JAVA PROGRAMMING-1

SLIP.1

Q1) Write a Program to print all Prime numbers in an array of 'n' elements. (use command line arguments)

ANS:

```
public class PrimeNumbers {

  public static boolean isPrime(int num) {
    if (num < 2) {
      return false;
    }
    for (int i = 2; i <= Math.sqrt(num); i++) {
      if (num % i == 0) {
        return false;
      }
    }
    return true;
  }

  public static void main(String[] args) {
        if (args.length == 0) {
            System.out.println("Please provide an array of numbers as
command line arguments.");
            return;
        }

        System.out.println("Prime numbers in the array are:");
        for (String arg : args) {
            int num = Integer.parseInt(arg);
            if (isPrime(num)) {
                System.out.println(num);
            }
        }
    }
}
```

Q2) Define an abstract class Staff with protected members id and name. Define a parameterized constructor. Define one subclass OfficeStaff with member department. Create n objects of OfficeStaff and display all details.

ANS:

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

```
}

abstract class Staff {
   protected int id;
   protected String name;

   public Staff(int id, String name) {
      this.id = id;
      this.name = name;
   }
}

class OfficeStaff extends Staff {
   private String department;

   public OfficeStaff(int id, String name, String department) {
      super(id, name);
      this.department = department;
   }

   public void displayDetails() {
      System.out.println("ID: " + id + ", Name: " + name + ",
Department: " + department);
   }
```

```
}

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

        System.out.print("Enter the number of OfficeStaff objects: ");
        int n = scanner.nextInt();
        OfficeStaff[] staffArray = new OfficeStaff[n];

        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for OfficeStaff " + (i + 1) + ":");
            System.out.print("ID: ");
            int id = scanner.nextInt();
            scanner.nextLine();  // Consume newline
            System.out.print("Name: ");
            String name = scanner.nextLine();
            System.out.print("Department: ");
            String department = scanner.nextLine();

            staffArray[i] = new OfficeStaff(id, name, department);
        }
        System.out.println("\nDisplaying details of all OfficeStaff:");
        for (OfficeStaff staff : staffArray) {
            staff.displayDetails();
        }

        scanner.close();
    }
}
```

SLIP.2

Q1) Write a program to read the First Name and Last Name of a person, his weight and height using command line arguments. Calculate the BMI Index which is defined as the individual's body mass divided by the square of their height. (Hint : BMI = Wts. In kgs / (ht) ^2)

ANS :

```
public class BMICalculator {
   public static void main(String[] args) {
      if (args.length < 4) {
         System.out.println("Please provide First Name, Last Name, Weight (in kgs), and Height (in meters) as command line arguments.");
         return;
      }

      String firstName = args[0];
      String lastName = args[1];
      double weight = Double.parseDouble(args[2]);
      double height = Double.parseDouble(args[3]);

      double bmi = weight / (height * height);
```

```
        System.out.println("Person's Name: " + firstName + " " +
lastName);
        System.out.println("Weight: " + weight + " kg");
        System.out.println("Height: " + height + " meters");
        System.out.println("BMI Index: " + bmi);
    }
}
```

Q2) Define a class CricketPlayer (name,no_of_innings,no_of_times_notout, totatruns, bat_avg). Create an array of n player objects .Calculate the batting average for each player using static method avg(). Define a static sort method which sorts the array on the basis of average. Display the player details in sorted order.

ANS:

```
import java.util.Arrays;
import java.util.Scanner;

class CricketPlayer {
    String name;
    int no_of_innings;
    int no_of_times_notout;
    int totalruns;
    double bat_avg;

    public CricketPlayer(String name, int no_of_innings, int
no_of_times_notout, int totalruns) {
        this.name = name;
        this.no_of_innings = no_of_innings;
        this.no_of_times_notout = no_of_times_notout;
        this.totalruns = totalruns;
        this.bat_avg = avg(totalruns, no_of_innings,
no_of_times_notout);
    }

    public static double avg(int totalruns, int no_of_innings, int
no_of_times_notout) {
        return (double) totalruns / (no_of_innings - no_of_times_notout);
    }

    public static void sort(CricketPlayer[] players) {
        Arrays.sort(players, (p1, p2) -> Double.compare(p2.bat_avg,
p1.bat_avg));
    }

    public void displayDetails() {
        System.out.println("Name: " + name + ", Innings: " +
no_of_innings + ", Not Out: " + no_of_times_notout +
                              ", Total Runs: " + totalruns + ", Batting Average: " +
bat_avg);
    }
}

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

        System.out.print("Enter the number of players: ");
        int n = scanner.nextInt();
        CricketPlayer[] players = new CricketPlayer[n];

        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for player " + (i + 1) + ":");
            System.out.print("Name: ");
            String name = scanner.next();
            System.out.print("No of Innings: ");
            int innings = scanner.nextInt();
            System.out.print("No of Times Not Out: ");
            int notout = scanner.nextInt();
            System.out.print("Total Runs: ");
            int totalruns = scanner.nextInt();

            players[i] = new CricketPlayer(name, innings, notout,
totalruns);
        }

        CricketPlayer.sort(players);

        System.out.println("\nPlayer details sorted by batting
average:");
        for (CricketPlayer player : players) {
            player.displayDetails();
        }

        scanner.close();
    }
}
```

SLIP.3

Q1) Write a program to accept ‘n’ name of cities from the user and sort them in ascending order.

ANS:

```
import java.util.Arrays;

import java.util.Scanner;


public class CitySorter {

   public static void main(String[] args) {

      Scanner scanner = new Scanner(System.in);

      System.out.print("Enter the number of cities: ");

      int n = scanner.nextInt();

      scanner.nextLine();

      String[] cities = new String[n];

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

         System.out.print("Enter the name of city " + (i + 1) + ": ");

         cities[i] = scanner.nextLine();

      }

      Arrays.sort(cities);

      System.out.println("\nCities in ascending order:");

      for (String city : cities) {
        System.out.println(city);
      }
      scanner.close();
   }
}
```

Q2) Define a class patient (patient_name, patient_age, patient_oxy_level,patient_HRCT_report). Create an object of patient. Handle appropriate exception while patient oxygen level less than 95% and HRCT scan report greater than 10, then throw user defined Exception "Patient is Covid Positive(+) and Need to Hospitalized" otherwise display its information.

ANS:

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

class Patient {
   String patient_name;
   int patient_age;
   double patient_oxy_level;
}

    double patient_HRCT_report;

    public Patient(String patient_name, int patient_age, double
patient_oxy_level, double patient_HRCT_report) throws
PatientException {
        this.patient_name = patient_name;
        this.patient_age = patient_age;
        this.patient_oxy_level = patient_oxy_level;
        this.patient_HRCT_report = patient_HRCT_report;

        if (patient_oxy_level < 95 || patient_HRCT_report > 10) {
            throw new PatientException("Patient is Covid Positive(+) and
Need to Hospitalized");
        }
    }

    public void displayInfo() {
        System.out.println("Patient Name: " + patient_name);
        System.out.println("Patient Age: " + patient_age);
        System.out.println("Oxygen Level: " + patient_oxy_level);
        System.out.println("HRCT Report: " + patient_HRCT_report);
    }
```

```
}

public class Main {
    public static void main(String[] args) {
        try {
            Patient patient = new Patient("John Doe", 30, 94.5, 12);
            patient.displayInfo();
        } catch (PatientException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

SLIP.4

Q1) Write a program to print an array after changing the rows and columns of a given two-dimensional array.

ANS:

```
 import java.util.Scanner;

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

      System.out.print("Enter the number of rows: ");
      int rows = scanner.nextInt();
      System.out.print("Enter the number of columns: ");
      int cols = scanner.nextInt();

      int[][] matrix = new int[rows][cols];
      int[][] transpose = new int[cols][rows];

      System.out.println("Enter the elements of the matrix:");
      for (int i = 0; i < rows; i++) {
         for (int j = 0; j < cols; j++) {
            matrix[i][j] = scanner.nextInt();
         }
      }

      for (int i = 0; i < rows; i++) {
         for (int j = 0; j < cols; j++) {
            transpose[j][i] = matrix[i][j];
         }
      }

      System.out.println("Transposed matrix:");
      for (int i = 0; i < cols; i++) {
         for (int j = 0; j < rows; j++) {
            System.out.print(transpose[i][j] + " ");
         }
         System.out.println();
      }

      scanner.close();
   }
}
```

Q2) Write a program to design a screen using Awt that will take a user name and password. If the user name and password are not same, raise an Exception with appropriate message. User can have 3 login chances only. Use clear button to clear the TextFields.

ANS:

```
import java.awt.*;

import java.awt.event.*;


class LoginException extends Exception {

   public LoginException(String message) {

      super(message);

   }

}


public class LoginScreenAWT extends Frame implements
ActionListener {

   Label userLabel, passLabel, msgLabel;

   TextField userText, passText;

   Button loginButton, clearButton;

   int attempts = 0;


   public LoginScreenAWT() {

      setLayout(new FlowLayout());
```

```
userLabel = new Label("Username:");

passLabel = new Label("Password:");

msgLabel = new Label();


userText = new TextField(20);

passText = new TextField(20);

passText.setEchoChar('*');


loginButton = new Button("Login");

clearButton = new Button("Clear");


add(userLabel);

add(userText);

add(passLabel);

add(passText);

add(loginButton);

add(clearButton);

add(msgLabel);


loginButton.addActionListener(this);

clearButton.addActionListener(this);
```

```
        setSize(300, 200);
        setTitle("Login Screen");
        setVisible(true);

        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                System.exit(0);
            }
        });
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == loginButton) {
            try {
                validateLogin();
            } catch (LoginException ex) {
                msgLabel.setText(ex.getMessage());
            }
        } else if (e.getSource() == clearButton) {
            userText.setText("");
            passText.setText("");
            msgLabel.setText("");
        }
    }

    public void validateLogin() throws LoginException {
        String username = userText.getText();
        String password = passText.getText();

        if (!username.equals(password)) {
            attempts++;
            if (attempts >= 3) {
                throw new LoginException("Maximum login attempts
exceeded.");
            } else {
                throw new LoginException("Username and password do not
match. Attempt " + attempts + " of 3.");
            }
        } else {
            msgLabel.setText("Login successful!");
        }
    }
    public static void main(String[] args) {
        new LoginScreenAWT();
    }
}
```

SLIP.5

Q1) Write a program for multilevel inheritance such that Country is inherited from Continent. State is inherited from Country. Display the place, State, Country and Continent.

