message = "Hello, this is my custom text!";
    private final int x = 100; // x-coordinate
    private final int y = 150; // y-coordinate
    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        // Draw the main message
        g.setFont(new Font("SansSerif", Font.PLAIN, 16));
        g.drawString(message, x, y);
        // Draw the coordinates just below the message
        g.setFont(new Font("SansSerif", Font.ITALIC, 12));
        g.drawString("x = " + x + ", y = " + y, x, y + 20);
    }
    public static void main(String[] args) {
        JFrame frame = new JFrame("Text at Specific Location");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 300);
        frame.add(new TextAtLocation());
        frame.setLocationRelativeTo(null); // Center the window
        frame.setVisible(true);
    }
}
```

39.2 Program output

40 Demonstrate labels

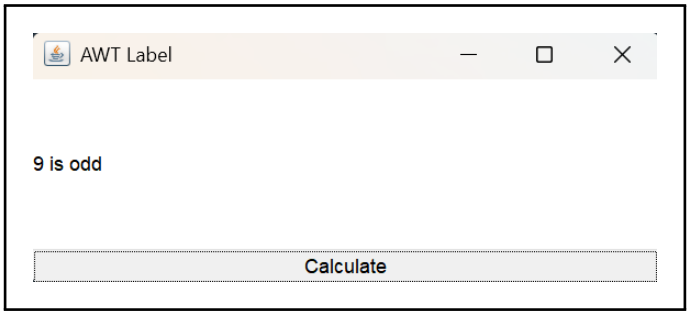
Write a Java program to demonstrate labels.

40.1 Java implementation

```
import java.awt.*;
import java.awt.event.*;

public class LabelDemo extends Frame implements ActionListener {
    // Frame Label and Button Initialised
    private Frame frame;
    private Label label;
    private Button button;
    public int count = 0;

    public LabelDemo()
    {
        // Memory Allocated to frame
        frame = new Frame("AWT Label");

        // Memory Allocated to label
        label = new Label("Check if a Number is Even or Odd");

        // Memory Allocated to button
        button = new Button("Calculate");

        // Register ActionListener
        button.addActionListener(this);

        // Add components to the frame
        frame.add(label, BorderLayout.CENTER);
        frame.add(button, BorderLayout.SOUTH);

        // Dimensions of Frame
        frame.setSize(600, 600);
        frame.setVisible(true);

        // Using WindowListener for closing the window
        frame.addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });
    }

    // Action Listener checks when
    // Button is Pressed
    @Override public void actionPerformed(ActionEvent e)
    {
        if (e.getSource() == button) {
```

```
            String str;
            if (count % 2 == 0)
                str = count + " is Even";
            else
                str = count + " is odd";

            label.setText(str);
            count++;
        }
    }
    public static void main(String[] args)
    {
        LabelDemo example = new LabelDemo();
    }
}
```

40.2 Program output



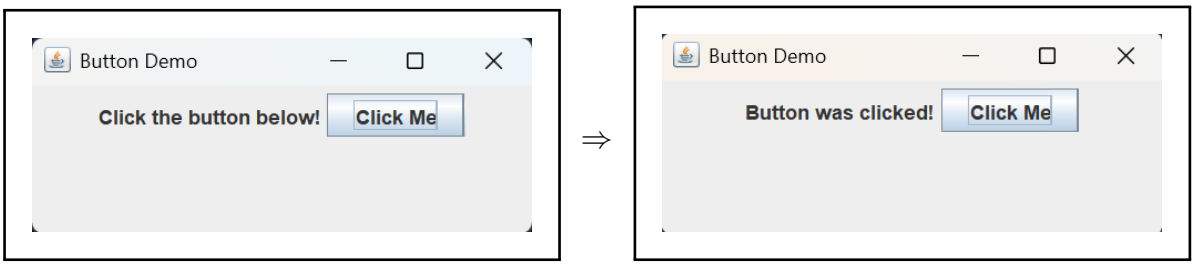
41 Demonstrate buttons

Write a Java program to demonstrate buttons.

41.1 Java implementation

