Slip No. 1

1.write java program to print all prime numbers in an array of n elements (Using Commandline Arguments)

```
import java.util.Scanner;

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


        // Input the size of the array
        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();


        // Input the elements of the array
        int[] arr = new int[n];
        System.out.println("Enter the elements of the array: ");
        for (int i = 0; i < n; i++)
{
            arr[i] = scanner.nextInt();
        }


        // Print all prime numbers in the array
        System.out.println("Prime numbers in the array are: ");
```

```
        for (int i = 0; i < n; i++)
{
            if (isPrime(arr[i]))
{
                System.out.print(arr[i] + " ");
            }
        }
    }

    // Method to check if a number is prime
    public static boolean isPrime(int num)
{
        if (num <= 1)
{
            return false;
        }
        for (int i = 2; i <= Math.sqrt(num); i++)
{
            if (num % i == 0)
{
                return false;
            }
        }
        return true;
```

```
    }
}
```

2. Define abstract class staff with protected members id and name, Define parameterized constructor. Define one subclass OfficeStaff with member department. create n objects of OfficeStaff and display all the details.

```
package PracSlipsSemIV;

import java.util.Scanner;

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

// Parameterized constructor
public Staff(int id, String name)
{
    this.id = id;
    this.name = name;
}
```

```
 // Abstract method to display details (can be implemented in
subclasses)
 public abstract void displayDetails();
}
//Subclass OfficeStaff
class OfficeStaff extends Staff
{
 private String department;

 // Parameterized constructor
 public OfficeStaff(int id, String name, String department)
{
    super(id, name);   // Calling the constructor of the superclass
    this.department = department;
}

// Implementing the abstract method to display details
@Override
public void displayDetails()
{
    System.out.println("ID: " + id);
    System.out.println("Name: " + name);
    System.out.println("Department: " + department);
```

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

//Main class to create and display OfficeStaff objects
public class OfficeStaffMain
{
 public static void main(String[] args)
 {
    Scanner scanner = new Scanner(System.in);

    // Input number of OfficeStaff objects to create
    System.out.print("Enter the number of OfficeStaff members: ");
    int n = scanner.nextInt();
    scanner.nextLine();  // Consume newline

    // Create an array of OfficeStaff objects
    OfficeStaff[] staffArray = new OfficeStaff[n];

    // Input details for each OfficeStaff member and create objects
    for (int i = 0; i < n; i++)
{
        System.out.println("Enter details for OfficeStaff " + (i + 1) + ":");
        System.out.print("Enter ID: ");
```

```
        int id = scanner.nextInt();
        scanner.nextLine();  // Consume newline
        System.out.print("Enter Name: ");
        String name = scanner.nextLine();
        System.out.print("Enter Department: ");
        String department = scanner.nextLine();

        // Create a new OfficeStaff object and add it to the array
        staffArray[i] = new OfficeStaff(id, name, department);
    }

    // Display details of all OfficeStaff members
    System.out.println("\nDetails of OfficeStaff members:");
    for (OfficeStaff staff : staffArray)
    {
        staff.displayDetails();
    }
}
}
```

Slip No.2

1.write a java program to read 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 body mass divided by the square of their height.

```
import java.util.*;
public class BMICalculator {
    public static void main(String[] args)
{
        // Check if the correct number of arguments is provided
        if (args.length != 4)
{
            System.out.println("Please provide the first name, last name,
weight (in kg), and height (in meters) as command-line arguments.");
            return;
        }

        // Parse the command-line arguments
        String firstName = args[0];
        String lastName = args[1];
        double weight = Double.parseDouble(args[2]); // Weight in
kilograms
        double height = Double.parseDouble(args[3]); // Height in meters

        // Calculate the BMI
        double bmi = weight / (height * height);

        // Display the results
        System.out.println("Name: " + firstName + " " + lastName);
        System.out.println("Weight: " + weight + " kg");
```

```
        System.out.println("Height:"+height+"meters");
        System.out.println("BMI:"+bmi);
    }
}
```

2. 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.

```
package ChaTwo;

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

class CricketPlayer
{
    private String name;
    private int no_of_innings;
    private int no_of_times_notout;
    private int total_runs;
    private double bat_avg;


    public CricketPlayer(String name, int no_of_innings, int
no_of_times_notout, int total_runs)
    {
        this.name = name;
```

```
this.no_of_innings=no_of_innings;

this.no_of_times_notout=no_of_times_notout;

this.total_runs=total_runs;

this.bat_avg=avg();
}


private double avg() {
    int outs = no_of_innings - no_of_times_notout;
    if (outs > 0)
    {
        return (double) total_runs / outs;
    }
    else
    {
        return total_runs;
    }
}


public double getBatAvg()
{
    return bat_avg;
}


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

```
public void display()
{
    System.out.println("Name:" +name);

    System.out.println("Innings:" +no_of_innings);

    System.out.println("Not Out:" +no_of_times_notout);

    System.out.println("Total Runs:" +total_runs);

    System.out.println("Batting Average:" +bat_avg);

    System.out.println();
}

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

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

    int n = scanner.nextInt();


    scanner.nextLine();

    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.nextLine();
```

```
            System.out.print("Number of Innings: ");

            int no_of_innings = scanner.nextInt();

            System.out.print("Number of Times Not Out: ");

            int no_of_times_notout = scanner.nextInt();

            System.out.print("Total Runs: ");

            int total_runs = scanner.nextInt();

            scanner.nextLine();


            players[i] = new CricketPlayer(name, no_of_innings,
    no_of_times_notout, total_runs);
        }


        CricketPlayer.sortPlayers(players);


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

        scanner.close();
    }
}
```

Slip No.3

1. write java program to accept 'n' name of cities from the user and sort them in ascending order.

```
package PracSlipsSemIV;

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

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

        // Input the number of cities
        System.out.print("Enter the number of cities: ");
        int n = scanner.nextInt();
        scanner.nextLine();  // Consume newline

        // Array to store city names
        String[] cities = new String[n];

        // Input city names
        System.out.println("Enter the names of the cities:");
        for (int i = 0; i < n; i++)
        {
            System.out.print("City " + (i + 1) + ": ");
            cities[i] = scanner.nextLine();
        }

        // Sort the cities in ascending order
        Arrays.sort(cities);
```

```
// Display the sorted list of cities
System.out.println("\nCities in ascending order:");
for (String city : cities)
{
    System.out.println(city);
}
    }
}
```

2. Define a classpatient(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.

```
package Exception;

import java.io.*;

class CovidException extends Exception

{

public CovidException()

{

System.out.println("Patient is Covid Positive and needs to be hospitalized");

}

}

class Patient
```

```
{
String name;
int age;
double level,hrct;
public Patient(String name,int age,double level,double hrct)
{
this.name=name;
this.age=age;
this.level=level;
this.hrct=hrct;
}
public static void main(String[] args)throws IOException
{
String name;
int age;
double level,hrct;
BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));
System.out.println("Enter name: ");
name=br.readLine();
System.out.println("Enter the age: ");
age=Integer.parseInt(br.readLine());
System.out.println("Oxygen level: ");
level=Double.parseDouble(br.readLine());
```

```
System.out.println("HRCT report: ");


hrct=Double.parseDouble(br.readLine());
Patient ob=new Patient(name,age,level,hrct);
try
{
if(ob.level<95&&ob.hrct>10)

throw new CovidException();


else
System.out.println("Patient Info: \n"+"Name: "+ob.name+"\nAge:
"+ob.age+"\nHRCT report: "+ob.hrct+"\nOxygen level:"+ob.level);
}
catch(CovidException e)
{
}
}
}
```

Slip No.4

1.write a java program to print an array after changing the rows and column of a given two diamensional array..

```
package PracSlipsSemIV;
```

```
import java.util.Scanner;

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


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


        // Declare a 2D array (matrix)
        int[][] matrix = new int[rows][cols];


        // Input the elements of the matrix
        System.out.println("Enter the elements of the matrix:");
            for (int i = 0; i < rows; i++)
{
                for (int j = 0; j < cols; j++)
{
```

```
            System.out.print("Element at [" + i + "][" + j + "]: ");
            matrix[i][j] = scanner.nextInt();
        }
    }

    // Create the transposed matrix
    int[][] transposedMatrix = new int[cols][rows];

    // Transpose the matrix (swap rows and columns)
    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < cols; j++)
    {
            transposedMatrix[j][i] = matrix[i][j];
        }
    }

    // Display the transposed matrix
    System.out.println("\nTransposed Matrix:");
    for (int i = 0; i < cols; i++)
    {
        for (int j = 0; j < rows; j++)
```

```
{
            System.out.print(transposedMatrix[i][j]+" ");
        }
        System.out.println();
    }
}
```

2. Write a Java 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..

```
package ChapFive;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class LoginScreen extends Frame implements
ActionListener
{
    private TextField usernameField, passwordField;
    private Label messageLabel;
    private Button loginButton, clearButton;
    private int loginAttempts = 3;
```

```
public LoginScreen()
{
    setTitle("Login Screen");
    setSize(400, 300);
    setLayout(new GridLayout(5, 2));

    Label usernameLabel = new Label("Username:");
    Label passwordLabel = new Label("Password:");
    usernameField = new TextField();
    passwordField = new TextField();
    passwordField.setEchoChar('*');

    loginButton = new Button("Login");
    clearButton = new Button("Clear");
    messageLabel = new Label();

    loginButton.addActionListener(this);
    clearButton.addActionListener(this);

    add(usernameLabel);
    add(usernameField);
    add(passwordLabel);
    add(passwordField);
    add(new Label("")); // empty cell
    add(loginButton);
    add(new Label("")); // empty cell
    add(clearButton);
    add(messageLabel);
```

```
        addWindowListener(new
java.awt.event.WindowAdapter()
        {
            public void
windowClosing(java.awt.event.WindowEvent
windowEvent)
            {
                System.exit(0);
            }
        });
    }

    @Override
    public void actionPerformed(ActionEvent e)
    {
        if (e.getSource() == loginButton)
        {
            try
            {
                checkLogin();
            }
            catch (LoginException ex)
            {
                messageLabel.setText(ex.getMessage());
                loginAttempts--;
                if (loginAttempts == 0)
{
                    loginButton.setEnabled(false);
```

```
                messageLabel.setText("Account locked.");
            }
        }
    }
    else if (e.getSource() == clearButton)
    {
        usernameField.setText("");
        passwordField.setText("");
        messageLabel.setText("");
    }
}

private void checkLogin() throws LoginException
{
    String username = usernameField.getText();
    String password = passwordField.getText();

    if (!username.equals(password))
    {
        throw new LoginException("Username and
Password do not match.");
    }
else
{
    messageLabel.setText("Login successful!");
    }
}

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

```
    {
        LoginScreen loginScreen = new LoginScreen();
        loginScreen.setVisible(true);
    }
}
```

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

Slip No.5

1.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.

```
package Inheritance;

class Continent
{
String continentName;


public Continent(String continentName)
```

```
{
    this.continentName = continentName;
}

public String getContinentName()
{
    return continentName;
}
}

class Country extends Continent
{
String countryName;

public Country(String continentName, String countryName)
{
    super(continentName);
    this.countryName = countryName;
}

public String getCountryName()
{
    return 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 String getStateName()
{
    return stateName;
}
public String getPlaceName()
{
    return placeName;
}
public void display()
{
```

```
System.out.println("Place: " + getPlaceName());

System.out.println("State: " + getStateName());

System.out.println("Country: " + getCountryName());
        System.out.println("Continent: " + getContinentName());
    }
}

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

2.write java menu driven program to perform following operation on multidiamensional array.