ANS:

```
class Continent {
   String continentName;

   public Continent(String continentName) {
      this.continentName = continentName;
   }

   public void displayContinent() {
      System.out.println("Continent: " + continentName);
   }
}

class Country extends Continent {
   String countryName;

   public Country(String continentName, String countryName) {
      super(continentName);
```

```
        this.countryName = countryName;
    }

    public void displayCountry() {
        System.out.println("Country: " + countryName);
    }
}

class State extends Country {
    String stateName;
    String placeName;

    public State(String continentName, String countryName, String
stateName, String placeName) {
        super(continentName, countryName);
        this.stateName = stateName;
        this.placeName = placeName;
    }

    public void displayState() {
        System.out.println("State: " + stateName);
    }
    public void displayPlace() {
        System.out.println("Place: " + placeName);
    }
}

public class Main {
    public static void main(String[] args) {
        State state = new State("Asia", "India", "Maharashtra",
"Mumbai");

        state.displayPlace();
        state.displayState();
        state.displayCountry();
        state.displayContinent();
    }
}
```

Q2) Write a menu driven program to perform the following operations on multidimensional array ie matrices :

1. Addition

2.Multiplication

3.EXIT

ANS:

```
import java.util.Scanner;

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

      System.out.print("Enter the number of rows: ");
      int rows = scanner.nextInt();
      System.out.print("Enter the number of columns: ");
      int cols = scanner.nextInt();

      int[][] matrix1 = new int[rows][cols];
      int[][] matrix2 = new int[rows][cols];

      System.out.println("Enter elements for the first matrix:");
      for (int i = 0; i < rows; i++) {
         for (int j = 0; j < cols; j++) {
            matrix1[i][j] = scanner.nextInt();
         }
      }
```

```
System.out.println("Enter elements for the second matrix:");
for (int i = 0; i < rows; i++) {
   for (int j = 0; j < cols; j++) {
      matrix2[i][j] = scanner.nextInt();
   }
}

int choice;
do {
   System.out.println("\nMenu:");
   System.out.println("1. Addition");
   System.out.println("2. Multiplication");
   System.out.println("3. EXIT");
   System.out.print("Enter your choice: ");
   choice = scanner.nextInt();

   switch (choice) {
      case 1:
         int[][] sum = addMatrices(matrix1, matrix2, rows, cols);
         System.out.println("Result of Addition:");
         printMatrix(sum, rows, cols);
```

```
                break;

            case 2:
                int[][] product = multiplyMatrices(matrix1, matrix2, rows,
cols);
                System.out.println("Result of Multiplication:");
                printMatrix(product, rows, cols);
                break;

            case 3:
                System.out.println("Exiting...");
                break;

            default:
                System.out.println("Invalid choice. Please choose
again.");
          }
        } while (choice != 3);

        scanner.close();
    }
```

```
   public static int[][] addMatrices(int[][] matrix1, int[][] matrix2, int
rows, int cols) {
      int[][] sum = new int[rows][cols];
      for (int i = 0; i < rows; i++) {
         for (int j = 0; j < cols; j++) {
            sum[i][j] = matrix1[i][j] + matrix2[i][j];
         }
      }
      return sum;
   }

   public static int[][] multiplyMatrices(int[][] matrix1, int[][] matrix2,
int rows, int cols) {
      int[][] product = new int[rows][cols];
      for (int i = 0; i < rows; i++) {
         for (int j = 0; j < cols; j++) {
            product[i][j] = 0;
            for (int k = 0; k < cols; k++) {
               product[i][j] += matrix1[i][k] * matrix2[k][j];
            }
         }
      }
```

```
        return product;
    }

    public static void printMatrix(int[][] matrix, int rows, int cols) {
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

SLIP.6

Q1) Write a program to display the Employee(Empid, Empname, Empdesignation, Empsal) information using toString().

ANS:

```
class Employee {
   int empId;
   String empName;
   String empDesignation;
   double empSalary;

   public Employee(int empId, String empName, String
empDesignation, double empSalary) {
      this.empId = empId;
      this.empName = empName;
      this.empDesignation = empDesignation;
      this.empSalary = empSalary;
   }

   @Override
   public String toString() {
      return "Employee ID: " + empId +
            "\nEmployee Name: " + empName +
            "\nEmployee Designation: " + empDesignation +
            "\nEmployee Salary: " + empSalary;
    }
}

public class Main {
    public static void main(String[] args) {
        Employee emp = new Employee(101, "John Doe", "Software
Developer", 75000.0);
        System.out.println(emp);
    }
}
```

Q2) Create an abstract class “order” having members id, description. Create two subclasses “PurchaseOrder” and “Sales Order” having members customer name and Vendor name respectively. Definemethods accept and display in all cases. Create 3 objects each of Purchase Order and Sales Order and accept and display details.

ANS:

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

```
abstract class Order {
   int id;
   String description;

   public Order(int id, String description) {
      this.id = id;
      this.description = description;
   }

   abstract void accept();
   abstract void display();
}

class PurchaseOrder extends Order {
   String customerName;

   public PurchaseOrder(int id, String description) {
      super(id, description);
   }

   @Override
   public void accept() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter Customer Name: ");
        customerName = sc.nextLine();
    }

    @Override
    public void display() {
        System.out.println("Purchase Order ID: " + id);
        System.out.println("Description: " + description);
        System.out.println("Customer Name: " + customerName);
    }
}

class SalesOrder extends Order {
    String vendorName;

    public SalesOrder(int id, String description) {
        super(id, description);
    }

    @Override
    public void accept() {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter Vendor Name: ");

        vendorName = sc.nextLine();

    }


    @Override

    public void display() {

        System.out.println("Sales Order ID: " + id);

        System.out.println("Description: " + description);

        System.out.println("Vendor Name: " + vendorName);

    }

}


public class Main {

    public static void main(String[] args) {

        PurchaseOrder[] purchaseOrders = new PurchaseOrder[3];

        SalesOrder[] salesOrders = new SalesOrder[3];


        for (int i = 0; i < 3; i++) {

            purchaseOrders[i] = new PurchaseOrder(i + 1, "Purchase
Order Description " + (i + 1));

            purchaseOrders[i].accept();
```

```
      }

      for (int i = 0; i < 3; i++) {
         salesOrders[i] = new SalesOrder(i + 1, "Sales Order
Description " + (i + 1));
         salesOrders[i].accept();
      }

      System.out.println("\n--- Purchase Orders ---");
      for (PurchaseOrder po : purchaseOrders) {
         po.display();
         System.out.println();
      }

      System.out.println("\n--- Sales Orders ---");
      for (SalesOrder so : salesOrders) {
         so.display();
         System.out.println();
      }
   }
}
```

SLIP.7

Q1) Design a class for Bank. Bank Class should support following operations:

a. Deposit a certain amount into an account

b. Withdraw a certain amount from an account

c. Return a Balance value specifying the amount with details

ANS:

```
import java.util.Scanner;

class Bank {
    private double balance;

    public Bank() {
        this.balance = 0.0;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.printf("Deposited: ₹%.2f%n", amount);
        } else {
            System.out.println("Deposit amount must be positive.");
        }
    }

    public void withdraw(double amount) {
        if (amount > 0) {
            if (amount <= balance) {
                balance -= amount;
                System.out.printf("Withdrawn: ₹%.2f%n", amount);
            } else {
                System.out.println("Insufficient balance.");
            }
        } else {
            System.out.println("Withdrawal amount must be positive.");
        }
    }

    public double getBalance() {
        return balance;
    }
}
```

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

        do {
            System.out.println("1. Deposit");
            System.out.println("2. Withdraw");
            System.out.println("3. Check Balance");
            System.out.println("4. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter amount to deposit: ");
                    double depositAmount = scanner.nextDouble();
                    bank.deposit(depositAmount);
                    break;
                case 2:
```

```
          System.out.print("Enter amount to withdraw: ");
          double withdrawAmount = scanner.nextDouble();
          bank.withdraw(withdrawAmount);
          break;
        case 3:
          System.out.printf("Current Balance: ₹%.2f%n",
bank.getBalance());
          break;
        case 4:
          System.out.println("Exiting...");
          break;
        default:
          System.out.println("Invalid choice. Please try again.");
      }
      System.out.println();
    } while (choice != 4);

    scanner.close();
  }
}
```

Q2) Write a program to accept a text file from user and display the contents of a file in reverse order and change its case.

ANS:

```
import java.io.*;
import java.util.*;

public class ReverseFileContent {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter the path of the text file: ");
      String filePath = scanner.nextLine();

      try {
         BufferedReader reader = new BufferedReader(new
FileReader(filePath));
         List<String> lines = new ArrayList<>();
         String line;

         while ((line = reader.readLine()) != null) {
            lines.add(line.toUpperCase());
         }
         reader.close();
```

```
            Collections.reverse(lines);


            for (String reversedLine : lines) {
                System.out.println(reversedLine);
            }
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

}

SLIP.8

Q1) Create a class Sphere, to calculate the volume and surface area of sphere. (Hint : Surface area=4*3.14(r*r), Volume=(4/3)3.14(r*r*r))

ANS:

```
import java.util.Scanner;


class Sphere {
  private double radius;


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


  public double calculateVolume() {
    return (4.0 / 3.0) * 3.14 * (radius * radius * radius);
  }


  public double calculateSurfaceArea() {
    return 4 * 3.14 * (radius * radius);
  }
}
```

```
public class SphereTest {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter the radius of the sphere: ");
      double radius = scanner.nextDouble();

      Sphere sphere = new Sphere(radius);
      double volume = sphere.calculateVolume();
      double surfaceArea = sphere.calculateSurfaceArea();

      System.out.printf("Volume of the sphere: %.2f%n", volume);
      System.out.printf("Surface area of the sphere: %.2f%n",
surfaceArea);
   }
}
```

Q2) Design a screen to handle the Mouse Events such as MOUSE_MOVED and MOUSE_CLICKED and display the position of the Mouse_Click in a TextField.

ANS:

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

```
public class MouseEventDemo extends JFrame {
   private JTextField textField;

   public MouseEventDemo() {
      textField = new JTextField();
      textField.setBounds(50, 50, 200, 30);
      add(textField);

      addMouseMotionListener(new MouseMotionAdapter() {
         public void mouseMoved(MouseEvent e) {
            textField.setText("Mouse Moved: (" + e.getX() + ", " + e.getY()
+ ")");
         }
      });

      addMouseListener(new MouseAdapter() {
         public void mouseClicked(MouseEvent e) {
            textField.setText("Mouse Clicked: (" + e.getX() + ", " + e.getY()
+ ")");
         }
      });
```

```
        setSize(300, 200);

        setLayout(null);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setVisible(true);

    }


    public static void main(String[] args) {

        new MouseEventDemo();

    }

}
```

SLIP.9

Q1) Define a “Clock” class that does the following :

a. Accept Hours, Minutes and Seconds

b. Check the validity of numbers

c. Set the time to AM/PM mode Use the necessary constructors and methods to do the above task

ANS:

```
import java.util.Scanner;

class Clock {
   private int hours;
   private int minutes;
   private int seconds;
   private String period;

   public Clock(int hours, int minutes, int seconds, String period) {
      setTime(hours, minutes, seconds, period);
   }

   public void setTime(int hours, int minutes, int seconds, String
period) {
      if (isValidTime(hours, minutes, seconds, period)) {
            this.hours = hours % 12;

            this.minutes = minutes % 60;

            this.seconds = seconds % 60;

            this.period = period.equalsIgnoreCase("AM") ||
period.equalsIgnoreCase("PM") ? period.toUpperCase() : "AM";

        } else {

            throw new IllegalArgumentException("Invalid time format");

        }

    }


    private boolean isValidTime(int hours, int minutes, int seconds,
String period) {

        return (hours >= 0 && hours < 24) && (minutes >= 0 && minutes <
60) && (seconds >= 0 && seconds < 60) &&

               (period.equalsIgnoreCase("AM") ||
period.equalsIgnoreCase("PM"));

    }


    public String displayTime() {

        return String.format("%02d:%02d:%02d %s", hours, minutes,
seconds, period);