```
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class ButtonDemo {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Button Demo");
        frame.setSize(400, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panel = new JPanel();

        JLabel label = new JLabel("Click the button below!");
        JButton button = new JButton("Click Me");

        // Add action listener to button
        button.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                label.setText("Button was clicked!");
            }
        });

        panel.add(label);
        panel.add(button);

        frame.add(panel);
        frame.setVisible(true);
    }
}
```

41.2 Program output:



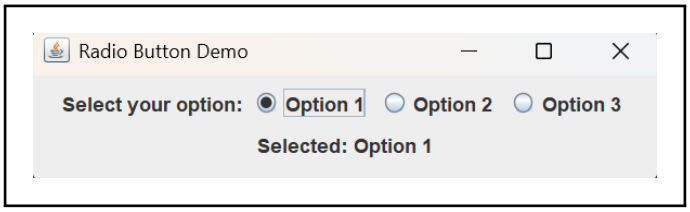
42 Checkbox

Write a Java programme to demonstrate checkbox.

42.1 Java implementation

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class CheckboxDemo {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Radio Button Demo");
        frame.setSize(400, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel panel = new JPanel();
        JLabel label = new JLabel("Select your option:");
        JRadioButton radio1 = new JRadioButton("Option 1");
        JRadioButton radio2 = new JRadioButton("Option 2");
        JRadioButton radio3 = new JRadioButton("Option 3");
        ButtonGroup group = new ButtonGroup();
        group.add(radio1);
        group.add(radio2);
        group.add(radio3);
        JLabel resultLabel = new JLabel("Selected: None");
        ActionListener listener = new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                if (radio1.isSelected()) {
                    resultLabel.setText("Selected: Option 1");
                } else if (radio2.isSelected()) {
                    resultLabel.setText("Selected: Option 2");
                } else if (radio3.isSelected()) {
                    resultLabel.setText("Selected: Option 3");
                }
            }
        };
        radio1.addActionListener(listener);
        radio2.addActionListener(listener);
        radio3.addActionListener(listener);
        panel.add(label);
        panel.add(radio1);
        panel.add(radio2);
        panel.add(radio3);
        panel.add(resultLabel);
        frame.add(panel);
        frame.setVisible(true);
    }
}
```

42.2 Program output



43 Checkbox group

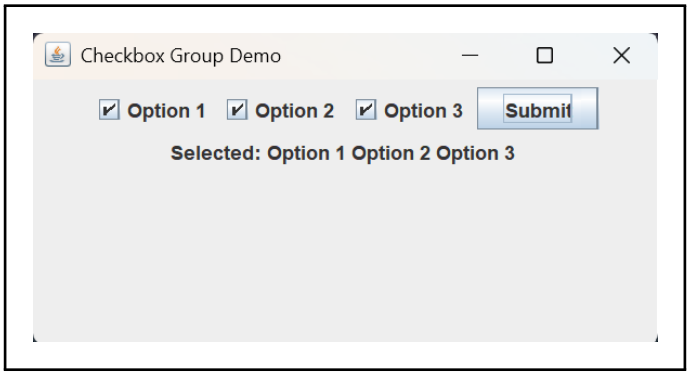
Write a Java program to demonstrate checkbox group.

43.1 Java implementation

```
 import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class CheckboxGroupDemo extends JFrame implements ActionListener {

    JCheckBox option1, option2, option3;
    JButton submitButton;
    JLabel resultLabel;

    public CheckboxGroupDemo() {
        setTitle("Checkbox Group Demo");
        setSize(400, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout());

        // Create checkboxes
        option1 = new JCheckBox("Option 1");
        option2 = new JCheckBox("Option 2");
        option3 = new JCheckBox("Option 3");

        // Add checkboxes to frame
        add(option1);
        add(option2);
        add(option3);

        // Submit button
        submitButton = new JButton("Submit");
        submitButton.addActionListener(this);
        add(submitButton);

        // Label to display selected checkboxes
        resultLabel = new JLabel("Selected: None");
        add(resultLabel);

        setLocationRelativeTo(null); // Center window
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        StringBuilder selectedOptions = new StringBuilder("Selected: ");

        boolean anySelected = false;
        if (option1.isSelected()) {
            selectedOptions.append("Option 1 ");
            anySelected = true;
        }
```

```
        if (option2.isSelected()) {
            selectedOptions.append("Option 2 ");
            anySelected = true;
        }
        if (option3.isSelected()) {
            selectedOptions.append("Option 3 ");
            anySelected = true;
        }

        if (!anySelected) {
            selectedOptions.append("None");
        }

        resultLabel.setText(selectedOptions.toString());
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            new CheckboxGroupDemo().setVisible(true);
        });
    }
}
```