1.additional 2.Multiplication 3.Exit

```
package PracSlipsSemIV;

import java.util.Scanner;
```

```
public class ArrayOperations
{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int choice;
        do
{
            System.out.println("Menu:");
            System.out.println("1. Addition of Two Matrices");
            System.out.println("2. Multiplication of Two Matrices");
            System.out.println("3. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch (choice)
{
                case 1:
                    addMatrices(sc);
                    break;
                case 2:
                    multiplyMatrices(sc);
```

```
            break;
        case 3:
            System.out.println("Exiting...");
            break;
        default:
                    System.out.println("Invalid choice! Please try
again.");
            }
        } while (choice != 3);
    }
    // Method to add two matrices
    public static void addMatrices(Scanner sc)
{
        System.out.print("Enter the number of rows and columns
for the matrices: ");
        int rows = sc.nextInt();
        int cols = sc.nextInt();

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

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

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

        // Adding the matrices
        for (int i = 0; i < rows; i++)
```

```
{
            for (int j = 0; j < cols; j++)
{
                sum[i][j] = matrix1[i][j] + matrix2[i][j];
            }
        }

        System.out.println("Sum of the matrices:");
        for (int i = 0; i < rows; i++)
{
            for (int j = 0; j < cols; j++)
{
                System.out.print(sum[i][j] + " ");
            }
            System.out.println();
        }
    }

    // Method to multiply two matrices
    public static void multiplyMatrices(Scanner sc)
{
```

```
        System.out.print("Enter the number of rows and columns
for the first matrix: ");
        int rows1 = sc.nextInt();
        int cols1 = sc.nextInt();

        System.out.print("Enter the number of rows and columns
for the second matrix: ");
        int rows2 = sc.nextInt();
        int cols2 = sc.nextInt();


        if (cols1 != rows2)
{
 System.out.println("Matrix multiplication not possible. Number
of columns in the first matrix must equal the number of rows in
the second matrix.");
            return;
        }


        int[][] matrix1 = new int[rows1][cols1];
        int[][] matrix2 = new int[rows2][cols2];
        int[][] product = new int[rows1][cols2];

        System.out.println("Enter elements of the first matrix:");
```

```
        for (int i = 0; i < rows1; i++)
{
            for (int j = 0; j < cols1; j++)
 {
                matrix1[i][j] = sc.nextInt();
            }
        }

        System.out.println("Enter elements of the second
matrix:");
        for (int i = 0; i < rows2; i++)
 {
            for (int j = 0; j < cols2; j++)
{
                matrix2[i][j] = sc.nextInt();
            }
        }

        // Multiplying the matrices
        for (int i = 0; i < rows1; i++)
{
            for (int j = 0; j < cols2; j++)
```

```
{
                for (int k = 0; k < cols1; k++)
{
                    product[i][j] += matrix1[i][k] * matrix2[k][j];
                }
            }
        }

        System.out.println("Product of the matrices:");
        for (int i = 0; i < rows1; i++)
{
            for (int j = 0; j < cols2; j++)
{
                System.out.print(product[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Slip No.6

1. Write a java Program to display Employee(Empid,Empname,Empdesignation,Empsal) information using toString()..

```
package PracSlipsSemIV;

class Employee
{
    // Attributes of the Employee class
    private int empId;
    private String empName;
    private String empDesignation;
    private double empSal;

    // Constructor to initialize the Employee object
    public Employee(int empId, String empName, String empDesignation, double empSal)
    {
        this.empId = empId;
        this.empName = empName;
        this.empDesignation = empDesignation;
        this.empSal = empSal;
    }

    // Overriding the toString() method to display employee information
    @Override
    public String toString()
    {
```

```
        return "Employee Details:\n" +
                "ID: " + empId + "\n" +
                "Name: " + empName + "\n" +
                "Designation: " + empDesignation + "\n" +
                "Salary: " + empSal;
    }

    // Main method to test the Employee class
    public static void main(String[] args)
{
        // Creating an Employee object
        Employee emp = new Employee(101, "John",
"Software Engineer", 75000.0);

        // Displaying the Employee information using the
    overridden toString() method
            System.out.println(emp);
        }
    }
```

2. Create an abstract class 'order' having members id,description. Create two Subclasses 'PurchaseOrder" And "SalesOrder" Having Memebers customer name and vendor name. Define Method Accept And display in all cases.. create 3 Object each of this purchase Order and Sales order and accept and display details…

```
package PracSlipsSemIV;

import java.util.Scanner;
abstract class Order
{
protected int id;
protected String description;

// Abstract methods to be implemented by subclasses
abstract void accept(Scanner sc);

abstract void display();
}

//Subclass PurchaseOrder
class PurchaseOrder extends Order

{
private String vendorName;

@Override
void accept(Scanner sc)
{
    System.out.print("Enter Purchase Order ID: ");
```

```
id = sc.nextInt();
sc.nextLine(); // Consume newline
System.out.print("Enter Purchase Order Description: ");
description = sc.nextLine();
System.out.print("Enter Vendor Name: ");
    vendorName = sc.nextLine();
}

@Override
void display()
{
    System.out.println("Purchase Order Details:");
    System.out.println("ID: " + id);
    System.out.println("Description: " + description);
    System.out.println("Vendor Name: " + vendorName);
    System.out.println();
 }
}

//Subclass SalesOrder
class SalesOrder extends Order
{
 private String customerName;
```

```
@Override
void accept(Scanner sc)
{
    System.out.print("Enter Sales Order ID: ");
    id = sc.nextInt();
    sc.nextLine(); // Consume newline
    System.out.print("Enter Sales Order Description: ");
    description = sc.nextLine();
    System.out.print("Enter Customer Name: ");
    customerName = sc.nextLine();
}

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

//Main class
```

```
public class Main
{
 public static void main(String[] args)
{
    Scanner sc = new Scanner(System.in);
    // Create and accept details for 3 Purchase Orders
    PurchaseOrder[] purchaseOrders = new PurchaseOrder[3];
    for (int i = 0; i < purchaseOrders.length; i++)
{
        purchaseOrders[i] = new PurchaseOrder();
        System.out.println("Enter details for Purchase Order " + (i + 1) + ":");
        purchaseOrders[i].accept(sc);
    }

    // Create and accept details for 3 Sales Orders
    SalesOrder[] salesOrders = new SalesOrder[3];
    for (int i = 0; i < salesOrders.length; i++)
{
        salesOrders[i] = new SalesOrder();
        System.out.println("Enter details for Sales Order " + (i + 1) + ":");
        salesOrders[i].accept(sc);
}
```

```
        // Display details of all Purchase Orders
        System.out.println("Displaying Purchase Orders:");
        for (PurchaseOrder po : purchaseOrders)
{
            po.display();
        }

        // Display details of all Sales Orders
        System.out.println("Displaying Sales Orders:");
        for (SalesOrder so : salesOrders)
{
            so.display();
        }
    }
}
```

Slip No.7

1.Design Class for 'Bank' .'Bank' class should support following operations

1. Deposit certain amount into an account.

2. Withdraw a certain amount into account

3. Return balance value specifying amount with details

```
package PracSlipsSemIV;
```

```
import java.util.Scanner;
class Bank
{
    // Attributes of the Bank class
    private String accountHolderName;
    private int accountNumber;
    private double balance;

    // Constructor to initialize the Bank object
    public Bank(String accountHolderName, int accountNumber,
double initialBalance)
    {
        this.accountHolderName = accountHolderName;
        this.accountNumber = accountNumber;
        this.balance = initialBalance;
    }

    // Method to deposit money into the account
    public void deposit(double amount)
{
        if (amount > 0)
{
```

```
            balance += amount;
            System.out.println("Successfully deposited $" +
amount);
        } else
{
            System.out.println("Deposit amount must be positive.");
    }
}

// Method to withdraw money from the account
public void withdraw(double amount)
{
        if (amount > 0 && amount <= balance)
{
            balance -= amount;
            System.out.println("Successfully withdrew $" +
amount);
        }
        else if (amount > balance)
{
            System.out.println("Insufficient balance. Withdrawal
failed.");
        }
```

```
else
{
        System.out.println("Withdrawal amount must be
positive.");
    }
    }

    // Method to return the current balance with details
    public void displayBalance()
{
        System.out.println("Account Holder: " +
accountHolderName);
        System.out.println("Account Number: " +
accountNumber);
        System.out.println("Current Balance: $" + balance);
    }

    // Main method to test the Bank class
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);

        // Creating a bank account object
```

```
        System.out.print("Enter account holder name: ");
        String name = sc.nextLine();
        System.out.print("Enter account number: ");
        int accountNumber = sc.nextInt();
        System.out.print("Enter initial balance: ");
        double initialBalance = sc.nextDouble();

        Bank account = new Bank(name, accountNumber,
initialBalance);

        // Menu-driven operations
        int choice;
        do {
            System.out.println("\nBank Operations:");
            System.out.println("1. Deposit");
            System.out.println("2. Withdraw");
            System.out.println("3. Display Balance");
            System.out.println("4. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch (choice)
```

```
{
                case 1:
                    System.out.print("Enter the amount to deposit: ");
                    double depositAmount = sc.nextDouble();
                    account.deposit(depositAmount);
                    break;
                case 2:
                    System.out.print("Enter the amount to withdraw: ");
                    double withdrawAmount = sc.nextDouble();
                    account.withdraw(withdrawAmount);
                    break;
                case 3:
                    account.displayBalance();
                    break;
                case 4:
                    System.out.println("Exiting...");
                    break;
                default:
                    System.out.println("Invalid choice! Please try
again.");
                }
            } while (choice != 4);
```

```
        sc.close();
    }
}
```

2. Write a program to read a text file "sample.txt" and display the contents of a file in reverse order and also original contents change the case (display in upper case).

```
package ChapFive;
import java.io.*;
import java.util.*;
class Read
{
public static void main(String args[]) throws IOException
{
FileReader f = new
FileReader("C:/Users/vrush/OneDrive/Desktop/New Text
Document.txt");
Scanner sc = new Scanner(f);
String CH,CH2;
while(sc.hasNext())
{
StringBuilder CH1 = new StringBuilder();
CH = sc.next();
CH2=CH.toUpperCase();
CH1.append(CH2);
```

```
CH1.reverse();
System.out.println(CH1);
}
f.close();
}
}
```

Slip No .8

1. **Write Java program Create Class Sphere to calculate volume and surface area of sphere..**

```
import java.util.Scanner;

class Sphere {
    // Attribute of the Sphere class
    private double radius;

    // Constructor to initialize the Sphere object
    public Sphere(double radius) {
        this.radius = radius;
    }

    // Method to calculate the volume of the sphere
    public double calculateVolume() {
        return (4.0 / 3.0) * Math.PI * Math.pow(radius,
3);
```

```
    }

    // Method to calculate the surface area of the
sphere
    public double calculateSurfaceArea() {
        return 4.0 * Math.PI * Math.pow(radius, 2);
    }

    // Method to display the volume and surface area
    public void display() {
        System.out.println("Radius of the sphere: " +
radius);
        System.out.println("Volume of the sphere: " +
calculateVolume());
            System.out.println("Surface area of the sphere:
" + calculateSurfaceArea());
    }

    // Main method to test the Sphere class
    public static void main(String[] args)
{
        Scanner sc = new Scanner(System.in);

        // Prompt user to enter the radius of the sphere
        System.out.print("Enter the radius of the
sphere: ");
```

```
double radius = sc.nextDouble();

// Create a Sphere object
Sphere sphere = new Sphere(radius);

        // Display the volume and surface area of the
sphere
        sphere.display();


        sc.close();
    }
}
```

2. Design a screen to handle the Mouse Events such as MOUSE_MOVED and

MOUSE_CLICK and display the position of the Mouse_Click in a TextField.

```
package ChapFive;


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

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

    public MouseEventHandler()
    {
        setTitle("Mouse Event Handler");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());

        textField = new JTextField();
        textField.setEditable(false);
        textField.setFont(new Font("Arial", Font.PLAIN, 24));
        add(textField, BorderLayout.SOUTH);

        JPanel panel = new JPanel()
        {
            @Override
            protected void paintComponent(Graphics g)
            {
                super.paintComponent(g);
                setBackground(Color.WHITE);
            }
```

```
};

panel.addMouseListener(new MouseAdapter()
{
    @Override
    public void mouseClicked(MouseEvent e)
    {
        int x = e.getX();
        int y = e.getY();
        textField.setText("Mouse Clicked at: ("+x+","+y+")");
    }
});

        panel.addMouseMotionListener(new MouseAdapter()
{
        @Override
        public void mouseMoved(MouseEvent e)
{
            int x = e.getX();
            int y = e.getY();
            setTitle("Mouse Moved to: ("+x+","+y+")");
    }
});
```

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

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

Slip No. 9

1. Define Clock class that does following 1. Accept hours , Minutes And Seconds 2. Check the Validity Of numbers. 3. Set the time AM/PM Mode. Use the neccessory construcots and methods

```
import java.util.Scanner;


class Clock {
    // Attributes of the Clock class
    private int hours;
    private int minutes;
    private int seconds;
    private String period;   // AM or PM
```

```
// Constructor to initialize the Clock object
public Clock(int hours, int minutes, int seconds, String period) {
    if (isValidTime(hours, minutes, seconds, period)) {
        this.hours = hours;
        this.minutes = minutes;
        this.seconds = seconds;
        this.period = period;
    } else {
        System.out.println("Invalid time provided. Setting to default 12:00:
00 AM.");
        this.hours = 12;
        this.minutes = 0;
        this.seconds = 0;
        this.period = "AM";
    }
}

// Method to validate the time
private boolean isValidTime(int hours, int minutes, int seconds, String
period) {
        if (hours < 1 || hours > 12) {
            return false;
        }
        if (minutes < 0 || minutes >= 60) {
```

```
        return false;
    }
    if (seconds < 0 || seconds >= 60) {
        return false;
    }
        if (!period.equalsIgnoreCase("AM")
&& !period.equalsIgnoreCase("PM")) {
            return false;
        }
        return true;
    }


    // Method to set the time
    public void setTime(int hours, int minutes, int seconds, String period) {
        if (isValidTime(hours, minutes, seconds, period)) {
            this.hours = hours;
            this.minutes = minutes;
            this.seconds = seconds;
            this.period = period;
        } else {
            System.out.println("Invalid time provided. Time not set.");
        }
    }
```

```
    // Method to display the time
    public void displayTime() {
        System.out.printf("Current Time: %02d:%02d:%02d %s\n", hours,
minutes, seconds, period.toUpperCase());
    }

// Main method to test the Clock class
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    // Prompt user to enter time
    System.out.print("Enter hours (1-12): ");
    int hours = sc.nextInt();
    System.out.print("Enter minutes (0-59): ");
    int minutes = sc.nextInt();
    System.out.print("Enter seconds (0-59): ");
    int seconds = sc.nextInt();
    System.out.print("Enter period (AM/PM): ");
    String period = sc.next();

    // Create a Clock object
    Clock clock = new Clock(hours, minutes, seconds, period);

    // Display the current time
```

```
clock.displayTime();

// Optionally, set a new time
System.out.println("\nSetting new time:");
System.out.print("Enter hours (1-12): ");
hours = sc.nextInt();
System.out.print("Enter minutes (0-59): ");
minutes = sc.nextInt();
System.out.print("Enter seconds (0-59): ");
seconds = sc.nextInt();
System.out.print("Enter period (AM/PM): ");
period = sc.next();

// Set the new time
clock.setTime(hours, minutes, seconds, period);

// Display the updated time
clock.displayTime();

sc.close();
    }
}
```

2. Write program to using marker interface create class Product (product_id,product_name,product_cost,product_quantity)

default and parameterised constructors. create object of class product and display contents of each objects and display object count.