    }

}
```

```
public class ClockTest {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter hours (0-23): ");
      int hours = scanner.nextInt();
      System.out.print("Enter minutes (0-59): ");
      int minutes = scanner.nextInt();
      System.out.print("Enter seconds (0-59): ");
      int seconds = scanner.nextInt();
      System.out.print("Enter period (AM/PM): ");
      String period = scanner.next();

      try {
         Clock clock = new Clock(hours, minutes, seconds, period);
         System.out.println("Current Time: " + clock.displayTime());
      } catch (IllegalArgumentException e) {
         System.out.println(e.getMessage());
      }
   }
}
```

Q2) Write a program to using marker interface create a class Product (product_id, product_name, product_cost, product_quantity) default and parameterized constructor. Create objectsof class product and display the contents of each object and Also display the object count.

ANS:

```
interface Marker {}


class Product implements Marker {
   private int productId;
   private String productName;
   private double productCost;
   private int productQuantity;
   private static int objectCount = 0;


   public Product() {
      this.productId = 0;
      this.productName = "Unknown";
      this.productCost = 0.0;
      this.productQuantity = 0;
      objectCount++;
```

```
    }

    public Product(int productId, String productName, double
productCost, int productQuantity) {
        this.productId = productId;
        this.productName = productName;
        this.productCost = productCost;
        this.productQuantity = productQuantity;
        objectCount++;
    }

    public void displayProduct() {
        System.out.printf("Product ID: %d, Name: %s, Cost: ₹%.2f,
Quantity: %d%n", productId, productName, productCost,
productQuantity);
    }

    public static int getObjectCount() {
        return objectCount;
    }
}
```

```
public class ProductTest {
   public static void main(String[] args) {
      Product product1 = new Product(101, "Laptop", 65000.00, 10);
      Product product2 = new Product(102, "Smartphone", 30000.00,
20);
      Product product3 = new Product();

      product1.displayProduct();
      product2.displayProduct();
      product3.displayProduct();
      System.out.println("Total Products Created: " +
Product.getObjectCount());
   }
}
```

SLIP.10

Q1) Write a program to find the cube of given number using functional interface.

ANS:

```
import java.util.Scanner;

@FunctionalInterface
interface CubeCalculator {
   int calculate(int number);
}

public class CubeFinder {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter a number to find its cube: ");
      int number = scanner.nextInt();

      CubeCalculator cubeCalculator = (n) -> n * n * n;
      int cube = cubeCalculator.calculate(number);

      System.out.printf("The cube of %d is %d.%n", number, cube);
```

```
    }
}
```

Q2) Write a program to create a package name student. Define class StudentInfo with method to display information about student such as rollno, class, and percentage. Create another class StudentPer with method to find percentage of the student. Accept student details like rollno, name, class and marks of 6 subject from user.

ANS:

StudentInfo.java

```
package student;

public class StudentInfo {
    private int rollNo;
    private String name;
    private String studentClass;

    public StudentInfo(int rollNo, String name, String studentClass) {
        this.rollNo = rollNo;
        this.name = name;
        this.studentClass = studentClass;
    }
    public void displayInfo() {
        System.out.printf("Roll No: %d%nName: %s%nClass: %s%n",
rollNo, name, studentClass);
    }
}
```

StudentPer.java

```
package student;

import java.util.Scanner;

public class StudentPer {
    private int[] marks = new int[6];

    public void findPercentage() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter marks for 6 subjects: ");
        int totalMarks = 0;

        for (int i = 0; i < 6; i++) {
            marks[i] = scanner.nextInt();
            totalMarks += marks[i];
        }
```

```java
        double percentage = (totalMarks / 6.0);
        System.out.printf("Percentage: %.2f%%%n", percentage);
    }
}
```

Main.java

```java
import student.StudentInfo;
import student.StudentPer;

import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter Roll No: ");
        int rollNo = scanner.nextInt();
        scanner.nextLine(); // Consume the newline
        System.out.print("Enter Name: ");
        String name = scanner.nextLine();
        System.out.print("Enter Class: ");
        String studentClass = scanner.nextLine();
        StudentInfo studentInfo = new StudentInfo(rollNo, name,
studentClass);
        studentInfo.displayInfo();

        StudentPer studentPer = new StudentPer();
        studentPer.findPercentage();
    }
}
```

SLIP.11

Q1) Define an interface “Operation” which has method volume( ).Define a constant PI having a value 3.142 Create a class cylinder which implements this interface (members – radius,height). Create one object and calculate the volume.

ANS:

```
interface Operation {
   double PI = 3.142;
   double volume();
}

class Cylinder implements Operation {
   private double radius;
   private double height;

   public Cylinder(double radius, double height) {
      this.radius = radius;
      this.height = height;
   }

   @Override
   public double volume() {
        return PI * radius * radius * height;
    }
}

public class CylinderVolume {
    public static void main(String[] args) {
        Cylinder cylinder = new Cylinder(5, 10);
        double volume = cylinder.volume();
        System.out.printf("The volume of the cylinder is: %.2f%n", 
volume);
    }
}
```

2) Write a program to accept the username and password from user if username and password are not same then raise "Invalid Password" with appropriate msg.

ANS:

```
import java.util.Scanner;

class InvalidPasswordException extends Exception {
    public InvalidPasswordException(String message) {
        super(message);
    }
}
public class UserAuth {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter username: ");
        String username = scanner.nextLine();

        System.out.print("Enter password: ");
        String password = scanner.nextLine();
        try {
           if (!username.equals(password)) {
              throw new InvalidPasswordException("Invalid Password:
Username and password do not match.");
           } else {
              System.out.println("Login successful!");
           }
        } catch (InvalidPasswordException e) {
           System.out.println(e.getMessage());
        }
    }
}
```

}

SLIP.12

Q1) Write a program to create parent class College(cno, cname, caddr) and derived class Department(dno, dname) from College. Write a necessary methods to display College details.

ANS:

```
class College {
   private int cno;
   private String cname;
   private String caddr;

   public College(int cno, String cname, String caddr) {
      this.cno = cno;
      this.cname = cname;
      this.caddr = caddr;
   }

   public void displayCollegeDetails() {
      System.out.println("College No: " + cno);
      System.out.println("College Name: " + cname);
      System.out.println("College Address: " + caddr);
   }
```

```
}

class Department extends College {
   private int dno;
   private String dname;

   public Department(int cno, String cname, String caddr, int dno,
String dname) {
      super(cno, cname, caddr);
      this.dno = dno;
      this.dname = dname;
   }

   public void displayDepartmentDetails() {
      displayCollegeDetails();
      System.out.println("Department No: " + dno);
      System.out.println("Department Name: " + dname);
   }
}

public class Main {
   public static void main(String[] args) {
        Department dept = new Department(101, "ABC College", "123
College Street", 12, "Computer Science");
        dept.displayDepartmentDetails();
    }
}
```

Q2) Write a java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -, *, % operations. Add a text field to display the result.

ANS:

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

public class SimpleCalculator extends Frame implements ActionListener {
    TextField textField;
    Button[] buttons;
    GridLayout layout;

    public SimpleCalculator() {
        super("Simple Calculator");
        setLayout(new BorderLayout());

        textField = new TextField(20);
        add(textField, BorderLayout.NORTH);
```

```
buttons = new Button[16];
for (int i = 0; i < buttons.length; i++) {
  buttons[i] = new Button("" + (i + 1));
  buttons[i].addActionListener(this);
}
buttons[9] = new Button("+");
buttons[9].addActionListener(this);
buttons[10] = new Button("-");
buttons[10].addActionListener(this);
buttons[11] = new Button("*");
buttons[11].addActionListener(this);
buttons[12] = new Button("/");
buttons[12].addActionListener(this);
buttons[13] = new Button("=");
buttons[13].addActionListener(this);
buttons[14] = new Button("0");
buttons[14].addActionListener(this);
buttons[15] = new Button(".");
buttons[15].addActionListener(this);

layout = new GridLayout(4, 4);
Panel panel = new Panel(layout);
for (int i = 0; i < buttons.length; i++) {
  panel.add(buttons[i]);
}
```

```
    add(panel, BorderLayout.CENTER);


    setSize(200, 250);
    setVisible(true);
}

public void actionPerformed(ActionEvent e) {
    String command = e.getActionCommand();
    if (command.equals("=")) {
        try {
            double result = evaluateExpression(textField.getText());
            textField.setText(Double.toString(result));
        } catch (Exception ex) {
            textField.setText("Error");
        }
    } else {
        textField.setText(textField.getText() + command);
    }
}

private double evaluateExpression(String expression) {
    // Implement your expression evaluation logic here
    // You can use a library like JEP or create your own parser
    return 0; // Replace with actual evaluation result
}
```

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

        new SimpleCalculator();

    }

}
```

SLIP.13

Q1) Write a program to accept a file name from command prompt, if the file exits then display number of words and lines in that file.

ANS:

```
import java.io.*;

import java.util.Scanner;


public class FileWordLineCount {

   public static void main(String[] args) {

      if (args.length != 1) {

         System.out.println("Usage: java FileWordLineCount
<filename>");

         return;

      }


      String filename = args[0];

      File file = new File(filename);


      if (!file.exists()) {

         System.out.println("File does not exist.");

         return;

      }
```

```
        int lineCount = 0;
        int wordCount = 0;

        try (Scanner scanner = new Scanner(file)) {
            while (scanner.hasNextLine()) {
                String line = scanner.nextLine();
                lineCount++;
                String[] words = line.split("\\s+");
                wordCount += words.length;
            }

            System.out.println("Number of lines: " + lineCount);
            System.out.println("Number of words: " + wordCount);
        } catch (IOException e) {
            System.out.println("An error occurred while reading the file.");
        }
    }
}
```

Q2) Write a program to display the system date and time in various formats shown below: Current date is : 31/08/2021 Current date is : 08-31-2021 Current date is : Tuesday August 31 2021 Current date and time is : Fri August 31 15:25:59 IST 2021 Current date and time is : 31/08/21 15:25:59 PM +0530

ANS:

```
import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Locale;

import java.util.TimeZone;


public class DateTimeFormats {

   public static void main(String[] args) {

      Date currentDate = new Date();


      SimpleDateFormat format1 = new
SimpleDateFormat("dd/MM/yyyy");

      System.out.println("Current date is : " +
format1.format(currentDate));


      SimpleDateFormat format2 = new SimpleDateFormat("MM-dd-
yyyy");

      System.out.println("Current date is : " +
format2.format(currentDate));
```

```
        SimpleDateFormat format3 = new SimpleDateFormat("EEEE
MMMM dd yyyy", Locale.ENGLISH);

        System.out.println("Current date is : " +
format3.format(currentDate));


        SimpleDateFormat format4 = new SimpleDateFormat("EEE
MMMM dd HH:mm:ss z yyyy", Locale.ENGLISH);

        format4.setTimeZone(TimeZone.getTimeZone("IST"));

        System.out.println("Current date and time is : " +
format4.format(currentDate));


        SimpleDateFormat format5 = new
SimpleDateFormat("dd/MM/yy HH:mm:ss a Z", Locale.ENGLISH);

        format5.setTimeZone(TimeZone.getTimeZone("IST"));

        System.out.println("Current date and time is : " +
format5.format(currentDate));

    }

}
```

SLIP.14

Q1) Write a program to accept a number from the user, if number is zero then throw user defined exception “Number is 0” otherwise check whether no is prime or not (Use static keyword).

ANS:

```
import java.util.Scanner;

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

public class PrimeNumberChecker {

   public static void checkPrime(int num) {
      if (num < 2) {
         System.out.println(num + " is not a prime number.");
         return;
      }

      for (int i = 2; i <= Math.sqrt(num); i++) {
            if (num % i == 0) {
                System.out.println(num + " is not a prime number.");
                return;
            }
        }

        System.out.println(num + " is a prime number.");
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");

        try {
            int number = scanner.nextInt();

            if (number == 0) {
                throw new ZeroException("Number is 0");
            }

            checkPrime(number);
        } catch (ZeroException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Q2) Write a Java program to create a Package “SY” which has a class SYMarks (members – ComputerTotal, MathsTotal, and ElectronicsTotal). Create another package TY which has a class TYMarks (members – Theory, Practicals). Create ‘n’ objects of Student class (having rollNumber, name, SYMarks and TYMarks). Add the marks of SY and TY computer subjects and calculate the Grade (‘A’ for >= 70, ‘B’ for >= 60 ‘C’ for >= 50, Pass Class for > =40 else‘FAIL’) and display the result of the student in proper format.