43.2 Program output



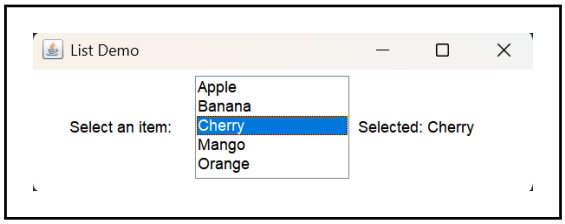
44 List

Write a Java program to demonstrate list.

44.1 Java implementation

```
import java.awt.*;
import java.awt.event.*;

public class ListDemo extends Frame {
    public ListDemo() {
        setTitle("List Demo");
        setSize(400, 300);
        setLayout(new FlowLayout());
        Label label = new Label("Select an item:");
        List itemList = new List(5, false);
        itemList.add("Apple");
        itemList.add("Banana");
        itemList.add("Cherry");
        itemList.add("Mango");
        itemList.add("Orange");
        Label resultLabel = new Label("Selected: None");
        itemList.addItemListener(new ItemListener() {
            public void itemStateChanged(ItemEvent e) {
                String selected = itemList.getSelectedItem();
                resultLabel.setText("Selected: " + selected);
            }
        });
        add(label);
        add(itemList);
        add(resultLabel);
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                dispose();
            }
        });
        setVisible(true);
    }
    public static void main(String[] args) {
        new ListDemo();
    }
}
```

44.2 Program output



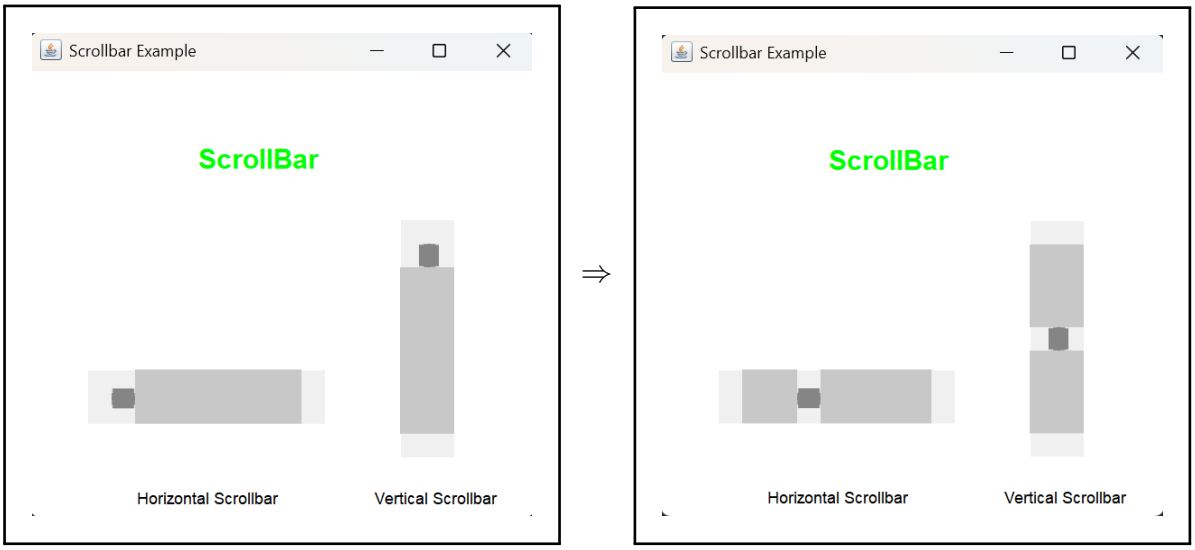
45 Scroll bar

Write a Java program to demonstrate handling a scroll bar.