```
// Marker interface

interface Trackable {}

// Product class implementing the marker interface

class Product implements Trackable {

    // Attributes of the Product class

    private int productId;

    private String productName;

    private double productCost;

    private int productQuantity;


    // Static variable to keep track of object count

    private static int objectCount = 0;


    // Default constructor

    public Product() {

        this.productId = 0;

        this.productName = "Unknown";

        this.productCost = 0.0;
```

```
        this.productQuantity = 0;
        objectCount++;  // Increment object count when a new
object is created
    }

    // Parameterized constructor
    public Product(int productId, String productName, double
productCost, int productQuantity) {
        this.productId = productId;
        this.productName = productName;
        this.productCost = productCost;
        this.productQuantity = productQuantity;
        objectCount++;  // Increment object count when a new
object is created
    }


    // Method to display the product details
    public void displayProductDetails() {
        System.out.println("Product ID: " + productId);
        System.out.println("Product Name: " + productName);
        System.out.println("Product Cost: $" + productCost);
        System.out.println("Product Quantity: " +
productQuantity);
```

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

    // Static method to get the object count
    public static int getObjectCount() {
        return objectCount;
    }

    // Main method to test the Product class
    public static void main(String[] args) {
        // Create Product objects using default and parameterized
constructors
        Product product1 = new Product();
        Product product2 = new Product(101, "Laptop", 1500.00,
5);
        Product product3 = new Product(102, "Smartphone",
800.00, 10);

        // Display the details of each product
        product1.displayProductDetails();
        product2.displayProductDetails();
        product3.displayProductDetails();
```

```
        // Display the total number of Product objects created
        System.out.println("Total number of Product objects created: " + Product.getObjectCount());
    }
}
```

Slip No.10

1. Write a program to fine cube of given number using functional interface..

```
// Functional interface with a single abstract method
@FunctionalInterface
interface CubeCalculator
{
    // Abstract method to calculate the cube
    int calculate(int x);
}

public class CubeOfNumber
{
    public static void main(String[] args)
{
        // Using lambda expression to define the calculate method
        CubeCalculator cube = (int x) -> x * x * x;
```

```
// Test the cube calculation with a given number
int number = 5;
int result = cube.calculate(number);

        // Display the result
        System.out.println("The cube of " + number + " is: " +
result);
    }
}
```

2. Write a program create 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 student. Accept student details like rollno, name, class and marks of 6 subject form user. (One package 3 classes)

```
// File: src/student/StudentInfo.java
package student;

public class StudentInfo {
    private int rollNo;
```

```
    private String name;
    private String studentClass;
    private double percentage;

    // Constructor to initialize StudentInfo object
    public StudentInfo(int rollNo, String name, String
studentClass, double percentage) {
        this.rollNo = rollNo;
        this.name = name;
        this.studentClass = studentClass;
        this.percentage = percentage;
    }

    // Method to display student information
    public void displayInfo() {
        System.out.println("Student Roll No: " + rollNo);
        System.out.println("Student Name: " + name);
        System.out.println("Class: " + studentClass);
        System.out.println("Percentage: " +
percentage + "%");
    }
}
```

```
// File: src/student/StudentPer.java
package student;
```

```
public class StudentPer {
    // Method to calculate the percentage of a
student
    public double calculatePercentage(int[] marks) {
        int totalMarks = 0;
        for (int mark : marks) {
            totalMarks += mark;
        }
        // Calculate the percentage
        return (totalMarks / (double) (marks.length *
100)) * 100;
    }
}


// File: Main.java
import student.StudentInfo;
import student.StudentPer;

import java.util.Scanner;

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

```
        // Accept student details from the user
        System.out.print("Enter Roll No: ");
        int rollNo = sc.nextInt();
        sc.nextLine();  // Consume newline left-over
        System.out.print("Enter Name: ");
        String name = sc.nextLine();

        System.out.print("Enter Class: ");
        String studentClass = sc.nextLine();

        int[] marks = new int[6];
        System.out.println("Enter marks of 6 subjects
(out of 100): ");
        for (int i = 0; i < 6; i++) {
            System.out.print("Subject " + (i + 1) + ": ");
            marks[i] = sc.nextInt();
        }

        // Create an object of StudentPer to calculate
the percentage
        StudentPer studentPer = new StudentPer();
        double percentage =
studentPer.calculatePercentage(marks);
```

```
        // Create an object of StudentInfo to store and
display the student information
        StudentInfo studentInfo = new
StudentInfo(rollNo, name, studentClass,
percentage);
            studentInfo.displayInfo();

            sc.close();
    }
}
```

Slip No 11

1. Define an interface operation which has method volume() . define constant PI having value 3.142 create class cylinder which implements this interface (members - radius , height) Create one object and calculate volume.

```
package PracSlipsSemIV;
```

```
interface VolumeCalculable {
    double PI = 3.142;  // Define constant PI
    double volume();  // Abstract method to calculate volume
}
class Cylinder implements VolumeCalculable {
    private double radius;
    private double height;
    // Constructor to initialize radius and height
    public Cylinder(double radius, double height) {
        this.radius = radius;
        this.height = height;
    }

    // Implement the volume() method
    @Override
```

```
    public double volume() {
        return PI * radius * radius * height;  // Volume
of a cylinder = πr²h
    }
}
public class Mainn {
    public static void main(String[] args) {
        // Create an object of Cylinder
        Cylinder cylinder = new Cylinder(5, 10);

        // Calculate and print the volume
        double volume = cylinder.volume();
        System.out.println("The volume of the
cylinder is: " + volume);
    }
}
```

2.write java program to accept username and password from user if username and password are

not same then raise "Invalid Password" with appropriate message

```
import java.util.Scanner;

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

public class UserAuthentication {

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

        try {
            // Accept username from the user
            System.out.print("Enter username: ");
            String username = scanner.nextLine();
```

```
            // Accept password from the user
            System.out.print("Enter password: ");
            String password = scanner.nextLine();

            // Check if username and password are the same
            if (!username.equals(password)) {
                // If not the same, raise
InvalidPasswordException
                throw new InvalidPasswordException("Invalid
Password: Username and Password do not match.");
            }

            // If they match, print a success message
            System.out.println("Login successful!");

        } catch (InvalidPasswordException e) {
            // Handle the exception and print the message
            System.out.println(e.getMessage());
        } finally {
            scanner.close();  // Close the scanner to prevent
resource leaks
```

```
        }
    }
}
```

Slip No 12

1.Write a program to create parent class College(cno,cname,caddr) and derived class Department (dno,dname) from College .Write neccessary method to diaplay collage details.

```
// Parent Class: College
class College {
    protected int cno;
    protected String cname;
    protected String caddr;

    // Constructor to initialize college details
    public College(int cno, String cname, String caddr) {
        this.cno = cno;
        this.cname = cname;
        this.caddr = caddr;
    }
```

```
    // Method to display college details
    public void displayCollegeDetails() {
        System.out.println("College Number: " + cno);
        System.out.println("College Name: " + cname);
        System.out.println("College Address: " + caddr);
    }
}

// Derived Class: Department
class Department extends College {
    private int dno;
    private String dname;

    // Constructor to initialize department details and call
super class constructor
    public Department(int cno, String cname, String
caddr, int dno, String dname) {
        super(cno, cname, caddr);  // Call the parent class
(College) constructor
        this.dno = dno;
        this.dname = dname;
```

```
    }

    // Method to display department details along with
college details
    public void displayDepartmentDetails() {
        // Display college details (inherited from the parent
class)
        displayCollegeDetails();
        // Display department details
        System.out.println("Department Number: " + dno);
        System.out.println("Department Name: " +
dname);
    }
}

// Main Class
public class Main {
    public static void main(String[] args) {
        // Create an object of Department
```

```
        Department dept = new Department(101, "ABC College", "123 College Street", 201, "Computer Science");

        // Display department and college details
        dept.displayDepartmentDetails();
    }
}
```

2. 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.

```
package ChapFive;

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

class BuildCalculator extends JFrame implements ActionListener
{
    JFrame actualWindow;
    JPanel resultPanel, buttonPanel, infoPanel;
```

```
    JTextField resultTxt;
    JButton btn_digits[] = new JButton[10];
    JButton btn_plus, btn_minus, btn_mul, btn_div, btn_equal, btn_dot,
btn_clear;

char eventFrom;

JLabel expression, appTitle, siteTitle;

double oparand_1 = 0, operand_2 = 0;
String operator = "=";

BuildCalculator()
{
    Font txtFont = new Font("SansSerif", Font.BOLD, 20);
    Font titleFont = new Font("SansSerif", Font.BOLD, 30);
    Font expressionFont = new Font("SansSerif", Font.BOLD, 15);

    actualWindow = new JFrame("Calculator");
    resultPanel = new JPanel();
    buttonPanel = new JPanel();
    infoPanel = new JPanel();
```

```
actualWindow.setLayout(new GridLayout(3, 1));
buttonPanel.setLayout(new GridLayout(4, 4));
infoPanel.setLayout(new GridLayout(3, 1));
actualWindow.setResizable(false);

 appTitle = new JLabel("Simple Calculator");
 appTitle.setFont(titleFont);
 expression = new JLabel("Expression shown here");
 expression.setFont(expressionFont);
siteTitle = new JLabel("www.btechsmartclass.com");
 siteTitle.setFont(expressionFont);
 siteTitle.setHorizontalAlignment(SwingConstants.CENTER);
 siteTitle.setForeground(Color.BLUE);

 resultTxt = new JTextField(15);
 resultTxt.setBorder(null);
 resultTxt.setPreferredSize(new Dimension(15, 50));
 resultTxt.setFont(txtFont);
 resultTxt.setHorizontalAlignment(SwingConstants.RIGHT);

for(int i = 0; i < 10; i++)
{
    btn_digits[i] = new JButton(""+i);
```

```
    btn_digits[i].addActionListener(this);
}
btn_plus=newJButton("+");
btn_plus.addActionListener(this);
btn_minus=newJButton("-");
btn_minus.addActionListener(this);
btn_mul=newJButton("*");
btn_mul.addActionListener(this);
btn_div=newJButton("/");
btn_div.addActionListener(this);
btn_dot=newJButton(".");
btn_dot.addActionListener(this);
btn_equal=newJButton("=");
btn_equal.addActionListener(this);
btn_clear=newJButton("Clear");
btn_clear.addActionListener(this);

resultPanel.add(appTitle);
resultPanel.add(resultTxt);
resultPanel.add(expression);
for(inti=0;i<10;i++)
{
    buttonPanel.add(btn_digits[i]);
}
```

```
        buttonPanel.add(btn_plus);
        buttonPanel.add(btn_minus);
        buttonPanel.add(btn_mul);
        buttonPanel.add(btn_div);
        buttonPanel.add(btn_dot);
        buttonPanel.add(btn_equal);
        infoPanel.add(btn_clear);
        infoPanel.add(siteTitle);

        actualWindow.add(resultPanel);
        actualWindow.add(buttonPanel);
        actualWindow.add(infoPanel);

        actualWindow.setSize(300, 500);
        actualWindow.setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent e)
    {
        eventFrom = e.getActionCommand().charAt(0);
        String buildNumber;
```

```
if(Character.isDigit(eventFrom))
{
    buildNumber=resultTxt.getText()+eventFrom;
    resultTxt.setText(buildNumber);
}
else
        if(e.getActionCommand()==".")
{
    buildNumber=resultTxt.getText()+eventFrom;
    resultTxt.setText(buildNumber);
}
elseif(eventFrom!='=')
{
    oparand_1=Double.parseDouble(resultTxt.getText());
    operator=e.getActionCommand();
    expression.setText(oparand_1+" "+operator);
    resultTxt.setText("");
}
else
        if(e.getActionCommand()=="Clear")
{
    resultTxt.setText("");
}
else
```

```
{
    operand_2 = Double.parseDouble(resultTxt.getText());

    expression.setText(expression.getText() + " " + operand_2);
    switch(operator)
    {
        case "+":    resultTxt.setText(""+(oparand_1+operand_2));
        break;
        case "-":    resultTxt.setText(""+(oparand_1-operand_2));
        break;
        case "*":    resultTxt.setText(""+(oparand_1*operand_2));
        break;
                case "/":resultTxt.setText(""+(oparand_1/operand_2));
                    try
                    {
                        if(operand_2==0)
                            throw new ArithmeticException();
                        resultTxt.setText(""+(oparand_1/operand_2));
break;
                     }
                    catch(ArithmeticException ae)
                    {

JOptionPane.showMessageDialog(actualWindow, "Divisor can not be
ZERO");
```

```
                    }
                }
            }
        }
    }
```

```
public class SimpleCalculator
{
    public static void main(String[] args)
{
        new BuildCalculator();
    }
}
```

Slip No.13

1.Write to program to accept file name from command prompt ,if the file exist then display number of words and files in that files.