ANS:

Create the SY package and the SYMarks class.

```
// File: SY/SYMarks.java
package SY;


public class SYMarks {
    public int ComputerTotal;
    public int MathsTotal;
    public int ElectronicsTotal;
    public SYMarks(int computerTotal, int mathsTotal, int
electronicsTotal) {
        this.ComputerTotal = computerTotal;
        this.MathsTotal = mathsTotal;
        this.ElectronicsTotal = electronicsTotal;
    }
}
```

Create the TY package and the `TYMarks` class.

```
// File: TY/TYMarks.java
package TY;

public class TYMarks {
    public int Theory;
    public int Practicals;

    public TYMarks(int theory, int practicals) {
        this.Theory = theory;
        this.Practicals = practicals;
    }
}
```

<u>Create the Student class in the main program to calculate the total marks, grades, and display the results.</u>

```
// File: Student.java
import SY.SYMarks;
import TY.TYMarks;
import java.util.Scanner;

class Student {
   int rollNumber;
   String name;
   SYMarks syMarks;
   TYMarks tyMarks;

   public Student(int rollNumber, String name, SYMarks syMarks,
TYMarks tyMarks) {
      this.rollNumber = rollNumber;
      this.name = name;
      this.syMarks = syMarks;
      this.tyMarks = tyMarks;
   }
```

```
    public void displayResult() {
        int totalComputerMarks = syMarks.ComputerTotal +
tyMarks.Theory + tyMarks.Practicals;
        String grade = calculateGrade(totalComputerMarks);

        System.out.println("Roll Number: " + rollNumber);
        System.out.println("Name: " + name);
        System.out.println("Total Computer Marks (SY + TY): " +
totalComputerMarks);
        System.out.println("Grade: " + grade);
        System.out.println("--------------------------------");
    }

    private String calculateGrade(int totalMarks) {
        if (totalMarks >= 70) {
            return "A";
        } else if (totalMarks >= 60) {
            return "B";
        } else if (totalMarks >= 50) {
            return "C";
        } else if (totalMarks >= 40) {
            return "Pass Class";
        } else {
            return "FAIL";
        }
    }
}

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

        Student[] students = new Student[n];

        for (int i = 0; i < n; i++) {
            System.out.print("Enter roll number: ");
            int rollNumber = scanner.nextInt();

            System.out.print("Enter name: ");
            String name = scanner.next();
            System.out.print("Enter SY Computer marks: ");
            int syComputer = scanner.nextInt();
            System.out.print("Enter SY Maths marks: ");
            int syMaths = scanner.nextInt();
            System.out.print("Enter SY Electronics marks: ");
            int syElectronics = scanner.nextInt();

            SYMarks syMarks = new SYMarks(syComputer, syMaths,
syElectronics);

            System.out.print("Enter TY Theory marks: ");
            int tyTheory = scanner.nextInt();
            System.out.print("Enter TY Practical marks: ");
            int tyPractical = scanner.nextInt();
            TYMarks tyMarks = new TYMarks(tyTheory, tyPractical);
            students[i] = new Student(rollNumber, name, syMarks,
tyMarks);
        }
        for (Student student : students) {
            student.displayResult();
```

```
      }
   }
}
```

SLIP.15

Q1) Accept the names of two files and copy the contents of the first to the second. First file having Book name and Author name in file.

ANS:

```
import java.io.*;
import java.util.Scanner;

public class FileCopy {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter the name of the first file (source): ");
      String sourceFile = scanner.nextLine();

      System.out.print("Enter the name of the second file
(destination): ");
      String destinationFile = scanner.nextLine();

      try {
         FileReader fr = new FileReader(sourceFile);
```

```
            BufferedReader br = new BufferedReader(fr);

            FileWriter fw = new FileWriter(destinationFile);
            BufferedWriter bw = new BufferedWriter(fw);

            String line;
            while ((line = br.readLine()) != null) {
                bw.write(line);
                bw.newLine();
            }

            br.close();
            bw.close();

            System.out.println("Contents copied successfully from " +
sourceFile + " to " + destinationFile);
        } catch (FileNotFoundException e) {
            System.out.println("File not found: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("Error while reading/writing files: " +
e.getMessage());
        }
```

```
    }
}
```

Q2) Write a program to define a class Account having members custname, accno. Define default and parameterized constructor. Create a subclass called SavingAccount with member savingbal, minbal. Create a derived class AccountDetail that extends the class SavingAccount with members, depositamt and withdrawalamt. Write a appropriate method to display customer details.

ANS:

```
class Account {
    String custname;
    int accno;

    public Account() {
        custname = "Unknown";
        accno = 0;
    }

    public Account(String custname, int accno) {
        this.custname = custname;
        this.accno = accno;
    }
}
```

```
class SavingAccount extends Account {
   double savingbal;
   double minbal;

   public SavingAccount() {
      super();
      savingbal = 0.0;
      minbal = 1000.0;
   }

   public SavingAccount(String custname, int accno, double
savingbal, double minbal) {
      super(custname, accno);
      this.savingbal = savingbal;
      this.minbal = minbal;
   }
}

class AccountDetail extends SavingAccount {
   double depositamt;
   double withdrawalamt;
```

```
    public AccountDetail() {
        super();
        depositamt = 0.0;
        withdrawalamt = 0.0;
    }

    public AccountDetail(String custname, int accno, double
savingbal, double minbal, double depositamt, double
withdrawalamt) {
        super(custname, accno, savingbal, minbal);
        this.depositamt = depositamt;
        this.withdrawalamt = withdrawalamt;
    }

    public void displayCustomerDetails() {
        System.out.println("Customer Name: " + custname);
        System.out.println("Account Number: " + accno);
        System.out.println("Saving Balance: " + savingbal);
        System.out.println("Minimum Balance: " + minbal);
        System.out.println("Deposit Amount: " + depositamt);
        System.out.println("Withdrawal Amount: " + withdrawalamt);
    }
}

public class BankDemo {
    public static void main(String[] args) {
        AccountDetail account = new AccountDetail("John Doe", 12345,
5000.0, 1000.0, 2000.0, 500.0);
        account.displayCustomerDetails();
    }
}
```

SLIP.16

Q1) Write a program to find the Square of given number using function interface

ANS:

```
import java.util.Scanner;

@FunctionalInterface
interface Square {
   int calculate(int num);
}

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

      Square squareFunction = (num) -> num * num;
      int result = squareFunction.calculate(number);

      System.out.println("The square of " + number + " is: " + result);
   }
}
```

Q2) Write a program to design a screen using Awt that

ANS:

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

public class AWTExample extends Frame implements
ActionListener {
   MenuBar menuBar;
   Menu menuFile, menuEdit;
   MenuItem menuItemNew, menuItemOpen, menuItemSave,
menuItemExit, menuItemAbout;

   public AWTExample() {
      super("Java AWT Examples");
      setSize(300, 200);
      setLayout(null);

      menuBar = new MenuBar();
      setMenuBar(menuBar);

      menuFile = new Menu("File");
```

```
menuBar.add(menuFile);
menuItemNew = new MenuItem("New");
menuFile.add(menuItemNew);
menuItemOpen = new MenuItem("Open");
menuFile.add(menuItemOpen);
menuItemSave = new MenuItem("Save");
menuFile.add(menuItemSave);
menuFile.addSeparator();
menuItemExit = new MenuItem("Exit");
menuFile.add(menuItemExit);

menuEdit = new Menu("Edit");
menuBar.add(menuEdit);
menuItemAbout = new MenuItem("About");
menuEdit.add(menuItemAbout);

menuItemNew.addActionListener(this);
menuItemOpen.addActionListener(this);
menuItemSave.addActionListener(this);
menuItemExit.addActionListener(this);
menuItemAbout.addActionListener(this);
```

```
        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == menuItemNew) System.out.println("New
clicked");
        else if (e.getSource() == menuItemOpen)
System.out.println("Open clicked");
        else if (e.getSource() == menuItemSave)
System.out.println("Save clicked");
        else if (e.getSource() == menuItemExit) System.exit(0);
        else if (e.getSource() == menuItemAbout)
System.out.println("About clicked");
    }

    public static void main(String[] args) {
        new AWTExample();
    }
}
```

SLIP.17

Q1) Design a Super class Customer (name, phone-number). Derive a class Depositor(accno , balance) from Customer. Again, derive a class Borrower (loan-no, loan-amt) from Depositor. Write necessary member functions to read and display the details of ‘n’customers.

ANS:

```
import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;


class Customer {

  String name;

  String phoneNumber;


  public void readDetails() {

    Scanner scanner = new Scanner(System.in);

    System.out.print("Enter customer name: ");

    name = scanner.nextLine();

    System.out.print("Enter phone number: ");

    phoneNumber = scanner.nextLine();

  }
```

```
    public void displayDetails() {
        System.out.println("Customer Name: " + name);
        System.out.println("Phone Number: " + phoneNumber);
    }
}

class Depositor extends Customer {
    String accNo;
    double balance;

    @Override
    public void readDetails() {
        super.readDetails();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter account number: ");
        accNo = scanner.nextLine();
        System.out.print("Enter balance: ");
        balance = scanner.nextDouble();
    }

    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Account Number: " + accNo);
        System.out.println("Balance: " + balance);
    }
}

class Borrower extends Depositor {
    String loanNo;
    double loanAmount;

    @Override
    public void readDetails() {
        super.readDetails();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter loan number: ");
        loanNo = scanner.nextLine();
        System.out.print("Enter loan amount: ");
        loanAmount = scanner.nextDouble();
    }

    @Override
    public void displayDetails() {
        super.displayDetails();

        System.out.println("Loan Number: " + loanNo);

        System.out.println("Loan Amount: " + loanAmount);

    }

}


public class CustomerDetails {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter number of customers: ");

        int n = scanner.nextInt();

        List<Borrower> customers = new ArrayList<>();


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

            System.out.println("Enter details for customer " + (i + 1) + ":");

            Borrower borrower = new Borrower();

            borrower.readDetails();

            customers.add(borrower);