45.1 Java implementation

```
import java.awt.*;
import java.awt.event.*;

public class ScrollBarDemo {
    public static void main(String[] args) {
        Frame f = new Frame("Scrollbar Example");

        Label l = new Label("ScrollBar");
        l.setBounds(130, 80, 200, 30);
        l.setFont(new Font("Arial", Font.BOLD, 20));
        l.setForeground(Color.GREEN);
        f.add(l);

        Scrollbar s1 = new Scrollbar();  // Vertical by default
        s1.setBounds(280, 140, 40, 175);
        f.add(s1);

        Label l1 = new Label("Vertical Scrollbar");
        l1.setBounds(260, 330, 200, 30);
        f.add(l1);

        Scrollbar s2 = new Scrollbar(Scrollbar.HORIZONTAL);
        s2.setBounds(50, 250, 175, 40);
        f.add(s2);

        Label l2 = new Label("Horizontal Scrollbar");
        l2.setBounds(85, 330, 200, 30);
        f.add(l2);

        f.setSize(400, 400);
        f.setLayout(null);
        f.setVisible(true);

        // Add window listener to handle window close
        f.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                f.dispose();
            }
        });
    }
}
```

45.2 Program output



46 Calculator

Write a Java program to create a calculator.

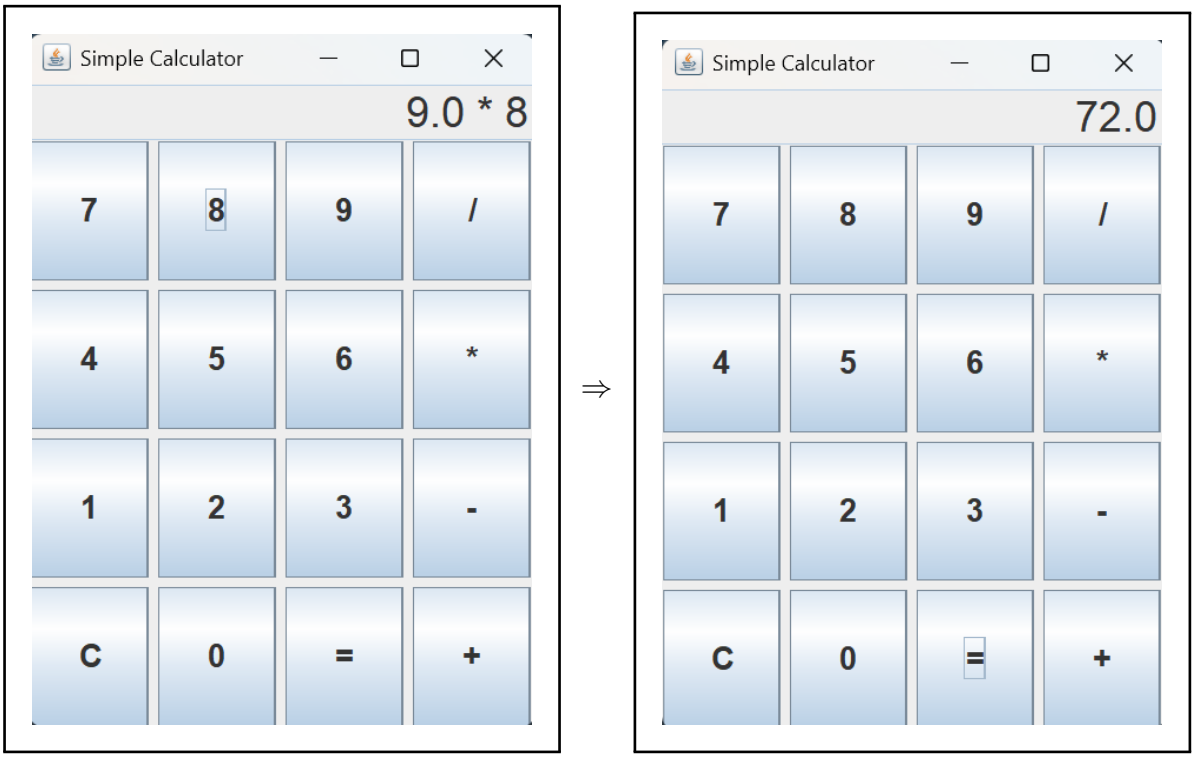
46.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class SimpleCalculator extends JFrame implements ActionListener {