```
import java.io.File;

import java.io.FileNotFoundException;
import java.util.Scanner;


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

```
        // Check if the user provided a filename as a
command line argument
        if (args.length != 1) {
            System.out.println("Please provide a filename as
a command line argument.");
            return;
        }

        // Get the filename from the command line
argument
        String filename = args[0];

        // Create a File object
        File file = new File(filename);

        // Check if the file exists
        if (!file.exists()) {
            System.out.println("The file " + filename + " does
not exist.");
            return;
        }
```

```
// Variables to count words and lines
int wordCount = 0;
int lineCount = 0;

try {
    // Create a Scanner to read the file
    Scanner scanner = new Scanner(file);

    // Read the file line by line
    while (scanner.hasNextLine()) {
        String line = scanner.nextLine();
        lineCount++;  // Increment line count
        wordCount += line.split("\\s+").length;  // Split
the line into words and increment word count
    }

    // Close the scanner
    scanner.close();
```

```
            // Display the results
            System.out.println("Number of lines: " +
lineCount);
            System.out.println("Number of words: " +
wordCount);
        } catch (FileNotFoundException e) {
            System.out.println("File not found: " + filename);
        }
    }
}
```

2) Write a java 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

```
import java.text.SimpleDateFormat;
import java.util.Date;
```

```
public class DateTimeFormats {
    public static void main(String[] args) {
        // Get the current date and time
        Date currentDate = new Date();

        // Format 1: Current date is : 31/08/2021
        SimpleDateFormat format1 = new
SimpleDateFormat("dd/MM/yyyy");
        System.out.println("Current date is : " +
format1.format(currentDate));


        // Format 2: Current date is : 08-31-2021
        SimpleDateFormat format2 = new
SimpleDateFormat("MM-dd-yyyy");
        System.out.println("Current date is : " +
format2.format(currentDate));


        // Format 3: Current date is : Tuesday August 31 2021
        SimpleDateFormat format3 = new
SimpleDateFormat("EEEE MMMM dd yyyy");
        System.out.println("Current date is : " +
format3.format(currentDate));
```

```
        // Format 4: Current date and time is : Fri August 31 15:25:59
IST 2021

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

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


        // Format 5: Current date and time is : 31/08/21 15:25:59 PM
+0530

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

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

}
```

Slip No.14

1.Write java program to accept number from user if number is zero then throw user defined exacption "Number is 0" Otherwise number is prime or not (Use static keyword) ..

```
import java.util.Scanner;

// Custom exception class
class ZeroNumberException extends Exception {
    public ZeroNumberException(String message) {
```

```
        super(message);
    }
}
public class PrimeChecker {

    // Static method to check if a number is prime
    public static boolean isPrime(int number) {
        if (number <= 1) {
            return false;  // 0 and 1 are not prime numbers
        }
        for (int i = 2; i <= Math.sqrt(number); i++) {
            if (number % i == 0) {
                return false;  // Not prime if divisible by any number
other than 1 and itself
            }
        }
        return true;
    }

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

```
try {
    // Accept number from user
    System.out.print("Enter a number: ");
    int number = scanner.nextInt();

    // Check if the number is zero and throw custom exception
    if (number == 0) {
        throw new ZeroNumberException("Number is 0");
    }

    // Check if the number is prime
    if (isPrime(number)) {
        System.out.println(number + " is a prime number.");
    } else {
        System.out.println(number + " is not a prime number.");
    }

} catch (ZeroNumberException e) {
    // Handle custom exception
```

```
            System.out.println(e.getMessage());
        } finally {
            scanner.close();  // Close the scanner to prevent
resource leaks
        }
    }
}
```

2. 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.

```
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;
    }
```

```
}


packageTY;

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

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

package First;

import SY.SYMarks;
import TY.TYMarks;

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

    public Student(int rollNumber, String name, SYMarks syMarks,
TYMarks tyMarks)
    {
```

```
        this.rollNumber = rollNumber;
        this.name = name;
        this.syMarks = syMarks;
        this.tyMarks = tyMarks;
    }

    private int calculateTotalMarks()
    {
        return syMarks.ComputerTotal + tyMarks.Theory;
    }

    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 void displayResult()
```

```
{
    int totalMarks = calculateTotalMarks();

    String grade = calculateGrade(totalMarks);

    System.out.println("Roll Number: " + rollNumber);
        System.out.println("Name: " + name);
        System.out.println("Total Marks (SY Computer + TY Theory): " +
totalMarks);

        System.out.println("Grade: " + grade);
    }

    public static void main(String[] args)
    {

        SYMarks syMarks1 = new SYMarks(75, 80, 65);
        TYMarks tyMarks1 = new TYMarks(70, 60);

        Student student1 = new Student(1, "Minal", syMarks1, tyMarks1);

        student1.displayResult();
    }
}
```

Slip No.15

1. write java prgorams Accept the names of two files and copy contents of first to second first file having bookname and author name in the file(Commandline)

```
import java.io.File;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Scanner;


public class FileCopyy {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);


        try {

// Accept the names of the two files from the user

System.out.print("Enter the name of the source file: ");

String sourceFileName = scanner.nextLine();
```

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

// Create File objects for the source and destination files
File sourceFile = new File(sourceFileName);
File destinationFile = new File(destinationFileName);

// Check if the source file exists
if (!sourceFile.exists()) {
    System.out.println("Source file does not exist.");
    return;
}

// FileReader and FileWriter for reading and writing the files
FileReader fileReader = new FileReader(sourceFile);
FileWriter fileWriter = new FileWriter(destinationFile);

// Variable to hold the content being read
int content;
```

```
            // Read the source file and write its content to the destination
file
            while ((content = fileReader.read()) != -1) {
                fileWriter.write(content);
            }

            // Close the FileReader and FileWriter
            fileReader.close();
            fileWriter.close();

            System.out.println("Content copied successfully from "+
sourceFileName+" to "+destinationFileName);
        } catch (IOException e) {
            System.out.println("An error occurred: " +
e.getMessage());
        } finally {
            scanner.close();  // Close the scanner to prevent resource
leaks
        }
    }
}
```

2.Write program to define class Account having member custname, accno. Define default and parameterised constructor. Create subclass for saving account with member savingbal, minbal. Create derived class Accountdetail that extends the class saving account with members, depositamt, withdrawlamt. Write appropriate method to display customer datails..

```
// Base Class: Account
class Account {
    protected String custname; // Customer name
    protected String accno;    // Account number

    // Default constructor
    public Account() {
        this.custname = "Unknown";
        this.accno = "0000";
    }

    // Parameterized constructor
    public Account(String custname, String accno) {
        this.custname = custname;
        this.accno = accno;
    }
```

```
    // Method to display account details
    public void displayAccountDetails() {
        System.out.println("Customer Name: " + custname);
        System.out.println("Account Number: " + accno);
    }
}

// Subclass: SavingAccount
class SavingAccount extends Account {
    protected double savingbal; // Savings balance
    protected double minbal;    // Minimum balance

    // Default constructor
    public SavingAccount() {
        super(); // Call the parent class default constructor
        this.savingbal = 0.0;
        this.minbal = 500.0; // Assuming minimum balance is 500
    }

    // Parameterized constructor
```

```
    public SavingAccount(String custname, String accno, double
savingbal, double minbal) {

        super(custname, accno); // Call the parent class
parameterized constructor

        this.savingbal = savingbal;
        this.minbal = minbal;
    }


    // Method to display savings account details
    public void displaySavingsDetails() {
        displayAccountDetails(); // Display base class details
        System.out.println("Savings Balance: " + savingbal);
        System.out.println("Minimum Balance: " + minbal);
    }
}


// Derived Class: AccountDetail
class AccountDetail extends SavingAccount {
    private double depositamt;    // Deposit amount
    private double withdrawlamt; // Withdrawal amount


    // Default constructor
```

```
public AccountDetail() {
    super(); // Call the SavingAccount default constructor
    this.depositamt = 0.0;
    this.withdrawlamt = 0.0;
    }


    // Parameterized constructor

    public AccountDetail(String custname, String accno, double
savingbal, double minbal, double depositamt, double
withdrawlamt) {

        super(custname, accno, savingbal, minbal); // Call the
SavingAccount parameterized constructor
        this.depositamt = depositamt;
        this.withdrawlamt = withdrawlamt;
    }


    // Method to display account details with deposit and withdrawal
amounts

    public void displayAccountDetail() {
        displaySavingsDetails(); // Display savings account details
        System.out.println("Deposit Amount: " + depositamt);
        System.out.println("Withdrawal Amount: " + withdrawlamt);
}
```

```
}


// Main Class to test the program

public class Main {

    public static void main(String[] args) {

        // Creating an object of AccountDetail using the
parameterized constructor

        AccountDetail accountDetail = new AccountDetail("John
Doe", "123456789", 10000.0, 500.0, 2000.0, 1500.0);



        // Displaying customer details

        accountDetail.displayAccountDetail();

    }

}
```

Slip No 16

1. Write a java program find square of given number using function interface

```
import java.util.function.Function;

import java.util.Scanner;


public class SquareUsingFunction {
```

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

    // Accept a number from the user
    System.out.print("Enter a number: ");
    int number = scanner.nextInt();

    // Define a Function interface to calculate the square of a number
    Function<Integer, Integer> squareFunction = n -> n * n;

    // Use the apply method of Function to get the square of the number
    int square = squareFunction.apply(number);

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

```
        scanner.close();  // Close the scanner to
prevent resource leaks
    }
}
```

Slip no 17

1. Design super class Customer (name, phonenumber). Derive class Depositor (accno, balance) from customer. Again derived class borrower (loanno, loanamt) from depositor. Write neccassary member function to read and diaplay details of n customer.

```
import java.util.Scanner;

// Superclass: Customer
class Customer {
    protected String name;          // Customer name
    protected String phonenumber;    // Customer
phone number

    // Default constructor
    public Customer() {
```

```
        this.name = "";
        this.phonenumber = "";
    }

    // Parameterized constructor
    public Customer(String name, String
phonenumber) {
        this.name = name;
        this.phonenumber = phonenumber;
    }

    // Method to read customer details
    public void readCustomerDetails(Scanner
scanner) {
        System.out.print("Enter customer name: ");
        this.name = scanner.nextLine();
        System.out.print("Enter customer phone
number: ");
        this.phonenumber = scanner.nextLine();
    }

    // Method to display customer details
    public void displayCustomerDetails() {
        System.out.println("Customer Name: " +
name);
```

```
        System.out.println("Phone Number: " +
phonenumber);
    }
}

// Subclass: Depositor
class Depositor extends Customer {
    protected String accno;     // Account number
    protected double balance;   // Balance

    // Default constructor
    public Depositor() {
        super();  // Call the superclass constructor
        this.accno = "";
        this.balance = 0.0;
    }

    // Parameterized constructor
    public Depositor(String name, String
phonenumber, String accno, double balance) {
        super(name, phonenumber);  // Call the
superclass constructor
        this.accno = accno;
        this.balance = balance;
    }
```

```
    // Method to read depositor details
    public void readDepositorDetails(Scanner
scanner) {
        readCustomerDetails(scanner);  // Read
customer details
        System.out.print("Enter account number: ");
        this.accno = scanner.nextLine();
        System.out.print("Enter balance: ");
        this.balance = scanner.nextDouble();
        scanner.nextLine();  // Consume the newline
character
    }

    // Method to display depositor details
    public void displayDepositorDetails() {
        displayCustomerDetails();  // Display
customer details
        System.out.println("Account Number: " +
accno);
        System.out.println("Balance: " + balance);
    }
}

// Derived class: Borrower
class Borrower extends Depositor {
    private String loanno;     // Loan number
```

```
    private double loanamt;    // Loan amount

    // Default constructor
    public Borrower() {
        super();  // Call the Depositor default
constructor
            this.loanno = "";
            this.loanamt = 0.0;
        }

        // Parameterized constructor
        public Borrower(String name, String
phonenumber, String accno, double balance,
String loanno, double loanamt) {
            super(name, phonenumber, accno, balance);
// Call the Depositor parameterized constructor
            this.loanno = loanno;
            this.loanamt = loanamt;
        }

        // Method to read borrower details
        public void readBorrowerDetails(Scanner
scanner) {
            readDepositorDetails(scanner);  // Read
depositor details
            System.out.print("Enter loan number: ");
```

```
        this.loanno = scanner.nextLine();
        System.out.print("Enter loan amount: ");
        this.loanamt = scanner.nextDouble();
        scanner.nextLine();  // Consume the newline
character
    }

    // Method to display borrower details
    public void displayBorrowerDetails() {
        displayDepositorDetails();  // Display
depositor details
        System.out.println("Loan Number: " + loanno);
        System.out.println("Loan Amount: " +
loanamt);
    }
}

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

        System.out.print("Enter the number of
customers: ");
        int n = scanner.nextInt();
```

```
        scanner.nextLine();  // Consume the newline
character

        // Create an array of Borrower objects
        Borrower[] borrowers = new Borrower[n];

        // Read details of each customer
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for
customer " + (i + 1) + ":");
            borrowers[i] = new Borrower();  // Create a
new Borrower object
            borrowers[i].readBorrowerDetails(scanner);  //
Read borrower details
        }

        // Display details of each customer
        System.out.println("\nCustomer Details:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nDetails of customer "
+ (i + 1) + ":");
            borrowers[i].displayBorrowerDetails();  //
Display borrower details
        }
```

```
        scanner.close();  // Close the scanner to
prevent resource leaks
    }
}
```