        }


        System.out.println("\nCustomer Details:");

        for (Borrower customer : customers) {
            customer.displayDetails();
            System.out.println();
        }
    }
}
```

Q2) Write Java program to design three text boxes and two buttons using swing. Enter different strings in first and second textbox. On clicking the First command button, concatenation of two strings should be displayed in third text box and on clicking second command button, reverse of string should display in third text box

ANS:

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

public class StringManipulationApp {
    public static void main(String[] args) {
        JFrame frame = new JFrame("String Manipulation");
        frame.setSize(400, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new FlowLayout());
```

```
JTextField textField1 = new JTextField(15);
JTextField textField2 = new JTextField(15);
JTextField resultField = new JTextField(15);
resultField.setEditable(false);

JButton concatButton = new JButton("Concatenate");
JButton reverseButton = new JButton("Reverse");

concatButton.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e) {
      String text1 = textField1.getText();
      String text2 = textField2.getText();
      resultField.setText(text1 + text2);
   }
});

reverseButton.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e) {
      String text = textField1.getText();
      String reversed = new StringBuilder(text).reverse().toString();
      resultField.setText(reversed);
```

```
            }
        });

        frame.add(new JLabel("Text 1:"));
        frame.add(textField1);
        frame.add(new JLabel("Text 2:"));
        frame.add(textField2);
        frame.add(concatButton);
        frame.add(reverseButton);
        frame.add(new JLabel("Result:"));
        frame.add(resultField);

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

SLIP.18

Q1) Write a program to implement Border Layout Manager.

ANS:

```
import javax.swing.*;

import java.awt.*;


public class BorderLayoutExample {

   public static void main(String[] args) {

      JFrame frame = new JFrame("Border Layout Example");

      frame.setSize(400, 300);

      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      frame.setLayout(new BorderLayout());


      JButton buttonNorth = new JButton("North");

      JButton buttonSouth = new JButton("South");

      JButton buttonEast = new JButton("East");

      JButton buttonWest = new JButton("West");

      JButton buttonCenter = new JButton("Center");


      frame.add(buttonNorth, BorderLayout.NORTH);

      frame.add(buttonSouth, BorderLayout.SOUTH);
```

```
        frame.add(buttonEast, BorderLayout.EAST);

        frame.add(buttonWest, BorderLayout.WEST);

        frame.add(buttonCenter, BorderLayout.CENTER);


        frame.setVisible(true);

    }

}
```

Q2) Define a class CricketPlayer (name,no_of_innings,no_of_times_notout, totatruns, bat_avg). Create an array of n player objects. Calculate the batting average for each player using static method avg(). Define a static sort method which sorts the array on the basis of average. Display the player details in sorted order.

ANS :

```
import java.util.Arrays;

import java.util.Comparator;

import java.util.Scanner;


class CricketPlayer {

    String name;

    int no_of_innings;

    int no_of_times_notout;

    int totalRuns;
```

```
   double bat_avg;

   CricketPlayer(String name, int no_of_innings, int
no_of_times_notout, int totalRuns) {
      this.name = name;
      this.no_of_innings = no_of_innings;
      this.no_of_times_notout = no_of_times_notout;
      this.totalRuns = totalRuns;
      this.bat_avg = avg();
   }

   static double avg(int totalRuns, int no_of_innings, int
no_of_times_notout) {
      if (no_of_innings - no_of_times_notout > 0) {
         return (double) totalRuns / (no_of_innings -
no_of_times_notout);
      }
      return 0;
   }

   void calculateAvg() {
      this.bat_avg = avg(this.totalRuns, this.no_of_innings,
this.no_of_times_notout);
```

```
    }

    static void sortByAverage(CricketPlayer[] players) {
        Arrays.sort(players, new Comparator<CricketPlayer>() {
            public int compare(CricketPlayer p1, CricketPlayer p2) {
                return Double.compare(p2.bat_avg, p1.bat_avg);
            }
        });
    }

    @Override
    public String toString() {
        return "Name: " + name + ", Innings: " + no_of_innings + ", Not
Out: " + no_of_times_notout + ", Total Runs: " + totalRuns + ", Batting
Average: " + bat_avg;
    }
}

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

        for (int i = 0; i < n; i++) {
            System.out.print("Enter name of player " + (i + 1) + ": ");
            String name = scanner.next();
            System.out.print("Enter number of innings: ");
            int innings = scanner.nextInt();
            System.out.print("Enter number of times not out: ");
            int notOut = scanner.nextInt();
            System.out.print("Enter total runs: ");
            int totalRuns = scanner.nextInt();

            players[i] = new CricketPlayer(name, innings, notOut,
totalRuns);
        }

        CricketPlayer.sortByAverage(players);

        System.out.println("\nPlayer details sorted by batting
average:");
        for (CricketPlayer player : players) {
            System.out.println(player);
        }
        scanner.close();
    }
}
```

SLIP.19

Q1) Write a program to accept the two dimensional array from user and display sum of its diagonal elements.

ANS:

```
import java.util.Scanner;

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

        if (rows != cols) {
            System.out.println("Matrix must be square to have diagonal
elements.");
            return;
        }

        int[][] matrix = new int[rows][cols];
        System.out.println("Enter the elements of the matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        int diagonalSum = 0;
        for (int i = 0; i < rows; i++) {
            diagonalSum += matrix[i][i];
        }

        System.out.println("Sum of diagonal elements: " +
diagonalSum);
        scanner.close();
    }
}
```

Q2) Write a program which shows the combo box which includes list of T.Y.B.Sc.(Comp. Sci) subjects. Display the selected subject in a text field.

ANS:

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

import java.awt.event.ActionListener;


public class SubjectComboBox {

   public static void main(String[] args) {

      JFrame frame = new JFrame("T.Y.B.Sc. (Comp. Sci) Subjects");

      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      frame.setSize(400, 200);

      frame.setLayout(null);


      String[] subjects = {

         "Data Structures",

         "Database Management Systems",

         "Operating Systems",

         "Computer Networks",

         "Software Engineering",

         "Web Technologies"

      };


      JComboBox<String> subjectComboBox = new
JComboBox<>(subjects);

      subjectComboBox.setBounds(50, 30, 250, 30);
```

```
        JTextField selectedSubjectField = new JTextField();
        selectedSubjectField.setBounds(50, 80, 250, 30);
        selectedSubjectField.setEditable(false);

        subjectComboBox.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String selectedSubject = (String)
subjectComboBox.getSelectedItem();
                selectedSubjectField.setText(selectedSubject);
            }
        });

        frame.add(subjectComboBox);
        frame.add(selectedSubjectField);
        frame.setVisible(true);
    }
}
```

SLIP.20

Q1) Write a Program to illustrate multilevel Inheritance such that country is inherited from continent. State is inherited from country. Display the place, state, country and continent.

ANS:

```
class Continent {
   String continentName;

   Continent(String continentName) {
      this.continentName = continentName;
   }

   void displayContinent() {
      System.out.println("Continent: " + continentName);
   }
}

class Country extends Continent {
   String countryName;

   Country(String continentName, String countryName) {
      super(continentName);
```

```
        this.countryName = countryName;
    }

    void displayCountry() {
        System.out.println("Country: " + countryName);
    }
}

class State extends Country {
    String stateName;

    State(String continentName, String countryName, String
stateName) {
        super(continentName, countryName);
        this.stateName = stateName;
    }

    void displayState() {
        System.out.println("State: " + stateName);
    }

    void displayDetails() {
```

```
    displayContinent();
    displayCountry();
    displayState();
    System.out.println("Place: Some Place"); // You can customize this if needed
  }
}

public class MultilevelInheritance {
  public static void main(String[] args) {
    State state = new State("Asia", "India", "Maharashtra");
    state.displayDetails();
  }
}
```

Q2) Write a package for Operation, which has two classes, Addition and Maximum. Addition has two methods add () and subtract (), which are used to add two integers and subtract two, float values respectively. Maximum has a method max () to display the maximum of two integers

ANS:

Step 1: Create the Package Structure

Step 2: Implement the `Addition` Class

```
package Operation;


public class Addition {
   public int add(int a, int b) {
      return a + b;
   }


   public float subtract(float a, float b) {
      return a - b;
   }
```

Step 3: Implement the `Maximum` Class

```
package Operation;


public class Maximum {
   public int max(int a, int b) {
      return (a > b) ? a : b;
   }
}
```

Step 4: Create a Test Class

```
import Operation.Addition;
import Operation.Maximum;
```

```
public class TestOperation {
   public static void main(String[] args) {
      Addition addition = new Addition();
      Maximum maximum = new Maximum();

      int sum = addition.add(5, 3);
      float difference = addition.subtract(5.5f, 2.2f);
      int maxValue = maximum.max(10, 20);

      System.out.println("Sum: " + sum);
      System.out.println("Difference: " + difference);
      System.out.println("Maximum: " + maxValue);
   }
}
```

SLIP.21

Q1) Define a class MyDate(Day, Month, year) with methods to accept and display a MyDateobject. Accept date as dd,mm,yyyy. Throw user defined exception "InvalidDateException" if the date is invalid.

ANS:

Step 1: Define the Custom Exception

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

Step 2: Define the `MyDate` Class

```
import java.util.Scanner;

public class MyDate {
   private int day;
   private int month;
   private int year;

   public void acceptDate() throws InvalidDateException {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter day (dd): ");
        day = scanner.nextInt();
        System.out.print("Enter month (mm): ");
        month = scanner.nextInt();
        System.out.print("Enter year (yyyy): ");
        year = scanner.nextInt();

        if (!isValidDate(day, month, year)) {
            throw new InvalidDateException("Invalid date: " + day + "/" +
month + "/" + year);
        }
    }

    public void displayDate() {
        System.out.println("Date: " + day + "/" + month + "/" + year);
    }

    private boolean isValidDate(int day, int month, int year) {
        if (year < 1) return false;
        if (month < 1 || month > 12) return false;
        if (day < 1) return false;
        int[] daysInMonth = {31, (isLeapYear(year) ? 29 : 28), 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
        return day <= daysInMonth[month - 1];
    }

    private boolean isLeapYear(int year) {
        return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
    }
}
```

Step 3: Create a Test Class Step 3: Create a Test Class

```
import java.util.Scanner;

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

        try {
            date.acceptDate();
            date.displayDate();
        } catch (InvalidDateException e) {
```

```
        System.out.println(e.getMessage());
      }
    }
  }
```

Q2) Create an employee class(id,name,deptname,salary). Define a default and parameterized constructor. Use 'this' keyword to initialize instance variables. Keep a count of objects created. Create objects using parameterized constructor and display the object count after each object is created. (Use static member and method). Also display the contents of each object.

ANS:

Employee Class Implementation

```
public class Employee {
  private int id;
  private String name;
  private String deptname;
  private double salary;
  private static int objectCount = 0;

  public Employee() {
    objectCount++;
    }

    public Employee(int id, String name, String deptname, double
salary) {
        this.id = id;
        this.name = name;
        this.deptname = deptname;
        this.salary = salary;
        objectCount++;
    }

    public static int getObjectCount() {
        return objectCount;
    }

    public void displayDetails() {
        System.out.println("ID: " + id + ", Name: " + name + ",
Department: " + deptname + ", Salary: " + salary);
    }
}
```

<u>Test Class to Create Objects</u>

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

```
public class TestEmployee {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of employees to create: ");
        int n = scanner.nextInt();
        Employee[] employees = new Employee[n];

        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for employee " + (i + 1) + ":");
            System.out.print("ID: ");
            int id = scanner.nextInt();
            scanner.nextLine();  // Consume newline
            System.out.print("Name: ");
            String name = scanner.nextLine();
            System.out.print("Department Name: ");
            String deptname = scanner.nextLine();
            System.out.print("Salary: ");
            double salary = scanner.nextDouble();

            employees[i] = new Employee(id, name, deptname, salary);
            System.out.println("Employee created. Total Employees: " +
Employee.getObjectCount());
        }

        System.out.println("\nEmployee Details:");
        for (Employee employee : employees) {
            employee.displayDetails();
        }
    }
}
```

SLIP.22

Q1) Write a program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea() that prints the area of the given shape. (use method overriding).