    private JTextField display;
    private double firstNumber = 0;
    private String operation = "";
    private boolean startNewNumber = true;
    private String secondInput = "";

    public SimpleCalculator() {
        setTitle("Simple Calculator");
        setSize(300, 400);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setLayout(new BorderLayout());

        display = new JTextField("0");
        display.setFont(new Font("Arial", Font.PLAIN, 24));
        display.setHorizontalAlignment(JTextField.RIGHT);
        display.setEditable(false);
        add(display, BorderLayout.NORTH);

        JPanel buttonPanel = new JPanel(new GridLayout(4, 4, 5, 5));
        String[] buttons = {
                "7", "8", "9", "/",
                "4", "5", "6", "*",
                "1", "2", "3", "-",
                "C", "0", "=", "+"
        };

        for (String text : buttons) {
            JButton button = new JButton(text);
            button.setFont(new Font("Arial", Font.BOLD, 18));
            button.addActionListener(this);
            buttonPanel.add(button);
        }

        add(buttonPanel, BorderLayout.CENTER);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        String cmd = e.getActionCommand();
```

```
        if (Character.isDigit(cmd.charAt(0))) {
            if (operation.isEmpty()) {
                // Entering first number
                if (startNewNumber) {
                    display.setText("");
                    startNewNumber = false;
                }
                display.setText(display.getText() + cmd);
            } else {
                // Entering second number
                secondInput += cmd;
                display.setText(firstNumber + " " + operation + " " +
                    secondInput);
            }

        } else if (cmd.equals("C")) {
            display.setText("0");
            firstNumber = 0;
            operation = "";
            secondInput = "";
            startNewNumber = true;

        } else if (cmd.equals("=")) {
            if (!operation.isEmpty() && !secondInput.isEmpty()) {
                double secondNumber = Double.parseDouble(secondInput);
                double result = calculate(firstNumber, secondNumber,
                    operation);
                String fullExpr = display.getText();

                // Show expression first
                display.setText(fullExpr);

                // After 1 second, show result
                Timer timer = new Timer(1000, ev -> display.setText(String
                    .valueOf(result)));
                timer.setRepeats(false);
                timer.start();

                // Reset for next operation
                firstNumber = result;
                operation = "";
                secondInput = "";
                startNewNumber = true;
            }

        } else { // Operators: + - * /
            if (!display.getText().isEmpty()) {
                try {
                    firstNumber = Double.parseDouble(display.getText());
                    operation = cmd;
                    startNewNumber = false;
                } catch (NumberFormatException ignored) {
                }
            }
```