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

```
package PracSlipsSemIV;

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

public class StringManipulator extends JFrame {
    private JTextField textField1, textField2,
textField3;
    private JButton concatButton, reverseButton;

    public StringManipulator() {
        // Set up the JFrame
        setTitle("String Manipulator");
        setSize(400, 200);
```

```
setDefaultCloseOperation(JFrame.EXIT_ON_CLO
SE);
        setLayout(null);

        // Create and position the text fields
        textField1 = new JTextField();
        textField1.setBounds(50, 30, 300, 25);
        add(textField1);

        textField2 = new JTextField();
        textField2.setBounds(50, 60, 300, 25);
        add(textField2);

        textField3 = new JTextField();
        textField3.setBounds(50, 120, 300, 25);
        textField3.setEditable(false);
        add(textField3);

        // Create and position the buttons
        concatButton = new JButton("Concatenate");
        concatButton.setBounds(50, 90, 140, 25);
        add(concatButton);

        reverseButton = new JButton("Reverse");
        reverseButton.setBounds(210, 90, 140, 25);
```

```
        add(reverseButton);

        // Add action listener for the concatenate
button
        concatButton.addActionListener(new
ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e)
{
                String str1 = textField1.getText();
                String str2 = textField2.getText();
                String result = str1 + str2;
                textField3.setText(result);
            }
        });

        // Add action listener for the reverse button
        reverseButton.addActionListener(new
ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e)
{
                String str1 = textField1.getText();
                String result = new
StringBuilder(str1).reverse().toString();
                textField3.setText(result);
```

```
        }
    });
}

public static void main(String[] args) {
    // Run the program
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new StringManipulator().setVisible(true);
        }
    });
}
```

Slip No.18

1. Write java program to implement border layout manager

```
package PracSlipsSemIV;

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

public class BorderLayoutDemo extends JFrame {

    public BorderLayoutDemo() {
```

```
        // Set the title of the JFrame
        setTitle("BorderLayout Manager Example");

        // Set the size of the JFrame
        setSize(400, 300);

        // Set the default close operation
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Set the layout to BorderLayout
        setLayout(new BorderLayout());

        // Create buttons for each region
        JButton northButton = new JButton("North");
        JButton southButton = new JButton("South");
        JButton eastButton = new JButton("East");
        JButton westButton = new JButton("West");
        JButton centerButton = new JButton("Center");

        // Add buttons to the BorderLayout regions
        add(northButton, BorderLayout.NORTH);
        add(southButton, BorderLayout.SOUTH);
        add(eastButton, BorderLayout.EAST);
```

```
        add(westButton, BorderLayout.WEST);
        add(centerButton, BorderLayout.CENTER);
    }

    public static void main(String[] args) {
        // Run the program
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                new
BorderLayoutDemo().setVisible(true);
            }
        });
    }
}
```

2. 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.

```
package ChaTwo;

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

class CricketPlayer
```

```
{
    private String name;
    private int no_of_innings;
    private int no_of_times_notout;
    private int total_runs;
    private double bat_avg;

    public CricketPlayer(String name, int no_of_innings, int
no_of_times_notout, int total_runs)
    {
        this.name = name;
        this.no_of_innings = no_of_innings;

        this.no_of_times_notout = no_of_times_notout;

        this.total_runs = total_runs;

        this.bat_avg = avg();
}


private double avg() {
    int outs = no_of_innings - no_of_times_notout;
    if (outs > 0)
    {
        return (double) total_runs / outs;
    }
    else
    {
        return total_runs;
    }
}

public double getBatAvg()
```

```
{
    return bat_avg;
}


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


    public void display()
    {
        System.out.println("Name:"+name);

        System.out.println("Innings:"+no_of_innings);

        System.out.println("Not Out:"+no_of_times_notout);

        System.out.println("Total Runs:"+total_runs);

        System.out.println("Batting Average:"+bat_avg);

        System.out.println();
    }

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

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

int n = scanner.nextInt();
```

```
scanner.nextLine();

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.nextLine();

System.out.print("Number of Innings: ");

int no_of_innings = scanner.nextInt();

System.out.print("Number of Times Not Out: ");

int no_of_times_notout = scanner.nextInt();

System.out.print("Total Runs: ");

int total_runs = scanner.nextInt();

scanner.nextLine();


players[i] = new CricketPlayer(name, no_of_innings,
no_of_times_notout, total_runs);
        }


        CricketPlayer.sortPlayers(players);
```

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

        scanner.close();
    }
}
```

Slip No 19

1. Write java program to accept two diamensional array from user and display sum of its diagonal elements

```
import java.util.Scanner;


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


        // Accepting the size of the matrix (assuming it's a square
matrix)
        System.out.print("Enter the size of the matrix (n x n): ");
        int n = scanner.nextInt();
```

```
int[][] matrix = new int[n][n];

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

// Calculating the sum of diagonal elements
int sum = 0;
for (int i = 0; i < n; i++) {
    sum += matrix[i][i]; // Primary diagonal
    sum += matrix[i][n - 1 - i]; // Secondary diagonal
}

// If the matrix has an odd size, the middle element is added
twice in the above loop.
        // So we subtract it once.
        if (n % 2 != 0) {
            sum -= matrix[n / 2][n / 2];
```

```
}

// Displaying the sum of the diagonal elements
System.out.println("Sum of diagonal elements: " + sum);

        scanner.close();
    }
}
```

3. **Write a program which shows the combo box which includes TYBsc(Comp. Sci) subjects. Display the selected subject in text field.**

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

public class SubjectSelector {
    public static void main(String[] args) {
        // Create a new JFrame container
        JFrame frame = new JFrame("TYBSc(Comp. Sci) Subject Selector");

        // Specify FlowLayout for the layout manager
        frame.setLayout(new FlowLayout());

        // Give the frame an initial size
```

```
        frame.setSize(400, 150);

        // Terminate the program when the user closes the
application

frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLO
SE);

        // Create an array of subjects for the combo box
        String[] subjects = {
            "Data Structures",
            "Operating Systems",
            "Database Management Systems",
            "Computer Networks",
            "Software Engineering",
            "Theory of Computation",
            "Web Technologies",
            "Artificial Intelligence"
        };

        // Create a combo box and add the subjects to it
        JComboBox<String> comboBox = new
JComboBox<>(subjects);

        // Create a text field to display the selected subject
        JTextField textField = new JTextField(25);
        textField.setEditable(false);

        // Add an action listener to the combo box
```

```
        comboBox.addActionListener(new ActionListener()
{
            public void actionPerformed(ActionEvent e) {
                // Get the selected item and set it in the text field
                String selectedSubject = (String)
comboBox.getSelectedItem();
                textField.setText(selectedSubject);
            }
});

// Add the combo box and text field to the frame
frame.add(new JLabel("Select a Subject:"));
frame.add(comboBox);
frame.add(new JLabel("Selected Subject:"));
frame.add(textField);

            // Display the frame
            frame.setVisible(true);
        }
}
```

Slip No.20

1.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.

```
package Inheritance;

class Continent

{
```

```
String continentName;

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

public String getContinentName()
{
    return continentName;
}
}

class Country extends Continent
{
String countryName;

public Country(String continentName, String countryName)
{
    super(continentName);
    this.countryName = countryName;
}

public String getCountryName()
```

```
{
    return 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 String getStateName()
{
    return stateName;
}
public String getPlaceName()
{
    return placeName;
```

```
}
public void display()
{
    System.out.println("Place: " + getPlaceName());

    System.out.println("State: " + getStateName());


    System.out.println("Country: " + getCountryName());


    System.out.println("Continent: " + getContinentName());
 }
}

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

2. Write package for operation which has two classes , Addition and Maximum . Addition has two methods add() and substract() which

are used two add two integers and substract two float values respectively. Maximum has method max() to display maximum of two integers.

```
// File: operation/Addition.java

package operation;

public class Addition {


    // Method to add two integers
    public int add(int a, int b) {
        return a + b;
    }

    // Method to subtract two float values
    public float subtract(float a, float b) {
        return a - b;
    }
}
```

```
// File: operation/Maximum.java

package operation;
```

```
public class Maximum {

    // Method to find the maximum of two integers
    public int max(int a, int b) {
        return (a > b) ? a : b;
    }
}


// File: TestOperation.java
import operation.Addition;
import operation.Maximum;

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

        // Testing the add method
        int sum = addition.add(10, 20);
        System.out.println("Sum of 10 and 20: " + sum);
```

```
        // Testing the subtract method
        float difference = addition.subtract(5.5f, 3.2f);
        System.out.println("Difference between 5.5 and 3.2: " +
difference);

        // Testing the max method
        int max = maximum.max(10, 20);
        System.out.println("Maximum between 10 and 20: " +
max);
    }
}
```

Slip No 21

1.Define class MyDate (Day, Month, Year) with methods to accept and display MyDateobject . Accept Date ad dd,mm,yyyy. throw user defined exception "InvalidDateExaception" if date is invalid

```
import java.util.Scanner;

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

```
// Method to accept date
public void acceptDate() throws InvalidDateException {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter day (dd): ");
    int day = scanner.nextInt();

    System.out.print("Enter month (mm): ");
    int month = scanner.nextInt();

    System.out.print("Enter year (yyyy): ");
        int year = scanner.nextInt();


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

// If valid, assign to fields
this.day = day;
this.month = month;
```

```
        this.year = year;
    }

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

    // Method to check if the date is valid
    private boolean isValidDate(int day, int month, int year) {
        if (month < 1 || month > 12) {
            return false;
        }
        if (day < 1 || day > 31) {
            return false;
        }

            // Check for months with 30 days
            if ((month == 4 || month == 6 || month == 9 || month == 11) &&
day > 30) {
                return false;
```

```
}

// Check for February and leap year
if (month == 2) {
    if (isLeapYear(year)) {
        if (day > 29) {
            return false;
        }
    } else {
        if (day > 28) {
            return false;
        }
    }
}

// Year check
if (year < 1) {
    return false;
}

return true;
}
```

```
// Method to check if a year is a leap year
private boolean isLeapYear(int year) {
    if (year % 4 == 0) {
        if (year % 100 == 0) {
            return year % 400 == 0;
        } else {
            return true;
        }
    }
    return false;
    }
}
public class InvalidDateException extends Exception {
    public InvalidDateException(String message) {
        super(message);
    }
}

public class TestMyDate {
    public static void main(String[] args) {
        MyDate date = new MyDate();
```

```
try {
    date.acceptDate();
    date.displayDate();
} catch (InvalidDateException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

2. Create an employee class (id name deptname,salary) . Define default and parameterized constructor . Use 'this' Keyword to intialize instance vaariable. Keep count of objects created. create object using parameterised constructor and dispaly the object count after each object is created (Use static member and method). Also display contents of each objects.