ANS

Shape Class and Subclasses Implementation

File: `Shape.java`

```
abstract class Shape {

  int dimension1;

  int dimension2;



  Shape(int dimension1, int dimension2) {

    this.dimension1 = dimension1;

    this.dimension2 = dimension2;

  }

    abstract void printArea();

}
```

File: Rectangle.java

```
class Rectangle extends Shape {
   Rectangle(int length, int width) {
      super(length, width);
   }

   void printArea() {
      int area = dimension1 * dimension2;
      System.out.println("Area of Rectangle: " + area);
   }
}
```

File: Triangle.java

```
class Triangle extends Shape {
   Triangle(int base, int height) {
      super(base, height);
   }

   void printArea() {
      double area = 0.5 * dimension1 * dimension2;
```

```
        System.out.println("Area of Triangle: " + area);
    }
}
```

File: `Circle.java`

```
class Circle extends Shape {
    Circle(int radius) {
        super(radius, 0);
    }


    void printArea() {
        double area = Math.PI * dimension1 * dimension1;
        System.out.println("Area of Circle: " + area);
    }
}
```

File: `Main.java`

```
public class Main {
    public static void main(String[] args) {
        Shape rectangle = new Rectangle(5, 10);
        Shape triangle = new Triangle(5, 10);
        Shape circle = new Circle(7);


        rectangle.printArea();
        triangle.printArea();
        circle.printArea();
    }
}
```

Q2) Write a program that handles all mouse events and shows the event name at the center of the Window, red in color when a mouse event is fired. (Use adapter classes).

ANS:

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

public class MouseEventExample extends Frame {
    String message = "";

    public MouseEventExample() {
        setTitle("Mouse Events Example");
        setSize(400, 400);
```

```
setLayout(new FlowLayout());


addMouseListener(new MouseAdapter() {

   public void mouseClicked(MouseEvent e) {

      message = "Mouse Clicked";

      repaint();

   }


   public void mousePressed(MouseEvent e) {

      message = "Mouse Pressed";

      repaint();

   }


   public void mouseReleased(MouseEvent e) {

      message = "Mouse Released";

      repaint();

   }


   public void mouseEntered(MouseEvent e) {
        message = "Mouse Entered";

        repaint();

    }


    public void mouseExited(MouseEvent e) {

        message = "Mouse Exited";

        repaint();

    }

});


addMouseMotionListener(new MouseMotionAdapter() {

    public void mouseDragged(MouseEvent e) {

        message = "Mouse Dragged";

        repaint();

    }


    public void mouseMoved(MouseEvent e) {

        message = "Mouse Moved";

        repaint();
        }
    });

    addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent we) {
            System.exit(0);
        }
    });
}

public void paint(Graphics g) {
    g.setColor(Color.RED);
    g.setFont(new Font("Arial", Font.BOLD, 20));
    FontMetrics fm = g.getFontMetrics();
    int x = (getWidth() - fm.stringWidth(message)) / 2;
    int y = (getHeight() + fm.getAscent()) / 2;
    g.drawString(message, x, y);
}
```

```
    public static void main(String[] args) {
        MouseEventExample example = new MouseEventExample();
        example.setVisible(true);
    }
}
```

SLIP.23

Q1) Define a class MyNumber having one private int data member. Write a default constructor to initialize it to 0 and another constructor to initialize it to a value (Use this). Write methods isNegative, isPositive, isZero, isOdd, isEven. Create an object in main.Use command line arguments to pass a value to the Object.

ANS:

```
public class MyNumber {
    private int data;
}

public MyNumber() {

   this.data = 0;

}


public MyNumber(int value) {

   this.data = value;

}


public boolean isNegative() {

   return data < 0;

}


public boolean isPositive() {

   return data > 0;

}


public boolean isZero() {

   return data == 0;

}
```

```
public boolean isOdd() {
   return data % 2 != 0;
}

public boolean isEven() {
   return data % 2 == 0;
}

public void displayProperties() {
   System.out.println("Value: " + data);
   System.out.println("Is Negative: " + isNegative());
   System.out.println("Is Positive: " + isPositive());
   System.out.println("Is Zero: " + isZero());
   System.out.println("Is Odd: " + isOdd());
   System.out.println("Is Even: " + isEven());
}

public static void main(String[] args) {
        int value = 0;

        if (args.length > 0) {

            try {

                value = Integer.parseInt(args[0]);

            } catch (NumberFormatException e) {

                System.out.println("Invalid input. Using default value 0.");

            }

        }


        MyNumber myNumber = new MyNumber(value);

        myNumber.displayProperties();

    }

}
```

Q2) Write a simple currency converter, as shown in the figure. User can enter the amount of "Singapore Dollars", "US Dollars", or "Euros", in floating-point number. The converted values shall be displayed to 2 decimal places. Assume that 1 USD = 1.41 SGD, 1 USD = 0.92 Euro, 1 SGD = 0.65 Euro.

ANS:

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


public class CurrencyConverter extends Frame implements
ActionListener {
   TextField textFieldSGD, textFieldUSD, textFieldEuro;
   Label labelSGD, labelUSD, labelEuro;
   Button buttonConvert;


   public CurrencyConverter() {
      super("Currency Converter");
      setLayout(new GridLayout(4, 2));


      labelSGD = new Label("Singapore Dollars");
      textFieldSGD = new TextField(10);
      labelUSD = new Label("US Dollars");
      textFieldUSD = new TextField(10);
      labelEuro = new Label("Euros");
      textFieldEuro = new TextField(10);


      add(labelSGD);
```

```
    add(textFieldSGD);

    add(labelUSD);

    add(textFieldUSD);

    add(labelEuro);

    add(textFieldEuro);


    buttonConvert = new Button("Convert");

    buttonConvert.addActionListener(this);

    add(buttonConvert);


    setSize(300, 200);

    setVisible(true);

}


public void actionPerformed(ActionEvent e) {

    if (e.getSource() == buttonConvert) {

        try {

            double sgd = Double.parseDouble(textFieldSGD.getText());

            double usd = Double.parseDouble(textFieldUSD.getText());
            double euro = Double.parseDouble(textFieldEuro.getText());


            double usdToSgd = 1.41;

            double usdToEuro = 0.92;

            double sgdToEuro = 0.65;


            if (!textFieldSGD.getText().isEmpty()) {

               textFieldUSD.setText(String.format("%.2f", sgd /
usdToSgd));

               textFieldEuro.setText(String.format("%.2f", sgd *
sgdToEuro));

            } else if (!textFieldUSD.getText().isEmpty()) {

               textFieldSGD.setText(String.format("%.2f", usd *
usdToSgd));

               textFieldEuro.setText(String.format("%.2f", usd *
usdToEuro));

            } else if (!textFieldEuro.getText().isEmpty()) {

               textFieldSGD.setText(String.format("%.2f", euro /
sgdToEuro));

               textFieldUSD.setText(String.format("%.2f", euro /
usdToEuro));
```

```
            }
         } catch (NumberFormatException ex) {
            textFieldSGD.setText("");
            textFieldUSD.setText("");
            textFieldEuro.setText("");
         }
      }
   }

   public static void main(String[] args) {
      new CurrencyConverter();
   }
}
```

SLIP.24

Q1) Create an abstract class 'Bank' with an abstract method 'getBalance'. Rs.100, Rs.150 and Rs.200 are deposited in banks A, B and C respectively. 'BankA', 'BankB' and 'BankC' are subclasses of class 'Bank', each having a method named 'getBalance'. Call this method by creating an object of each of the three classes.

ANS:

```
abstract class Bank {
    abstract int getBalance();
}


class BankA extends Bank {
    private int balance = 100;


    @Override
    int getBalance() {
        return balance;
    }
}
```

```
class BankB extends Bank {
   private int balance = 150;

   @Override
   int getBalance() {
      return balance;
   }
}

class BankC extends Bank {
   private int balance = 200;

   @Override
   int getBalance() {
      return balance;
   }
}

public class Main {
    public static void main(String[] args) {
        Bank bankA = new BankA();
        Bank bankB = new BankB();
        Bank bankC = new BankC();


        System.out.println("Balance in Bank A: Rs." +
bankA.getBalance());
        System.out.println("Balance in Bank B: Rs." +
bankB.getBalance());
        System.out.println("Balance in Bank C: Rs." +
bankC.getBalance());
    }
}
```

Q2) Program that displays three concentric circles where ever the user clicks the mouse on a frame. The program must exit when user clicks ‘X’ on the frame.

ANS:

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

```
}

public class ConcentricCircles extends Frame implements
MouseListener {

   int x, y;


   public ConcentricCircles() {

      super("Concentric Circles");

      setSize(400, 400);

      setVisible(true);

      addMouseListener(this);

   }


   public void paint(Graphics g) {

      super.paint(g);

      g.setColor(Color.GRAY);

      for (int i = 0; i < 3; i++) {

         g.drawOval(x - 50 + i * 10, y - 50 + i * 10, 100 - i * 20, 100 - i *
20);

         g.fillOval(x - 50 + i * 10, y - 50 + i * 10, 100 - i * 20, 100 - i * 20);

      }

   }
```

```
    public void mouseClicked(MouseEvent e) {
        x = e.getX();
        y = e.getY();
        repaint();
    }

    public void mousePressed(MouseEvent e) {}
    public void mouseReleased(MouseEvent e) {}
    public void mouseEntered(MouseEvent e) {}
    public void mouseExited(MouseEvent e) {}

    public static void main(String[] args) {
        new ConcentricCircles();
    }
}
```

SLIP.25

Q1) Create a class Student(rollno, name ,class, per), to read student information from the console and display them (Using BufferedReader class)

ANS:

```
import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;


class Student {

   private int rollno;

   private String name;

   private String studentClass;

   private float per;


   public void readStudentInfo() throws IOException {

      BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));


      System.out.print("Enter Roll Number: ");
```

```
        rollno = Integer.parseInt(reader.readLine());

        System.out.print("Enter Name: ");
        name = reader.readLine();

        System.out.print("Enter Class: ");
        studentClass = reader.readLine();

        System.out.print("Enter Percentage: ");
        per = Float.parseFloat(reader.readLine());
    }

    public void displayStudentInfo() {
        System.out.println("Roll Number: " + rollno);
        System.out.println("Name: " + name);
        System.out.println("Class: " + studentClass);
        System.out.println("Percentage: " + per);
    }
}
```

```
public class Main {

   public static void main(String[] args) {

      Student student = new Student();

      try {

         student.readStudentInfo();

         System.out.println("\nStudent Information:");

         student.displayStudentInfo();

      } catch (IOException e) {

         System.out.println("An error occurred while reading input.");

      }

   }

}
```

Q2) Create the following GUI screen using appropriate layout manager. Accept the name, class, hobbies from the user and display the selected options in a textbox.