```
        }
    }

    private double calculate(double a, double b, String op) {
        return switch (op) {
            case "+" -> a + b;
            case "-" -> a - b;
            case "*" -> a * b;
            case "/" -> b != 0 ? a / b : 0;
            default -> b;
        };
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new SimpleCalculator().setVisible
           (true));
    }
}
```

46.2 Program output



47 Menu bar

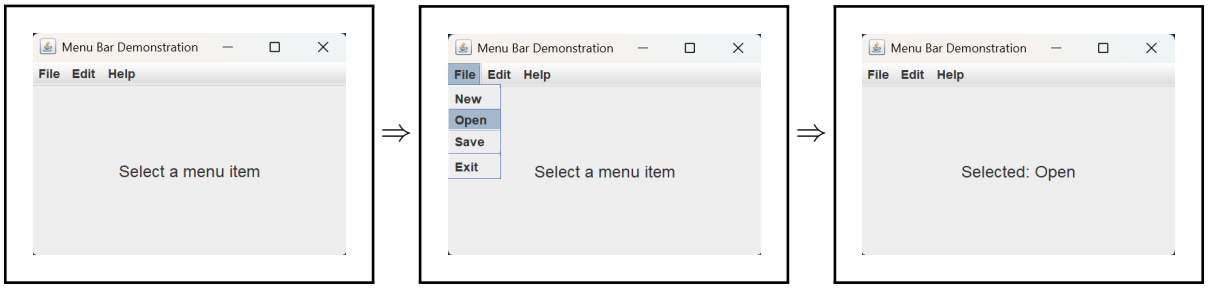
Write a Java program to demonstrate menu bar.

47.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class MenuBarDemo extends JFrame {

    public MenuBarDemo() {
        setTitle("Menu Bar Demonstration");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        // Create the menu bar
        JMenuBar menuBar = new JMenuBar();

        // Create File menu
        JMenu fileMenu = new JMenu("File");

        // Create menu items for File menu
        JMenuItem newItem = new JMenuItem("New");
        JMenuItem openItem = new JMenuItem("Open");
        JMenuItem saveItem = new JMenuItem("Save");
        JMenuItem exitItem = new JMenuItem("Exit");

        // Add action listeners
        exitItem.addActionListener(e -> System.exit(0));

        // Add items to File menu
        fileMenu.add(newItem);
        fileMenu.add(openItem);
        fileMenu.add(saveItem);
        fileMenu.addSeparator(); // Adds a separator line
        fileMenu.add(exitItem);

        // Create Edit menu
        JMenu editMenu = new JMenu("Edit");

        // Create menu items for Edit menu
        JMenuItem cutItem = new JMenuItem("Cut");
        JMenuItem copyItem = new JMenuItem("Copy");
        JMenuItem pasteItem = new JMenuItem("Paste");

        // Add items to Edit menu
        editMenu.add(cutItem);
        editMenu.add(copyItem);
        editMenu.add(pasteItem);
```

```
        // Create Help menu
        JMenu helpMenu = new JMenu("Help");
        JMenuItem aboutItem = new JMenuItem("About");
        helpMenu.add(aboutItem);

        // Add menus to menu bar
        menuBar.add(fileMenu);
        menuBar.add(editMenu);
        menuBar.add(helpMenu);

        // Set the menu bar for this frame
        setJMenuBar(menuBar);

        // Create a label to display menu selections
        JLabel messageLabel = new JLabel("Select a menu item", JLabel.
            CENTER);
        messageLabel.setFont(new Font("Arial", Font.PLAIN, 16));
        add(messageLabel, BorderLayout.CENTER);

        // Create a common action listener for menu items
        ActionListener menuAction = e -> {
            JMenuItem source = (JMenuItem) e.getSource();
            messageLabel.setText("Selected: " + source.getText());
        };

        // Add action listener to all menu items
        newItem.addActionListener(menuAction);
        openItem.addActionListener(menuAction);
        saveItem.addActionListener(menuAction);
        cutItem.addActionListener(menuAction);
        copyItem.addActionListener(menuAction);
        pasteItem.addActionListener(menuAction);
        aboutItem.addActionListener(menuAction);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            MenuBarDemo demo = new MenuBarDemo();
            demo.setVisible(true);
        });
    }
}
```