```
public class Employee {
    private int id;
    private String name;
    private String deptName;
    private double salary;

    // Static variable to keep track of the number of
Employee objects created
    private static int objectCount = 0;

    // Default constructor
    public Employee() {
```

```
        // Increment object count for every object created
        objectCount++;
    }

    // Parameterized constructor
    public Employee(int id, String name, String deptName,
double salary) {
        // Use 'this' keyword to refer to instance variables
        this.id = id;
        this.name = name;
        this.deptName = deptName;
        this.salary = salary;

        // Increment object count for every object created
        objectCount++;
    }

    // Static method to get the count of objects created
    public static int getObjectCount() {
        return objectCount;
    }

    // Method to display employee details
    public void display() {
        System.out.println("Employee ID: " + id);
        System.out.println("Employee Name: " + name);
        System.out.println("Department Name: " +
deptName);
        System.out.println("Salary: " + salary);
```

```
        System.out.println();
    }
}
public class TestEmployee {
    public static void main(String[] args) {
        // Create the first Employee object
        Employee emp1 = new Employee(101, "John Doe",
"IT", 50000.00);
        System.out.println("Employee 1 created.");
        emp1.display();
        System.out.println("Total number of Employee
objects: " + Employee.getObjectCount());
        System.out.println("-------------------------
-");

        // Create the second Employee object
        Employee emp2 = new Employee(102, "Jane Smith",
"HR", 55000.00);
        System.out.println("Employee 2 created.");
        emp2.display();
        System.out.println("Total number of Employee
objects: " + Employee.getObjectCount());
        System.out.println("-------------------------
-");

        // Create the third Employee object
        Employee emp3 = new Employee(103, "David
Johnson", "Finance", 60000.00);
        System.out.println("Employee 3 created.");
```

```
        emp3.display();
        System.out.println("Total number of Employee
objects: " + Employee.getObjectCount());
    }
}
```

Slip No. 22

1. Write program to create an abstract class named shape that constains two integers and an empty method named printArea() provide three classes Rectangle, Traingle and Circle such that each one of that classes extends the class Shape. Each one of the classes contain the method printArea() that prints the area of given shape (Use method Overriding)

```
package PracSlipsSemIV;


abstract class Shape {
    int dim1;
    int dim2;

    // Abstract method for calculating and printing the
area
    abstract void printArea();
}
```

```
class Rectangle extends Shape {

    // Constructor to initialize the dimensions of the
rectangle
    Rectangle(int length, int breadth) {
        this.dim1 = length;
        this.dim2 = breadth;
    }

    // Overriding the printArea method to calculate and
print the area of the rectangle
    @Override
    void printArea() {
        int area = dim1 * dim2;
        System.out.println("Rectangle Area: " + area);
    }
}

class Triangle extends Shape {

    // Constructor to initialize the base and height of the
triangle
```

```
Triangle(int base, int height) {
    this.dim1 = base;
    this.dim2 = height;
}

    // Overriding the printArea method to calculate and
print the area of the triangle
    @Override
    void printArea() {
        double area = 0.5 * dim1 * dim2;
        System.out.println("Triangle Area: " + area);
    }
}
class Circle extends Shape {
    int radius;

    // Constructor to initialize the radius of the circle
    Circle(int radius) {
        this.radius = radius;
    }
```

```
    // Overriding the printArea method to calculate and
print the area of the circle
    @Override
    void printArea() {
        double area = Math.PI * radius * radius;
        System.out.println("Circle Area: " + area);
    }
}
public class TestShape {
    public static void main(String[] args) {
        // Create a Rectangle object and print its area
        Shape rectangle = new Rectangle(10, 20);
        rectangle.printArea();

        // Create a Triangle object and print its area
        Shape triangle = new Triangle(10, 15);
        triangle.printArea();

        // Create a Circle object and print its area
```

```
        Shape circle = new Circle(7);
        circle.printArea();
    }
}
```

2. Write program that handles all mouse events and show the event name at the center of window , red in color when mouse event in fired(Use adapter classes)

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

public class MouseEventDemo extends JFrame {
    private String eventName = "";  // To store the name of the mouse event

    public MouseEventDemo() {
        // Set the title and size of the frame
        setTitle("Mouse Event Demo");
        setSize(400, 400);

setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
// Add mouse listeners using adapter classes
addMouseListener(new MouseAdapter() {
    public void mouseClicked(MouseEvent e) {
        eventName = "Mouse Clicked";
        repaint();  // Repaint the frame to update the
event name
    }

    public void mousePressed(MouseEvent e) {
        eventName = "Mouse Pressed";
        repaint();
    }

    public void mouseReleased(MouseEvent e) {
        eventName = "Mouse Released";
        repaint();
    }

    public void mouseEntered(MouseEvent e) {
```

```
        eventName = "Mouse Entered";
        repaint();
    }
    public void mouseExited(MouseEvent e) {
        eventName = "Mouse Exited";
        repaint();
    }
});
    addMouseMotionListener(new
MouseMotionAdapter() {
        public void mouseMoved(MouseEvent e) {
            eventName = "Mouse Moved";
            repaint();
        }

        public void mouseDragged(MouseEvent e) {
            eventName = "Mouse Dragged";
            repaint();
```

```
        }
    });
}

    // Override the paint method to display the event name
at the center
    public void paint(Graphics g) {
        super.paint(g);
        g.setColor(Color.RED);  // Set the text color to red
        g.setFont(new Font("Arial", Font.BOLD, 24));
        // Get the width and height of the frame
        int width = getWidth();
        int height = getHeight();
        // Calculate the position to center the text
        FontMetrics fm = g.getFontMetrics();
        int x = (width - fm.stringWidth(eventName)) / 2;
        int y = (height - fm.getHeight()) / 2 +
fm.getAscent();
        g.drawString(eventName, x, y);
    }
```

```
    public static void main(String[] args) {
        // Create an instance of the MouseEventDemo
class
        MouseEventDemo frame = new
MouseEventDemo();
        // Make the frame visible
        frame.setVisible(true);
    }
}
```

Slip No 23

1.Define class MyNumber having one private int data member. Write default constructor to initialize it to 0 and another constructor to initialize it to value. (use this) .Write methods is Negative, isPositive, isZero, isOdd, isEven. create an object in main use command line arguments to pass value to objects. (command line through pass value).

```
class MyNumber {
private int number;

// Default constructor to initialize number to 0
public MyNumber() {
```

```
        this.number = 0;
    }

    // Constructor to initialize number to a specific value
    public MyNumber(int number) {
        this.number = number;
    }

    // Method to check if the number is negative
    public boolean isNegative() {
        return this.number < 0;
    }

    // Method to check if the number is positive
    public boolean isPositive() {
        return this.number > 0;
    }

    // Method to check if the number is zero
    public boolean isZero() {
```

```
        return this.number == 0;
    }

    // Method to check if the number is odd
    public boolean isOdd() {
        return this.number % 2 != 0;
    }

    // Method to check if the number is even
    public boolean isEven() {
        return this.number % 2 == 0;
    }

    public static void main(String[] args) {
        if (args.length > 0) {
            try {
                int value = Integer.parseInt(args[0]);
                MyNumber myNum = new MyNumber(value);

                System.out.println("Number: " + value);
```

```
                System.out.println("Is Negative? " +
myNum.isNegative());

                System.out.println("Is Positive? " +
myNum.isPositive());

                System.out.println("Is Zero? " +
myNum.isZero());

                    System.out.println("Is Odd? " +
myNum.isOdd());

                    System.out.println("Is Even? " +
myNum.isEven());

                } catch (NumberFormatException e) {
                    System.out.println("Please enter a valid
integer.");
                }
            } else {
                System.out.println("Please pass a value as a
command-line argument.");
            }
        }
}
```

3. Write simple currency convertor as shown in figure. User can enter the amount of "Soingapure

dollars" ,"US Dollaros,"Euros" in floating point number. The converted values shall be display to 2 decimal places. Assume that 1USD= 1.41SGD , 1USD=0.92 Euro,1SGD=0.65 Euro

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

public class CurrencyConverter extends JFrame
implements ActionListener {
    private JTextField amountField;
    private JComboBox<String> currencyBox;
    private JLabel resultLabel;

    public CurrencyConverter() {
        setTitle("Currency Converter");
        setSize(350, 200);

setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(null);

        JLabel amountLabel = new JLabel("Enter Amount:");
        amountLabel.setBounds(30, 30, 100, 25);
        add(amountLabel);
```

```
amountField = new JTextField();
amountField.setBounds(140, 30, 150, 25);
add(amountField);

        JLabel currencyLabel = new
JLabel("Currency:");
        currencyLabel.setBounds(30, 70, 100, 25);
        add(currencyLabel);

        String[] currencies = {"Singapore Dollars
(SGD)", "US Dollars (USD)", "Euros (EUR)"};
        currencyBox = new
JComboBox<>(currencies);
        currencyBox.setBounds(140, 70, 150, 25);
        add(currencyBox);

        JButton convertButton = new
JButton("Convert");
        convertButton.setBounds(140, 110, 100, 25);
        convertButton.addActionListener(this);
        add(convertButton);

resultLabel = new JLabel("");
resultLabel.setBounds(30, 150, 300, 25);
add(resultLabel);
```

```
        setVisible(true);
    }

    @Override
        public void actionPerformed(ActionEvent e) {
            try {
                double amount =
Double.parseDouble(amountField.getText());
                String currency = (String)
currencyBox.getSelectedItem();
                double usd, sgd, eur;

                DecimalFormat df = new DecimalFormat("#.
00");

                if (currency.equals("Singapore Dollars
(SGD)")) {
                    usd = amount / 1.41;
                    eur = amount * 0.65;
                    resultLabel.setText("USD: " +
df.format(usd) + " | EUR: " + df.format(eur));
                } else if (currency.equals("US Dollars
(USD)")) {
                    sgd = amount * 1.41;
                    eur = amount * 0.92;
```

```
                resultLabel.setText("SGD: " +
df.format(sgd) + " | EUR: " + df.format(eur));
            } else if (currency.equals("Euros (EUR)")) {
                usd = amount / 0.92;
                sgd = amount / 0.65;
                resultLabel.setText("USD: " +
df.format(usd) + " | SGD: " + df.format(sgd));
            }
        } catch (NumberFormatException ex) {
            resultLabel.setText("Please enter a valid
number.");
        }
    }

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

Slip No.24

1. Create abstract class "Bank" with an abstract method "getBalance" Rs 100, Rs150, Rs200 are deposited into banks A, B, C. BankA, BankB, BankC are the subclasses of Bank class, each having mathod name "getBalance" , call this method by creating an object of each of three classes.

```
abstract class Bank {
```

```
    // Abstract method to be implemented by subclasses
    abstract int getBalance();
}

class BankA extends Bank {
    private int balance;

    // Constructor to set balance for BankA
    public BankA() {
        this.balance = 100;
    }

    // Implementation of the abstract method
    @Override
    int getBalance() {
        return this.balance;
    }
}

class BankB extends Bank {
    private int balance;

    // Constructor to set balance for BankB
    public BankB() {
        this.balance = 150;
    }

    // Implementation of the abstract method
    @Override
    int getBalance() {
        return this.balance;
    }
}
```

```
class BankC extends Bank {
    private int balance;

    // Constructor to set balance for BankC
    public BankC() {
        this.balance = 200;
    }

    // Implementation of the abstract method
    @Override
    int getBalance() {
        return this.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());
    }
}
```

2. Program that display three concentric circles where the user clicks mouse on a frame. The program must exist when user clicks 'X' on the frame.