ANS:

```
import java.awt.*;

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

```
public class GUIScreen extends Frame implements ActionListener {
   TextField textField;
   CheckboxGroup classGroup, hobbiesGroup;
   Checkbox cbFY, cbSY, cbTY, cbMusic, cbDance, cbSports;
   Label labelName, labelClass, labelHobbies;
   TextArea textArea;

   public GUIScreen() {
      super("GUI Screen");
      setLayout(new FlowLayout());

      labelName = new Label("Your Name");
      textField = new TextField(20);
      labelClass = new Label("Your Class");
      labelHobbies = new Label("Your Hobbies");

      classGroup = new CheckboxGroup();
      cbFY = new Checkbox("FY", classGroup, false);
      cbSY = new Checkbox("SY", classGroup, false);
```

```
cbTY = new Checkbox("TY", classGroup, false);

hobbiesGroup = new CheckboxGroup();

cbMusic = new Checkbox("Music", hobbiesGroup, false);

cbDance = new Checkbox("Dance", hobbiesGroup, false);

cbSports = new Checkbox("Sports", hobbiesGroup, false);


textArea = new TextArea(5, 20);


add(labelName);

add(textField);

add(labelClass);

add(cbFY);

add(cbSY);

add(cbTY);

add(labelHobbies);

add(cbMusic);

add(cbDance);

add(cbSports);

add(textArea);
```

```
setSize(300, 300);

setVisible(true);


textField.addActionListener(this);


cbFY.addItemListener(new ItemListener() {

   public void itemStateChanged(ItemEvent e) {

      updateTextArea();

   }

});

cbSY.addItemListener(new ItemListener() {

   public void itemStateChanged(ItemEvent e) {

      updateTextArea();

   }

});

cbTY.addItemListener(new ItemListener() {

   public void itemStateChanged(ItemEvent e) {

      updateTextArea();
```

```
        }
    });
    cbMusic.addItemListener(new ItemListener() {
        public void itemStateChanged(ItemEvent e) {
            updateTextArea();
        }
    });
    cbDance.addItemListener(new ItemListener() {
        public void itemStateChanged(ItemEvent e) {
            updateTextArea();
        }
    });
    cbSports.addItemListener(new ItemListener() {
        public void itemStateChanged(ItemEvent e) {
            updateTextArea();
        }
    });
}
```

```
    public void actionPerformed(ActionEvent e) {

        updateTextArea();

    }


    private void updateTextArea() {

        String name = textField.getText();

        String className =
classGroup.getSelectedCheckbox().getLabel();

        String hobbies = "";

        if (cbMusic.getState()) {

            hobbies += "Music, ";

        }

        if (cbDance.getState()) {

            hobbies += "Dance, ";

        }

        if (cbSports.getState()) {

            hobbies += "Sports, ";

        }

        hobbies = hobbies.substring(0, hobbies.length() - 2);
        textArea.setText("Name: " + name + "\nClass: " + className +
"\nHobbies: " + hobbies);
    }

    public static void main(String[] args) {
        new GUIScreen();
    }
}
```

SLIP.26

Q1) Define a Item class (item_number, item_name, item_price). Define a default and parameterized constructor. Keep a count of objects created. Create objects using parameterized constructor and display the object count after each object is created.(Use static member and method). Also display the contents of each object.

ANS:

```
class Item {
   private int itemNumber;
   private String itemName;
   private double itemPrice;
   private static int objectCount = 0;

   public Item() {
      itemNumber = 0;
      itemName = "Unknown";
      itemPrice = 0.0;
      objectCount++;
   }

   public Item(int itemNumber, String itemName, double itemPrice) {
      this.itemNumber = itemNumber;
      this.itemName = itemName;
```

```
        this.itemPrice = itemPrice;
        objectCount++;
    }

    public static int getObjectCount() {
        return objectCount;
    }

    public void displayItemInfo() {
        System.out.println("Item Number: " + itemNumber);
        System.out.println("Item Name: " + itemName);
        System.out.println("Item Price: " + itemPrice);
    }
}

public class Main {
    public static void main(String[] args) {
        Item item1 = new Item(101, "Laptop", 75000.0);
        System.out.println("Object Count: " + Item.getObjectCount());
        item1.displayItemInfo();

        Item item2 = new Item(102, "Smartphone", 30000.0);
        System.out.println("Object Count: " + Item.getObjectCount());
        item2.displayItemInfo();

        Item item3 = new Item();
        System.out.println("Object Count: " + Item.getObjectCount());
        item3.displayItemInfo();
    }
}
```

Q2) Define a class ‘Donor’ to store the below mentioned details of a blood donor. name, age, address, contactnumber, bloodgroup, date of last donation. Create ‘n’ objects of this class for all the regular donors at Pune. Write these objects to a file. Read these objects from the file and display only those donors’ details whose blood group is ‘A+ve’ and had not donated for the recent six months.

ANS:

```
import java.io.*;
import java.util.*;

class Donor implements Serializable {
    private String name;
    private int age;
    private String address;
    private String contactNumber;
    private String bloodGroup;

    private Date lastDonationDate;


    public Donor(String name, int age, String address, String
contactNumber, String bloodGroup, Date lastDonationDate) {

        this.name = name;

        this.age = age;

        this.address = address;

        this.contactNumber = contactNumber;

        this.bloodGroup = bloodGroup;

        this.lastDonationDate = lastDonationDate;

    }


    public String getBloodGroup() {

        return bloodGroup;

    }


    public Date getLastDonationDate() {

        return lastDonationDate;

    }


    public void displayDonorInfo() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Address: " + address);
        System.out.println("Contact Number: " + contactNumber);
        System.out.println("Blood Group: " + bloodGroup);
        System.out.println("Last Donation Date: " + lastDonationDate);
        System.out.println();
    }
}

public class Main {
    private static final String FILE_NAME = "donors.dat";

    public static void main(String[] args) {
        List<Donor> donors = new ArrayList<>();

        // Adding dummy donors for demonstration
        try {
            donors.add(new Donor("John Doe", 30, "123 Main St, Pune",
"9876543210", "A+ve", new GregorianCalendar(2023,
Calendar.MARCH, 15).getTime()));
```

```
            donors.add(new Donor("Jane Smith", 25, "456 Elm St, Pune",
"9876543211", "B+ve", new GregorianCalendar(2023,
Calendar.OCTOBER, 1).getTime()));
            donors.add(new Donor("Alice Johnson", 35, "789 Oak St,
Pune", "9876543212", "A+ve", new GregorianCalendar(2023,
Calendar.APRIL, 10).getTime()));
            donors.add(new Donor("Bob Brown", 40, "101 Pine St, Pune",
"9876543213", "O+ve", new GregorianCalendar(2023,
Calendar.SEPTEMBER, 5).getTime()));
            donors.add(new Donor("Charlie White", 28, "202 Maple St,
Pune", "9876543214", "A+ve", new GregorianCalendar(2023,
Calendar.FEBRUARY, 20).getTime()));

            writeDonorsToFile(donors);
            List<Donor> readDonors = readDonorsFromFile();
            displayEligibleDonors(readDonors);
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
    }

    private static void writeDonorsToFile(List<Donor> donors) throws
IOException {
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(FILE_NAME))) {
            oos.writeObject(donors);
        }
    }

    private static List<Donor> readDonorsFromFile() throws
IOException, ClassNotFoundException {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(FILE_NAME))) {
            return (List<Donor>) ois.readObject();
        }
    }

    private static void displayEligibleDonors(List<Donor> donors) {
        Date sixMonthsAgo = Calendar.getInstance();
        sixMonthsAgo.add(Calendar.MONTH, -6);

        System.out.println("Eligible Donors (A+ve and not donated in
the last 6 months):");
        for (Donor donor : donors) {
            if ("A+ve".equals(donor.getBloodGroup()) &&
donor.getLastDonationDate().before(sixMonthsAgo.getTime())) {
```

```
                donor.displayDonorInfo();
            }
        }
    }
}
```

SLIP.27

Q1) Define an Employee class with suitable attributes having getSalary() method, which returns salary withdrawn by a particular employee. Write a class Manager which extends a class Employee, override the getSalary() method, which will return salary of manager by adding traveling allowance, house rent allowance etc.

ANS:

```
class Employee {
  private String name;
  private double basicSalary;

  public Employee(String name, double basicSalary) {
    this.name = name;
    this.basicSalary = basicSalary;
  }

  public double getSalary() {
    return basicSalary;
  }

  public String getName() {
    return name;
  }
}

class Manager extends Employee {
  private double travelingAllowance;
```

```
    private double houseRentAllowance;

    public Manager(String name, double basicSalary, double travelingAllowance, double
houseRentAllowance) {
        super(name, basicSalary);
        this.travelingAllowance = travelingAllowance;
        this.houseRentAllowance = houseRentAllowance;
    }

    @Override
    public double getSalary() {
        return super.getSalary() + travelingAllowance + houseRentAllowance;
    }
}

public class Main {
    public static void main(String[] args) {
        Employee emp1 = new Employee("John Doe", 30000);
        Manager mgr1 = new Manager("Jane Smith", 50000, 10000, 15000);

        System.out.println("Employee Name: " + emp1.getName());
        System.out.println("Employee Salary: " + emp1.getSalary());

        System.out.println("Manager Name: " + mgr1.getName());
        System.out.println("Manager Salary: " + mgr1.getSalary());
    }
```

}

Q2) Write a program to accept a string as command line argument and check whether it is a file or directory. Also perform operations as follows:

i)If it is a directory,delete all text files in that directory. Confirm delete operation from user before deleting text files. Also, display a count showing the number of files deleted, if any, from the directory.

ii)If it is a file display various details of that file

ANS:

```
import java.io.File;

import java.io.IOException;

import java.util.Scanner;


public class FileDirectoryCheck {

   public static void main(String[] args) {

      if (args.length != 1) {

         System.out.println("Please provide a file or directory path as a
command line argument.");

         return;

      }


      String path = args[0];

      File file = new File(path);


      if (file.isDirectory()) {
            System.out.println("It is a directory: " + file.getAbsolutePath());
            deleteTextFiles(file);
        } else if (file.isFile()) {
            System.out.println("It is a file: " + file.getAbsolutePath());
            displayFileDetails(file);
        } else {
            System.out.println("The specified path is neither a file nor a
directory.");
        }
    }

    private static void deleteTextFiles(File directory) {
        File[] files = directory.listFiles();
        if (files == null || files.length == 0) {
            System.out.println("No files found in the directory.");
            return;
        }

        int countDeleted = 0;
        Scanner scanner = new Scanner(System.in);

        for (File file : files) {
```

```
        if (file.isFile() && file.getName().endsWith(".txt")) {
          System.out.print("Do you want to delete the file " +
file.getName() + "? (yes/no): ");
          String response = scanner.nextLine();
          if (response.equalsIgnoreCase("yes")) {
            if (file.delete()) {
              countDeleted++;
              System.out.println(file.getName() + " deleted.");
            } else {
              System.out.println("Failed to delete " + file.getName());
            }
          }
        }
      }
      System.out.println("Total text files deleted: " + countDeleted);
    }
    private static void displayFileDetails(File file) {
      System.out.println("File Name: " + file.getName());
      System.out.println("Absolute Path: " + file.getAbsolutePath());
      System.out.println("File Size: " + file.length() + " bytes");
      System.out.println("Readable: " + file.canRead());
```

```
        System.out.println("Writable: " + file.canWrite());

        System.out.println("Executable: " + file.canExecute());

        System.out.println("Last Modified: " + new
java.util.Date(file.lastModified()));

    }

}
```

SLIP.28

Q1) Write a program that reads on file name from the user, then displays information about whether the file exists, whether the file is readable, whether the file is writable, the type of file and the length of the file in bytes.

ANS:

```
import java.io.File;

import java.io.IOException;

import java.util.Scanner;


public class FileDirectoryCheck {

   public static void main(String[] args) {

      if (args.length != 1) {

         System.out.println("Please provide a file or directory path as a
command line argument.");

         return;