47.2 Program output



48 Image display

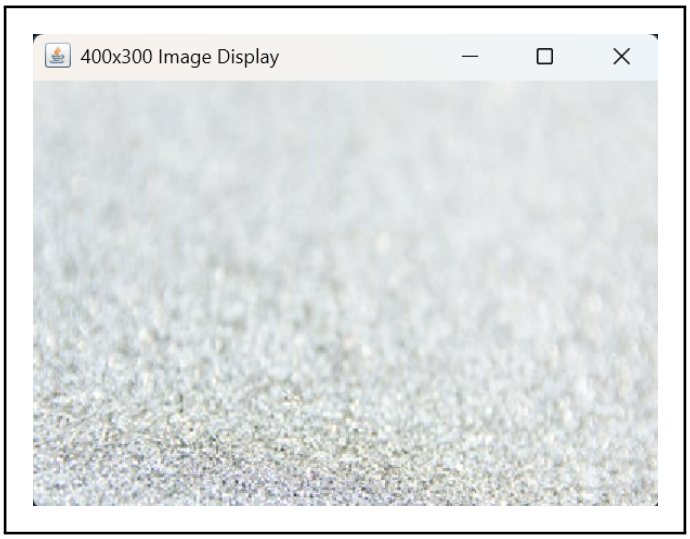
Write a Java program to display an image.

48.1 Java implementation

```
import javax.swing.*;
import java.awt.*;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;

public class ImageDisplay extends JFrame {

    public ImageDisplay(String imagePath) {
        setTitle("400x300 Image Display");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(400, 300); // Fixed window size

        try {
            // Load the original image
            BufferedImage originalImage = ImageIO.read(new File(imagePath)
                );

            // Resize the image to 400x400
            Image resizedImage = originalImage.getScaledInstance(400, 400,
                Image.SCALE_SMOOTH);

            // Create a white background BufferedImage
            BufferedImage bufferedResizedImage = new BufferedImage(400,
                400, BufferedImage.TYPE_INT_RGB);
            Graphics2D g2d = bufferedResizedImage.createGraphics();

            // Fill the background with white
            g2d.setColor(Color.WHITE);
            g2d.fillRect(0, 0, 300, 300);

            // Draw the resized image on top
            g2d.drawImage(resizedImage, 0, 0, null);
            g2d.dispose();

            // Display the resized image
            JLabel label = new JLabel(new ImageIcon(bufferedResizedImage))
                ;
            add(label);
            setLocationRelativeTo(null); // Center the window

        } catch (IOException e) {
            JOptionPane.showMessageDialog(this, "Error loading image: " +
                e.getMessage(),
                    "Error", JOptionPane.ERROR_MESSAGE);
            System.exit(1);
```

```
        }
    }

    public static void main(String[] args) {

        String imagePath = "src//pexels-photo-751378.jpeg";
        SwingUtilities.invokeLater(() -> {
            ImageDisplay display = new ImageDisplay(imagePath);
            display.setVisible(true);
        });
    }
}
```

48.2 Program output



49 Sound effect

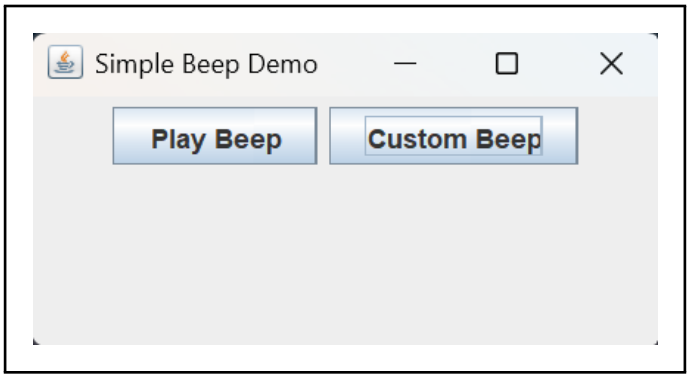
Write a Java program to demonstrate sound effect.