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

public class ConcentricCircles extends JFrame {
    private int x = -1;
    private int y = -1;

    public ConcentricCircles() {
        setTitle("Concentric Circles");
        setSize(500, 500);

setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);

        // Add a mouse listener to capture the mouse click
event
        addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                x = e.getX();
                y = e.getY();
                repaint();
            }
        });
    }

    @Override
```

```
    public void paint(Graphics g) {
        super.paint(g);

        if (x != -1 && y != -1) {
            // Draw three concentric circles with a radius of 30,
60, and 90
            g.drawOval(x - 30, y - 30, 60, 60);
            g.drawOval(x - 60, y - 60, 120, 120);
            g.drawOval(x - 90, y - 90, 180, 180);
        }
    }

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

Slip No.25

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

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

class Student {
    private int rollno;
    private String name;
    private String className;
```

```
    private double percentage;

    // Constructor to initialize Student object
    public Student(int rollno, String name, String className,
double percentage) {
        this.rollno = rollno;
        this.name = name;
        this.className = className;
        this.percentage = percentage;
    }

    // Method to display student information
    public void displayInfo() {
        System.out.println("Student Information:");
        System.out.println("Roll No: " + rollno);
        System.out.println("Name: " + name);
        System.out.println("Class: " + className);
        System.out.println("Percentage: " + percentage +
"%");
    }
}

public class Main {
    public static void main(String[] args) {
        BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));

        try {
            // Reading student information from console
```

```
            System.out.print("Enter Roll No: ");
            int rollno = Integer.parseInt(reader.readLine());

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

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

            System.out.print("Enter Percentage: ");
            double percentage =
    Double.parseDouble(reader.readLine());

            // Creating Student object
            Student student = new Student(rollno, name,
    className, percentage);

                // Displaying student information
                student.displayInfo();

            } catch (IOException e) {
                System.out.println("An error occurred while
    reading input.");
            } catch (NumberFormatException e) {
                    System.out.println("Please enter valid numeric
    values for Roll No and Percentage.");
                }
        }
    }
```

2.Create the following GUI screen using appropriate layout managers. Accept the name, class , hobbies of the user and apply the changes and display the selected options in a text box.

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

class Swing2 extends JFrame implements ActionListener
{
    JLabel l1,l2,l3;
        JButton b;
        JRadioButton r1,r2,r3;
        JCheckBox c1,c2,c3;
        JTextField t1,t2;
        ButtonGroup b1;
        JPanel p1,p2;
    static int cnt;
        private StringBuffer s1=new StringBuffer();

        Swing2()
        {

            b1=new ButtonGroup();
            p1=new JPanel();
            p2=new JPanel();
            b=new JButton("Clear");
```

```
b.addActionListener(this);

r1=new JRadioButton("FY");
r2=new JRadioButton("SY");
r3=new JRadioButton("TY");

b1.add(r1);
b1.add(r2);
b1.add(r3);
r1.addActionListener(this);
r2.addActionListener(this);
r3.addActionListener(this);

c1=new JCheckBox("Music");
c2=new JCheckBox("Dance");
c3=new JCheckBox("Sports");

c1.addActionListener(this);
c2.addActionListener(this);
c3.addActionListener(this);

l1=new JLabel("Your Name");
l2=new JLabel("Your Class");
l3=new JLabel("Your Hobbies");
t1=new JTextField(20);
t2=new JTextField(30);

p1.setLayout(new GridLayout(5,2));
p1.add(l1);p1.add(t1);
```

```
p1.add(l2);p1.add(l3);
p1.add(r1);p1.add(c1);
p1.add(r2);p1.add(c2);
p1.add(r3);p1.add(c3);

p2.setLayout(new FlowLayout());
    p2.add(b);
    p2.add(t2);

    setLayout(new BorderLayout());
    add(p1,BorderLayout.NORTH);
    add(p2,BorderLayout.EAST);

setSize(400,200);
    setVisible(true);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public void actionPerformed(ActionEvent e)
    {

        if(e.getSource()==r1)
        {
cnt++;
if(cnt==1)
{
    String s=t1.getText();
        s1.append("Name=");
```

```
            s1.append(s);
        }
        s1.append("Class=FY");
    }
    else if(e.getSource()==r2)
    {
        cnt++;
        if(cnt==1)
        {
            String s=t1.getText();
                s1.append("Name=");
                s1.append(s);
        }
        s1.append("Class=SY");
    }
    else if(e.getSource()==r3)
    {
        cnt++;
        if(cnt==1)
        {
            String s=t1.getText();
                s1.append("Name=");
                s1.append(s);
        }
        s1.append("Class=TY");
    }


    else if(e.getSource()==c1)
```

```
{
    s1.append("Hobbies=Music");
}
else if(e.getSource()==c2)
{
    s1.append("Hobbies=Dance");
}
else if(e.getSource()==c3)
{
    s1.append("Hobbies=Sports");
}

t2.setText(new String(s1));

        if(e.getSource()==b)
        {
            t2.setText("");
                t1.setText("");
        }

}

public static void main(String arg[])
        {
            Swing2 s=new Swing2();

        }
}
```

Slip No.26

1.Define a Item Class (item_number,item_name,item_price) . Define Default and parameterised constructor. Keep count of object created. create object using parameterised constructor and display object count after each object is created (use static member and method)Also display contents of each objects..

```
class Item {
    private int item_number;
    private String item_name;
    private double item_price;

    // Static variable to keep count of objects created
    private static int objectCount = 0;

    // Default constructor
    public Item() {
        this.item_number = 0;
        this.item_name = "Unknown";
        this.item_price = 0.0;
        objectCount++;
    }

    // Parameterized constructor
    public Item(int item_number, String item_name, double
item_price) {
        this.item_number = item_number;
        this.item_name = item_name;
```

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

    // Static method to get the object count
    public static int getObjectCount() {
        return objectCount;
    }

    // Method to display item details
    public void displayItemDetails() {
        System.out.println("Item Number: " + item_number);
        System.out.println("Item Name: " + item_name);
        System.out.println("Item Price: $" + item_price);
        System.out.println("-------------------------
");
    }
}

public class Main {
    public static void main(String[] args) {
        // Creating first item object using parameterized
constructor
        Item item1 = new Item(101, "Laptop", 999.99);
        System.out.println("Item 1 created.");
        item1.displayItemDetails();
        System.out.println("Total Objects Created: " +
Item.getObjectCount());
```

```
        // Creating second item object using parameterized
constructor
        Item item2 = new Item(102, "Smartphone", 599.99);
        System.out.println("Item 2 created.");
        item2.displayItemDetails();
        System.out.println("Total Objects Created: " +
Item.getObjectCount());

        // Creating third item object using parameterized
constructor
        Item item3 = new Item(103, "Tablet", 299.99);
        System.out.println("Item 3 created.");
        item3.displayItemDetails();
        System.out.println("Total Objects Created: " +
Item.getObjectCount());
    }
}
```

3. Define class "Donor" to store below mentioned details of a blood donorname,age,address,contactnumber,bloodgroup,dateoflastdonation. Crate n object of this class for all regular donars at pune. Write this object to a file. Read these objects from file and display only those donors whose blood group is A+ve and had not donated for recent six months.

```
import java.io.*;
import java.time.LocalDate;
import java.time.Period;
import java.util.ArrayList;

class Donor implements Serializable {
```

```
    private String name;
    private int age;
    private String address;
    private String contactNumber;
    private String bloodGroup;
    private LocalDate dateOfLastDonation;

    // Constructor to initialize Donor object
    public Donor(String name, int age, String address, String
contactNumber, String bloodGroup, LocalDate
dateOfLastDonation) {
        this.name = name;
        this.age = age;
        this.address = address;
        this.contactNumber = contactNumber;
        this.bloodGroup = bloodGroup;
        this.dateOfLastDonation = dateOfLastDonation;
    }

    // Getters
    public String getBloodGroup() {
        return bloodGroup;
    }

    public LocalDate getDateOfLastDonation() {
        return dateOfLastDonation;
    }

    // Method to display donor details
    public void displayDonorDetails() {
        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("Date of Last Donation: " +
dateOfLastDonation);
            System.out.println("--------------------------");
        }
    }

    public class DonorManagement {

        // Method to write donor objects to file
        public static void writeDonorsToFile(ArrayList<Donor> donors,
String filename) {
            try (ObjectOutputStream oos = new
ObjectOutputStream(new FileOutputStream(filename))) {
                for (Donor donor : donors) {
                    oos.writeObject(donor);
                }
            } catch (IOException e) {
                System.out.println("An error occurred while writing to
file.");
                e.printStackTrace();
            }
        }

        // Method to read donor objects from file and filter based on
criteria
        public static void readDonorsFromFile(String filename) {
            try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(filename))) {
                while (true) {
                    Donor donor = (Donor) ois.readObject();
```

```
                // Filter criteria: Blood group is A+ve and not donated in
the last 6 months
                if (donor.getBloodGroup().equals("A+ve") &&
Period.between(donor.getDateOfLastDonation(),
LocalDate.now()).getMonths() >= 6) {
                    donor.displayDonorDetails();
                }
            }
        } catch (EOFException e) {
            // End of file reached
        } catch (IOException | ClassNotFoundException e) {
            System.out.println("An error occurred while reading from
file.");
            e.printStackTrace();
        }
    }

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

        // Creating donor objects
        donors.add(new Donor("John Doe", 28, "123 Main St, Pune",
"1234567890", "A+ve", LocalDate.of(2023, 1, 10)));
        donors.add(new Donor("Jane Smith", 32, "456 Park Ave,
Pune", "0987654321", "B+ve", LocalDate.of(2024, 2, 15)));
        donors.add(new Donor("Raj Kumar", 25, "789 Hill Rd, Pune",
"1122334455", "A+ve", LocalDate.of(2023, 7, 1)));
        donors.add(new Donor("Priya Singh", 30, "1010 Lakeview,
Pune", "5566778899", "O+ve", LocalDate.of(2024, 3, 20)));
        donors.add(new Donor("Amit Patil", 29, "202 Riverbend,
Pune", "9988776655", "A+ve", LocalDate.of(2022, 12, 25)));

        // Writing donor objects to file
```

```
        String filename = "donors.dat";
        writeDonorsToFile(donors, filename);

        // Reading and displaying filtered donors from file
        System.out.println("Donors with blood group A+ve who
haven't donated in the last 6 months:");
        readDonorsFromFile(filename);
    }
}
```

Slip No 27

1. Define an Employee class with attributes getSalary() method which returns salary withdrawn by particular employee. write class Manger which extends Employee, override getSalary() method which will return salary of manager by adding travelling allowance , house rent allowance etc.

```
class Employee {
    protected double baseSalary;

    // Constructor to initialize base salary
    public Employee(double baseSalary) {
        this.baseSalary = baseSalary;
    }

    // Method to get the salary of the employee
    public double getSalary() {
        return baseSalary;
    }
}
```

```
class Manager extends Employee {
    private double travelAllowance;
    private double houseRentAllowance;

    // Constructor to initialize base salary and allowances
    public Manager(double baseSalary, double
travelAllowance, double houseRentAllowance) {
        super(baseSalary); // Calling the constructor of the
superclass (Employee)
        this.travelAllowance = travelAllowance;
        this.houseRentAllowance = houseRentAllowance;
    }

    // Overriding the getSalary method to include
allowances
    @Override
    public double getSalary() {
        return baseSalary + travelAllowance +
houseRentAllowance;
    }
}

public class Main {
    public static void main(String[] args) {
        // Creating an Employee object
        Employee employee = new Employee(50000);
        System.out.println("Employee's Salary: $" +
employee.getSalary());
```

```
        // Creating a Manager object
        Manager manager = new Manager(60000, 5000,
10000);
        System.out.println("Manager's Salary: $" +
manager.getSalary());
    }
}
```

2. Write program to accept string as command line arguments check whether it is file or directory also perform operation as follows: - 1. if it is directory delete all text files in that directory. Confirm delete operation from user before deleting text files. Also display count showing number of files deleted if any, from directory.
2. If is is file display various details of that file.
(command line java F_name txtfilepath(

```
import java.io.File;
import java.io.IOException;
import java.util.Scanner;

public class FileDirectoryOperations {
    public static void main(String[] args) {
        // Check if command-line argument is provided
        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 fileOrDir = new File(path);

// Check if the path is a directory
if (fileOrDir.isDirectory()) {
        System.out.println(path + " is a directory.");
        File[] files = fileOrDir.listFiles((dir, name) ->
name.toLowerCase().endsWith(".txt"));

        if (files != null && files.length > 0) {
            System.out.println("There are " + files.length + " .
txt files in this directory.");

            // Ask for confirmation to delete .txt files
            Scanner scanner = new Scanner(System.in);
            System.out.print("Do you want to delete all .txt
files in this directory? (yes/no): ");
            String response =
scanner.nextLine().trim().toLowerCase();

            if (response.equals("yes")) {
                int deleteCount = 0;
                for (File txtFile : files) {
                    if (txtFile.delete()) {
                        deleteCount++;
                    }
                }
```

```
                System.out.println(deleteCount + " .txt files
deleted from the directory.");
            } else {
                System.out.println("No files were deleted.");
            }
        } else {
            System.out.println("No .txt files found in the
directory.");
        }
    }
    // Check if the path is a file
    else if (fileOrDir.isFile()) {
        System.out.println(path + " is a file.");
        System.out.println("File details:");
        System.out.println("Name: " +
fileOrDir.getName());
        System.out.println("Path: " + fileOrDir.getPath());
        System.out.println("Absolute Path: " +
fileOrDir.getAbsolutePath());
        System.out.println("Size: " + fileOrDir.length() + "
bytes");
        System.out.println("Last Modified: " +
fileOrDir.lastModified());
        System.out.println("Readable: " +
fileOrDir.canRead());
        System.out.println("Writable: " +
fileOrDir.canWrite());
        System.out.println("Executable: " +
fileOrDir.canExecute());
```

```
        }
        // If the path does not exist
        else {
            System.out.println(path + " does not exist.");
        }
    }
}
```

Slip No.28

1.Write java program that reads on file name from user then display information about whether file exist , whether file is readable , whether file is writable , the type of file and lenghth of file in bytes. (Use CommandLine java C_name txtfilename)

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


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


        // Read file name from user
        System.out.print("Enter the file name: ");
        String fileName = scanner.nextLine();


        // Create File object
        File file = new File(fileName);
```

```
// Check if file exists
if (file.exists()) {
    System.out.println("File exists.");

    // Check if file is readable
    if (file.canRead()) {
        System.out.println("File is readable.");
    } else {
        System.out.println("File is not readable.");
    }

    // Check if file is writable
    if (file.canWrite()) {
        System.out.println("File is writable.");
    } else {
        System.out.println("File is not writable.");
    }

    // Determine the type of file
    if (file.isDirectory()) {
        System.out.println("It is a directory.");
    } else if (file.isFile()) {
        System.out.println("It is a regular file.");
```

```
        } else {
            System.out.println("It is neither a regular file nor a directory.");
        }

        // Display the length of the file in bytes
        if (file.exists()) {
            System.out.println("File length: " + file.length() + " bytes.");
        } else {
            System.out.println("File does not exist.");
        }

        scanner.close();
    }
}
```

2. Write program to called SwingTempratureConveter to convert temperature values between celsius and Farenheit. User can enter either celsius or Farenheit values, in floating-point number. Hint:- To diaplay flotting point number in specific format use static method String.format()which has same format printf() for example , String.format("%.1f",1.234)returns String "1.2".