      }


      String path = args[0];

      File file = new File(path);


      if (file.isDirectory()) {
        System.out.println("It is a directory: " + file.getAbsolutePath());
        deleteTextFiles(file);
      } else if (file.isFile()) {
        System.out.println("It is a file: " + file.getAbsolutePath());
        displayFileDetails(file);
      } else {
        System.out.println("The specified path is neither a file nor a
directory.");
      }
    }

    private static void deleteTextFiles(File directory) {
      File[] files = directory.listFiles();
      if (files == null || files.length == 0) {
        System.out.println("No files found in the directory.");
        return;
      }

      int countDeleted = 0;
      Scanner scanner = new Scanner(System.in);

      for (File file : files) {
        if (file.isFile() && file.getName().endsWith(".txt")) {
            System.out.print("Do you want to delete the file " +
file.getName() + "? (yes/no): ");
            String response = scanner.nextLine();
            if (response.equalsIgnoreCase("yes")) {
                if (file.delete()) {
                    countDeleted++;
                    System.out.println(file.getName() + " deleted.");
                } else {
                    System.out.println("Failed to delete " + file.getName());
                }
            }
        }
    }

    System.out.println("Total text files deleted: " + countDeleted);
  }

  private static void displayFileDetails(File file) {
    System.out.println("File Name: " + file.getName());
    System.out.println("Absolute Path: " + file.getAbsolutePath());
    System.out.println("File Size: " + file.length() + " bytes");
```

```
        System.out.println("Readable: " + file.canRead());
        System.out.println("Writable: " + file.canWrite());
        System.out.println("Executable: " + file.canExecute());
        System.out.println("Last Modified: " + new
java.util.Date(file.lastModified()));
    }
}
```

Q2) Write a program called SwingTemperatureConverter to convert temperature values between Celsius and Fahrenheit. User can enter either the Celsius or the Fahrenheit value, in floating-point number. Hints: To display a floating-point number in a specific format (e.g., 1 decimal place), use the static method String.format(), which has the same form as printf(). For example, String.format("%.1f", 1.234) returns String "1.2".

ANS:

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

public class SwingTemperatureConverter extends JFrame implements ActionListener {
    private JTextField celsiusField, fahrenheitField;
    private JButton celsiusToFahrenheitButton, fahrenheitToCelsiusButton;

    public SwingTemperatureConverter() {
        super("Temperature Converter");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new GridLayout(3, 2));
        JLabel celsiusLabel = new JLabel("Celsius:");
        celsiusField = new JTextField(10);
        JLabel fahrenheitLabel = new JLabel("Fahrenheit:");
        fahrenheitField = new JTextField(10);

        celsiusToFahrenheitButton = new JButton("Celsius to Fahrenheit");
        fahrenheitToCelsiusButton = new JButton("Fahrenheit to Celsius");

        add(celsiusLabel);
        add(celsiusField);
        add(fahrenheitLabel);
        add(fahrenheitField);
        add(celsiusToFahrenheitButton);
        add(fahrenheitToCelsiusButton);

        celsiusToFahrenheitButton.addActionListener(this);
        fahrenheitToCelsiusButton.addActionListener(this);

        setSize(300, 150);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == celsiusToFahrenheitButton) {
            try {
                double celsius = Double.parseDouble(celsiusField.getText());
                double fahrenheit = (celsius * 9 / 5) + 32;
                fahrenheitField.setText(String.format("%.1f", fahrenheit));
            } catch (NumberFormatException ex) {
                JOptionPane.showMessageDialog(this, "Invalid Celsius value.", "Error",
    JOptionPane.ERROR_MESSAGE);
            }
        } else if (e.getSource() == fahrenheitToCelsiusButton) {
            try {
                double fahrenheit = Double.parseDouble(fahrenheitField.getText());
                double celsius = (fahrenheit - 32) * 5 / 9;
                celsiusField.setText(String.format("%.1f", celsius));
            } catch (NumberFormatException ex) {
                JOptionPane.showMessageDialog(this, "Invalid Fahrenheit value.", "Error",
    JOptionPane.ERROR_MESSAGE);
            }
        }
    }

    public static void main(String[] args) {
        new SwingTemperatureConverter();
    }
    }
```

SLIP.29

Q1) Write a program to create a class Customer(custno,custname,contactnumber,custaddr). Write a method to search the customer name with given contact number and display the details.

ANS:

```
import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;


class Customer {

  private int custno;

  private String custname;

  private String contactnumber;

  private String custaddr;


  public Customer(int custno, String custname, String contactnumber, String custaddr) {

    this.custno = custno;

    this.custname = custname;

    this.contactnumber = contactnumber;

    this.custaddr = custaddr;

  }


  public String getContactnumber() {

    return contactnumber;

  }
```

```
    public void displayDetails() {
        System.out.println("Customer Number: " + custno);
        System.out.println("Customer Name: " + custname);
        System.out.println("Contact Number: " + contactnumber);
        System.out.println("Customer Address: " + custaddr);
    }
}

public class CustomerSearch {
    private List<Customer> customers;

    public CustomerSearch() {
        customers = new ArrayList<>();
    }

    public void addCustomer(Customer customer) {
        customers.add(customer);
    }

    public void searchCustomerByContactNumber(String contactNumber) {
        boolean found = false;
        for (Customer customer : customers) {
            if (customer.getContactnumber().equals(contactNumber)) {
                customer.displayDetails();
                found = true;
                break;
            }
        }
        if (!found) {
            System.out.println("Customer with contact number " + contactNumber + " not found.");
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        CustomerSearch customerSearch = new CustomerSearch();
        customerSearch.addCustomer(new Customer(1, "Alice", "1234567890", "123 Main St"));
        customerSearch.addCustomer(new Customer(2, "Bob", "0987654321", "456 Elm St"));
        customerSearch.addCustomer(new Customer(3, "Charlie", "1122334455", "789 Maple St"));
        System.out.print("Enter contact number to search for customer: ");
        String contactNumber = scanner.nextLine();
        customerSearch.searchCustomerByContactNumber(contactNumber);
        scanner.close();
    }
}
```

Q2) Write a program to create a super class Vehicle having members Company and price. Derive two different classes LightMotorVehicle(mileage) and HeavyMotorVehicle (capacity_in_tons). Accept the information for "n" vehicles and display the information in appropriate form. While taking data, ask user about the type of vehicle first.

ANS:

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

class Vehicle {
    protected String company;
    protected double price;

    public Vehicle(String company, double price) {
        this.company = company;
        this.price = price;
    }

    public void displayInfo() {
        System.out.println("Company: " + company);
        System.out.println("Price: " + price);
    }
}

class LightMotorVehicle extends Vehicle {
    private double mileage;

    public LightMotorVehicle(String company, double price, double mileage) {
        super(company, price);
        this.mileage = mileage;
    }
    @Override
    public void displayInfo() {
        super.displayInfo();
        System.out.println("Type: Light Motor Vehicle");
        System.out.println("Mileage: " + mileage + " km/l");
    }
}

class HeavyMotorVehicle extends Vehicle {
    private double capacityInTons;

    public HeavyMotorVehicle(String company, double price, double capacityInTons) {
        super(company, price);
        this.capacityInTons = capacityInTons;
    }

    @Override
    public void displayInfo() {
        super.displayInfo();
        System.out.println("Type: Heavy Motor Vehicle");
        System.out.println("Capacity: " + capacityInTons + " tons");
    }
}

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

```
Scanner scanner = new Scanner(System.in);

List<Vehicle> vehicles = new ArrayList<>();

System.out.print("Enter the number of vehicles: ");

int n = scanner.nextInt();

scanner.nextLine();


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

    System.out.print("Enter vehicle type (1 for Light, 2 for Heavy): ");

    int type = scanner.nextInt();

    scanner.nextLine();

    System.out.print("Enter company name: ");

    String company = scanner.nextLine();

    System.out.print("Enter price: ");

    double price = scanner.nextDouble();


    if (type == 1) {

        System.out.print("Enter mileage: ");

        double mileage = scanner.nextDouble();

        vehicles.add(new LightMotorVehicle(company, price, mileage));

    } else if (type == 2) {

        System.out.print("Enter capacity in tons: ");

        double capacityInTons = scanner.nextDouble();

        vehicles.add(new HeavyMotorVehicle(company, price, capacityInTons));

    }

}
```

```
        System.out.println("\nVehicle Information:");
        for (Vehicle vehicle : vehicles) {
            vehicle.displayInfo();
            System.out.println();
        }

        scanner.close();
    }
}
```

SLIP.30

Q1) Write program to define class Person with data member as Personname,Aadharno, Panno. Accept information for 5 objects and display appropriate information (use this keyword).

ANS:

```
import java.util.Scanner;


class Person {
    private String personName;
    private String aadharNo;
    private String panNo;


    public Person(String personName, String aadharNo, String panNo) {
        this.personName = personName;
        this.aadharNo = aadharNo;
        this.panNo = panNo;
    }


    public void displayInfo() {
        System.out.println("Name: " + this.personName);
        System.out.println("Aadhar No: " + this.aadharNo);
        System.out.println("PAN No: " + this.panNo);
        System.out.println();
    }
}
```

```
}

public class PersonInfo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Person[] persons = new Person[5];

        for (int i = 0; i < 5; i++) {
            System.out.println("Enter details for person " + (i + 1) + ":");
            System.out.print("Enter Name: ");
            String name = scanner.nextLine();
            System.out.print("Enter Aadhar No: ");
            String aadharNo = scanner.nextLine();
            System.out.print("Enter PAN No: ");
            String panNo = scanner.nextLine();

            persons[i] = new Person(name, aadharNo, panNo);
            System.out.println();
        }

        System.out.println("Person Information:");
        for (Person person : persons) {
            person.displayInfo();
        }

        scanner.close();
    }
}
```

Q2) Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Number1 and Number2. The division of Number1 and Number2 is displayed in the Result field when the Divide button is clicked. If Number1 or Number2 were not an integer, the program would throw a NumberFormatException. If Number2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

ANS:

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;


public class IntegerDivisionCalculator extends JFrame {

    private JTextField number1Field;

    private JTextField number2Field;

    private JTextField resultField;


    public IntegerDivisionCalculator() {

        setTitle("Integer Division Calculator");

        setSize(400, 200);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setLayout(new FlowLayout());


        JLabel label1 = new JLabel("Number 1:");

        number1Field = new JTextField(10);

        JLabel label2 = new JLabel("Number 2:");
```

```
number2Field = new JTextField(10);
JButton divideButton = new JButton("Divide");
JLabel resultLabel = new JLabel("Result:");
resultField = new JTextField(10);
resultField.setEditable(false);

divideButton.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent e) {
    try {
      int number1 = Integer.parseInt(number1Field.getText());
      int number2 = Integer.parseInt(number2Field.getText());
      int result = number1 / number2;
      resultField.setText(String.valueOf(result));
    } catch (NumberFormatException ex) {
      JOptionPane.showMessageDialog(null, "Please enter valid integers.");
    } catch (ArithmeticException ex) {
      JOptionPane.showMessageDialog(null, "Division by zero is not allowed.");
    }
  }
});

add(label1);
add(number1Field);
add(label2);
add(number2Field);
add(divideButton);
```

```
        add(resultLabel);
        add(resultField);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new IntegerDivisionCalculator().setVisible(true);
            }
        });
    }
}
```