49.1 Java implementation

```
import javax.sound.sampled.AudioFormat;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.SourceDataLine;
import javax.swing.*;
import java.awt.*;


public class SimpleBeepDemo extends JFrame {

    public SimpleBeepDemo() {
        setTitle("Simple Beep Demo");
        setSize(300, 150);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLayout(new FlowLayout());

        JButton beepButton = new JButton("Play Beep");
        beepButton.addActionListener(e -> Toolkit.getDefaultToolkit().beep
            ());
        add(beepButton);

        JButton customBeepButton = new JButton("Custom Beep");
        customBeepButton.addActionListener(e -> playCustomBeep());
        add(customBeepButton);
    }

    private void playCustomBeep() {
        try {
            // Generate a simple tone
            int duration = 1000; // milliseconds
            int freq = 440; // Hz (A4 note)

            byte[] buf = new byte[1];
            AudioFormat af = new AudioFormat(8000f, 8, 1, true, false);
            SourceDataLine sdl = AudioSystem.getSourceDataLine(af);
            sdl.open(af);
            sdl.start();

            for (int i = 0; i < duration * 8; i++) {
                double angle = i / (8000f / freq) * 2.0 * Math.PI;
                buf[0] = (byte)(Math.sin(angle) * 127.0);
                sdl.write(buf, 0, 1);
            }

            sdl.drain();
            sdl.stop();
            sdl.close();
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            SimpleBeepDemo demo = new SimpleBeepDemo();
            demo.setVisible(true);
        });
    }
}
```

49.2 Program output



50 Multiple exception

Write a Java Program for multiple exception in a single code.

50.1 Java Implementation

```
import java.util.Scanner;
import java.util.InputMismatchException;
import java.io.File;
import java.io.FileNotFoundException;

public class MultipleExceptionDemo {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        try {
            // Example 1: ArithmeticException (division by zero)
            System.out.print("Enter numerator: ");
            int numerator = scanner.nextInt();
            System.out.print("Enter denominator: ");
            int denominator = scanner.nextInt();

            int result = divideNumbers(numerator, denominator);
            System.out.println("Division result: " + result);

            // Example 2: ArrayIndexOutOfBoundsException
            System.out.print("Enter array index to access (0-4): ");
            int index = scanner.nextInt();
            int[] numbers = {10, 20, 30, 40, 50};
            System.out.println("Value at index " + index + ": " + numbers[
               index]);

            // Example 3: FileNotFoundException
            System.out.print("Enter file name to read: ");
            String fileName = scanner.next();
            readFile(fileName);

            // Example 4: NumberFormatException
            System.out.print("Enter a number as string to parse: ");
            String numberStr = scanner.next();
            int parsedNumber = Integer.parseInt(numberStr);
            System.out.println("Parsed number: " + parsedNumber);

        } catch (ArithmeticException e) {
            System.out.println("Error: Cannot divide by zero!");
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Error: Array index out of bounds!");
        } catch (FileNotFoundException e) {
            System.out.println("Error: File not found!");
        } catch (NumberFormatException e) {
            System.out.println("Error: Invalid number format!");
        } catch (InputMismatchException e) {
```

```
            System.out.println("Error: Invalid input type!");
        } finally {
            System.out.println("\nThis block always executes (cleanup code
                )");
            scanner.close();
        }

        System.out.println("\nProgram continues after exception handling
            ...");
    }

    public static int divideNumbers(int a, int b) throws
        ArithmeticException {
        return a / b;
    }

    public static void readFile(String fileName) throws
        FileNotFoundException {
        File file = new File(fileName);
        Scanner fileScanner = new Scanner(file);
        System.out.println("File exists and can be read");
        fileScanner.close();
    }
}
```

50.2 Program output

50.2.1 Output 1: ArithmeticException (Division by zero)

```
Enter numerator: 10
Enter denominator: 0
Error: Cannot divide by zero!

This block always executes (cleanup code)

Program continues after exception handling...
```

50.2.2 Output 2: ArrayIndexOutOfBoundsException

```
Enter numerator: 15
Enter denominator: 3
Division result: 5
Enter array index to access (0-4): 7
Error: Array index out of bounds!

This block always executes (cleanup code)

Program continues after exception handling...
```

50.2.3 Output 3: NumberFormatException

```
Enter numerator: 12
Enter denominator: 4
Division result: 3
Enter array index to access (0-4): 2
Value at index 2: 30
Enter a number as string to parse: hello
Error: Invalid number format!

This block always executes (cleanup code)

Program continues after exception handling...
```