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

public class SwingTemperatureConverter extends JFrame {
```

```
    private JTextField celsiusField;
    private JTextField fahrenheitField;

    public SwingTemperatureConverter() {
        setTitle("Temperature Converter");
        setSize(300, 150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new GridLayout(3, 2));

        // Celsius Label and Text Field
        JLabel celsiusLabel = new JLabel("Celsius:");
        celsiusField = new JTextField();
        add(celsiusLabel);
        add(celsiusField);

        // Fahrenheit Label and Text Field
        JLabel fahrenheitLabel = new JLabel("Fahrenheit:");
        fahrenheitField = new JTextField();
        add(fahrenheitLabel);
        add(fahrenheitField);

        // Convert Button
        JButton convertButton = new JButton("Convert");
```

```
        add(convertButton);

        // Reset Button
        JButton resetButton = new JButton("Reset");
        add(resetButton);

        // Action listener for the Convert button
        convertButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                convertTemperature();
            }
        });

        // Action listener for the Reset button
        resetButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                resetFields();
            }
        });

        setVisible(true);
    }
```

```
private void convertTemperature() {
    try {
        if (!celsiusField.getText().isEmpty()) {
            // Convert Celsius to Fahrenheit
            double celsius = Double.parseDouble(celsiusField.getText());
            double fahrenheit = (celsius * 9/5) + 32;
            fahrenheitField.setText(String.format("%.1f", fahrenheit));
        } else if (!fahrenheitField.getText().isEmpty()) {
            // Convert Fahrenheit to Celsius
            double fahrenheit =
Double.parseDouble(fahrenheitField.getText());
            double celsius = (fahrenheit - 32) * 5/9;
            celsiusField.setText(String.format("%.1f", celsius));
        }
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(this, "Please enter a valid
number.", "Invalid Input", JOptionPane.ERROR_MESSAGE);
    }
}

private void resetFields() {
    celsiusField.setText("");
    fahrenheitField.setText("");
}
```

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

Slip No 29

1.Write java program create class customer (custno, custname, custaddress, custsddr). Write method to search customer name with given contact number and display the details. (On terminal)

```
import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;


class Customer {
    private int custNo;
    private String custName;
    private String custAddress;
    private String custContactNo;

    // Constructor to initialize Customer details
    public Customer(int custNo, String custName, String custAddress,
String custContactNo) {
        this.custNo = custNo;
        this.custName = custName;
        this.custAddress = custAddress;
```

```
        this.custContactNo = custContactNo;
    }

    // Getter method for contact number
    public String getCustContactNo() {
        return custContactNo;
    }

    // Method to display customer details
    public void displayCustomerDetails() {
        System.out.println("Customer Number: " + custNo);
        System.out.println("Customer Name: " + custName);
        System.out.println("Customer Address: " + custAddress);
        System.out.println("Customer Contact Number: " +
custContactNo);
        System.out.println("----------------------------");
    }
}

public class CustomerSearch {
    public static void main(String[] args) {
        // Creating a list of customers
        List<Customer> customers = new ArrayList<>();
        customers.add(new Customer(1, "Alice", "123 Elm St",
"555-1234"));
```

```
        customers.add(new Customer(2, "Bob", "456 Maple Ave",
"555-5678"));
        customers.add(new Customer(3, "Charlie", "789 Oak Dr",
"555-9876"));

        Scanner scanner = new Scanner(System.in);
        // Asking user to input the contact number to search
        System.out.print("Enter the contact number to search: ");
        String contactNo = scanner.nextLine();

        boolean found = false;
        // Searching for the customer with the given contact number
        for (Customer customer : customers) {
            if (customer.getCustContactNo().equals(contactNo)) {
                System.out.println("Customer found:");
                customer.displayCustomerDetails();
                found = true;
                break;
            }
        }

        if (!found) {
```

```
            System.out.println("No customer found with the contact number:
" + contactNo);
        }


        scanner.close();
    }
}
```

Slip No.29

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

```
import java.util.ArrayList;

import java.util.Scanner;


class Customer {
    private int custNo;
    private String custName;
    private String custAddr;
    private String custNumber;

    // Constructor
    public Customer(int custNo, String custName, String
custAddr, String custNumber) {
        this.custNo = custNo;
```

```
        this.custName = custName;
        this.custAddr = custAddr;
        this.custNumber = custNumber;
    }

    // Getters
    public int getCustNo() {
        return custNo;
    }

    public String getCustName() {
        return custName;
    }

    public String getCustAddr() {
        return custAddr;
    }

    public String getCustNumber() {
        return custNumber;
    }
```

```
// Method to display customer details
public void displayDetails() {
    System.out.println("Customer Number: " + custNo);
    System.out.println("Customer Name: " + custName);
        System.out.println("Customer Address: " + custAddr);
        System.out.println("Customer Contact Number: " +
custNumber);
    }
}

public class CustomerSearchApp {
    private static ArrayList<Customer> customerList = new
ArrayList<>();

    // Method to search for a customer by contact number
    public static void searchCustomerByNumber(String
contactNumber) {
        for (Customer customer : customerList) {
            if
(customer.getCustNumber().equals(contactNumber)) {
                System.out.println("Customer found:");
                customer.displayDetails();
                return;
```

```
            }
        }
        System.out.println("No customer found with contact
number: " + contactNumber);
    }

    public static void main(String[] args) {
        // Adding some customers to the list
        customerList.add(new Customer(1, "John Doe", "123 Elm
Street", "1234567890"));
        customerList.add(new Customer(2, "Jane Smith", "456
Maple Avenue", "0987654321"));
        customerList.add(new Customer(3, "Alice Johnson", "789
Oak Road", "1112223333"));

        // Input contact number to search
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the contact number to search:");
        String contactNumber = scanner.nextLine();

        // Search for the customer
        searchCustomerByNumber(contactNumber);
    }
```

}

3. Write program to create super class vehicle having members company nad price. Derive two different classes LightMotorvehicle and HeavyMotorVehicle . Accept the information of n vehicles and display information in appropriate format , while taking data ask user about type of vehicle first.

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

// Superclass Vehicle
class Vehicle {
    protected String company;
    protected double price;

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

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

// Subclass LightMotorVehicle
class LightMotorVehicle extends Vehicle {
```

```
    private double mileage; // in kilometers per liter

    public LightMotorVehicle(String company, double price,
double mileage) {
        super(company, price);
        this.mileage = mileage;
    }

    @Override
    public void displayDetails() {
        System.out.println("Vehicle Type: Light Motor
Vehicle");
        super.displayDetails();
        System.out.println("Mileage: " + mileage + " km/l");
        System.out.println("----------------------------
-------");
    }
}

// Subclass HeavyMotorVehicle
class HeavyMotorVehicle extends Vehicle {
    private double capacity; // in tons

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

```
    @Override
    public void displayDetails() {
        System.out.println("Vehicle Type: Heavy Motor
Vehicle");
        super.displayDetails();
        System.out.println("Capacity: " + capacity + " tons");
        System.out.println("-----------------------------
-------");
    }
}

// Main class
public class VehicleInfoApp {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ArrayList<Vehicle> vehicleList = new ArrayList<>();

        System.out.print("Enter the number of vehicles: ");
        int n = Integer.parseInt(scanner.nextLine());

        for (int i = 0; i < n; i++) {
            System.out.println("\nEntering details for vehicle "
+ (i + 1) + ":");
            System.out.print("Enter vehicle type (1 for Light
Motor Vehicle, 2 for Heavy Motor Vehicle): ");
            int type = Integer.parseInt(scanner.nextLine());

            System.out.print("Enter company name: ");
            String company = scanner.nextLine();
```

```
            System.out.print("Enter price: $");
            double price =
Double.parseDouble(scanner.nextLine());

            if (type == 1) {
                System.out.print("Enter mileage (in km/l): ");
                double mileage =
Double.parseDouble(scanner.nextLine());
                vehicleList.add(new
LightMotorVehicle(company, price, mileage));
            } else if (type == 2) {
                System.out.print("Enter capacity (in tons): ");
                double capacity = Double.parseDouble(scanner.
nextLine());
                vehicleList.add(new
HeavyMotorVehicle(company, price, capacity));
            } else {
                System.out.println("Invalid vehicle type entered.
Skipping this entry.");
            }
        }

        // Displaying all vehicle details
        System.out.println("\n--- Vehicle Details ---");
        for (Vehicle vehicle : vehicleList) {
            vehicle.displayDetails();
        }
```

```
        scanner.close();
    }
}
```

Slip No.30

1.Write program define class person with data mebers personname , adharno,Panono. Accept inofrmation of 5 objects and display appropriate information(use this keyword)

```
import java.util.Scanner;


class Person {
    private String personName;
    private String adharNo;
    private String panoNo;


    // Constructor
    public Person(String personName, String adharNo, String panoNo) {
        this.personName = personName;
        this.adharNo = adharNo;
        this.panoNo = panoNo;
    }


    // Method to display person details
```

```
public void displayDetails() {
    System.out.println("Person Name: " + this.personName);
    System.out.println("Adhar No: " + this.adharNo);
    System.out.println("PAN No: " + this.panoNo);
    }
}


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

        // Accept information for 5 persons
        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 Adhar No: ");
            String adhar = scanner.nextLine();
```

```
            System.out.print("Enter PAN No: ");
            String pan = scanner.nextLine();

            // Create Person object using constructor
            persons[i] = new Person(name, adhar, pan);
        }

        // Display the details of all persons
        System.out.println("\nDisplaying Information of 5 Persons:
");
        for (int i = 0; i < 5; i++) {
            System.out.println("Details of person " + (i + 1) + ":");
            persons[i].displayDetails();
            System.out.println("----------------------------------
-");
        }
    }
}
```

3. Write program that create user interface to perform integer division. The user enters two numbers in text fields, Number1 and Number2 . The division of Number1 And Number2 is displayed in the Result Field when divide button is clicked. If

number1 or Number2 were not an integer, the program throw NumberFormatException. If Number2 were zero , the prgoram would throw an Arithmatic Exception. Display Exception in message in dialog box.

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

public class DivisionApp extends JFrame {

    private JTextField number1Field;
    private JTextField number2Field;
    private JTextField resultField;
    private JButton divideButton;

    public DivisionApp() {
        // Set up the frame
        setTitle("Integer Division");
        setSize(400, 200);

setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new GridLayout(4, 2));

        // Create labels and text fields
        JLabel number1Label = new JLabel("Number 1:");
        number1Field = new JTextField();
        JLabel number2Label = new JLabel("Number 2:");
        number2Field = new JTextField();
```

```
        JLabel resultLabel = new JLabel("Result:");
        resultField = new JTextField();
        resultField.setEditable(false);  // Make result field
read-only

        // Create Divide button
        divideButton = new JButton("Divide");

        // Add action listener to the button
        divideButton.addActionListener(new
ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                performDivision();
            }
        });

        // Add components to the frame
        add(number1Label);
        add(number1Field);
        add(number2Label);
        add(number2Field);
        add(resultLabel);
        add(resultField);
        add(new JLabel()); // Empty cell for layout alignment
        add(divideButton);

        // Make the frame visible
        setVisible(true);
```

```
}

private void performDivision() {
    try {
        // Get the numbers from the text fields
        int number1 =
Integer.parseInt(number1Field.getText());
            int number2 =
Integer.parseInt(number2Field.getText());

            // Perform the division
            int result = number1 / number2;

            // Display the result
            resultField.setText(String.valueOf(result));
        } catch (NumberFormatException nfe) {
            // Handle the case where the input is not a valid
integer
            JOptionPane.showMessageDialog(this, "Please
enter valid integers in both fields.", "Invalid Input",
JOptionPane.ERROR_MESSAGE);
        } catch (ArithmeticException ae) {
            // Handle the case where division by zero occurs
            JOptionPane.showMessageDialog(this, "Division
by zero is not allowed.", "Arithmetic Error",
JOptionPane.ERROR_MESSAGE);
        }
    }
```

```
    public static void main(String[] args) {
        // Run the application
        new DivisionApp();
    }
}
```