**Problem Statement:** Write a Java program to calculate sum of last digits of two numbers.

**Objective:** To develop a Java program that takes two integer inputs, extracts their last digits, and computes the sum of these digits.

## Code:

import java.util.Scanner; class LastDigit {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in); System.out.println("Enter first number:"); int num1=sc.nextInt(); System.out.println("Enter second number:"); int num2=sc.nextInt();

int sum=(num1%10)+(num2%10); System.out.println("Sum of last 2 digits: "+sum);

}

}

# Output:

Enter first number:

38

Enter second number:

46

Sum of last 2 digits: 14

**Problem Statement:** Write a Java program to calculate sum of even digits.

**Objective:** To develop a Java program that takes an integer as input, extracts its digits, identifies the even digits, and computes their sum.

## Code:

import java.util.Scanner; class EvenDigit {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in); System.out.println("Enter a number:"); int num=sc.nextInt();

int remainder,sum=0; while(num!=0)

{

remainder=num%10; if (remainder%2==0)

{

sum+=remainder;

}

num/=10;

}

System.out.println("Sum of even digits: "+sum);

}

}

# Output:

Enter a number: 75612

Sum of even digits: 8

**Problem Statement:** Write a Java program to calculate cyclic sum of given number. **Objective:** To develop a Java program that calculates the cyclic sum of a given number. **Code:**

import java.util.Scanner; class CyclicSum {

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

int num=sc.nextInt(); int totalSum=0;

String numStr=Integer.toString(num); for (int i=0; i<numStr.length(); i++) {

int subNumber=Integer.parseInt(numStr.substring(i)); int sum=0;

while (subNumber>0) { sum+=subNumber%10; subNumber/=10;

}

totalSum += sum;

}

System.out.println("Cyclic sum: "+totalSum);

}

}

# Output:

Enter a num:51364 Cyclic sum: 60

**Problem Statement:** Write a Java program to generate PIN according to following conditions:

* If the sum of length of input string is single digit, then PIN=SUM.
* If the sum is two or more digits then again calculate the sum of digits until the sum is single digit.

**Objective:** To develop a Java program that generates a PIN based on the length of an input string.

## Code:

import java.util.Scanner; class StringPin {

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in); System.out.println("Enter a string:"); String str=sc.nextLine();

int sum=str.replace(" ", "").length(); while (sum>=10) {

int temp=0; while (sum>0) {

temp+=sum % 10; sum/=10;

}

sum=temp;

}

System.out.println("Generated PIN: "+sum);

}

}

# Output:

Enter a string: Java Practical Generated PIN: 4

**Problem Statement:** Write a Java program to create PIN using three given numbers. Rule for generating the PIN:

* The three given input numbers will always consist of three digits.
* The PIN should be made up of 4 digits.
* The unit(ones) position of the PIN should be the least of the unit(ones) position of the three input numbers.
* The tens position of the PIN should be the least of the tens position of the three input numbers.
* The hundred position of the PIN should be the least of the hundred position of the three input numbers.
* The thousand position of the PIN should be the maximum of all the digits in the three input numbers.

**Objective:** To develop a Java program that generates a PIN using three given numbers.

## Code:

import java.util.Scanner; class GeneratePin

{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in); System.out.println("Enter first number:"); int num1=sc.nextInt(); System.out.println("Enter second number:"); int num2=sc.nextInt(); System.out.println("Enter third number:"); int num3=sc.nextInt();

int pin=0,m1=0,m2=0,m3=0,d,n1=num1,n2=num2,n3=num3;

m1=Math.max((n1%10),Math.max((n2%10),(n3%10)));

d=Math.min((n1%10),Math.min((n2%10),(n3%10))); pin=pin+d\*1;

n1=n1/10; n2=n2/10; n3=n3/10;

m2=Math.max((n1%10),Math.max((n2%10),(n3%10)));

d=Math.min((n1%10),Math.min((n2%10),(n3%10))); pin=pin+d\*10;

n3=n3/10;

m3=Math.max((n1%10),Math.max((n2%10),(n3%10)));

d=Math.min((n1%10),Math.min((n2%10),(n3%10))); pin=pin+d\*100;

n1=n1/10; n2=n2/10; n3=n3/10;

pin=pin+(1000\*Math.max(m1,Math.max(m2,m3))); System.out.println("4-digit PIN: "+pin);

}

}

# Output:

Enter first number:

625

Enter second number:

794

Enter third number:

836

4-digit PIN: 9624

**Problem Statement:** Write a Java program to find nth position Fibonacci number.

**Objective:** To develop a Java program that calculates the Nth Fibonacci number based on user input.

## Code:

import java.util.Scanner; class Fibonacci {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in); System.out.println("Enter the the position:"); int pos=sc.nextInt();

int a=0,b=1,sum=0,i; if(pos<=0)

{

System.out.println("Invalid Postion");

}

else if(pos==1)

{

System.out.println("Number: "+a);

}

else if(pos==2)

{

System.out.println("Number: "+b);

}

else

{

for(i=3; i<=pos; i++)

{

sum=a+b; a=b; b=sum;

}

System.out.println("Number: "+sum);

}

}

}

# Output:

Enter the position:

7

Number: 8

**Problem Statement:** Write a Java program to create a class Box that uses a parameterized constructor to initialize the dimensions of the box. The dimensions of the box are width, height, depth. The class should have method that can return the volume of the box. Create an object of the Box class and test the functionalities.

**Objective:** To develop a Java program that defines a Box class with a parameterized constructor to initialize its dimensions. The class should include a method to calculate and return the volume of the box.

## Code:

import java.util.Scanner; class Box {

int width,height,depth;

Box(int width,int height,int depth)

{

this.width=width; this.height=height; this.depth=depth;

}

int getvolume()

{

return width\*height\*depth;

}

}

class TestBox {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in); System.out.println("Enter width:"); int width=sc.nextInt(); System.out.println("Enter height:"); int height=sc.nextInt(); System.out.println("Enter depth:"); int depth=sc.nextInt();

Box obj=new Box(width, height, depth); System.out.println("Volume of Box: "+obj.getvolume());

}

}

**Name:** Rishi Rawat **Course:** MCA **Section:** B

**University Roll No:** 2401219

# Output:

Enter width:

8

Enter height:

4

Enter depth:

6

Volume of Box: 192

**Name:** Rishi Rawat **Course:** MCA **Section:** B

**University Roll No:** 2401219

**Problem Statement:** Write a Java program to create a new class called Calculator with the following methods:

1. A static method called powering (int n1, int n2): - This method should return n1 to the power n2.
2. A static method called division (double n1, double n2): - This method should return n1 divided by n2.

Invoke both the methods and test the functionalities.

**Objective:** To develop a Java program that defines a Calculator class with two static methods.

## Code:

import java.util.Scanner; class Calculator {

static int powering(int n1,int n2)

{

if (n2==0) { return 1;

}

else {

int power=1;

for(int i=1; i<=n2; i++)

{

power=power\*n1;

}

return power;

}

}

static double division(double n1,double n2)

{

return n1/n2;

}

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in); System.out.println("Powering Method:-"); System.out.println("Enter number:");

int num1=sc.nextInt(); System.out.println("Enter its power:"); int num2=sc.nextInt();

System.out.println("Result: "+powering(num1, num2)); System.out.println("Division Method:-"); System.out.println("Enter first number:");

double n1=sc.nextDouble(); System.out.println("Enter second number:"); double n2=sc.nextDouble(); System.out.println("Result: "+division(n1, n2));

}

}

# Output:

Powering Method: - Enter number:

4

Enter its power:

3

Result: 64 Division Method: - Enter first number: 38

Enter second number:

8

Result: 4.75

**Problem Statement:** Write a Java program to calculate the factorial of a number. **Objective:** To develop a Java program that calculates the factorial of given number. **Code:**

import java.util.Scanner; class Factorial {

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

int fact=1;

for(int i=num; i>1; i--) { fact\*=i;

}

System.out.println("Factorial is: "+fact);

}

}

# Output:

Enter a number: 6 Factorial is: 720

**Problem Statement:** Write a Java program check whether a number is prime or not.

**Objective:** To develop a Java program that checks whether a given number is prime or not.

## Code:

import java.util.Scanner; class Prime {

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

for(int i=2; i<num; i++)

{

if(num%i==0) {

System.out.println(num+" is not Prime"); return;

}

}

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

}

}

# Output:

Enter a number: 7 7 is prime

Enter a number: 8 8 is not Prime

**Problem Statement:** Write a Java program to calculate the area of following shapes: circle, rectangle and square. Overload the area method.

**Objective:** To develop a Java program that calculates the area of different shapes (circle, rectangle, and square) using method overloading**.**

## Code:

import java.util.Scanner; class Area {

void area(int l, int b)

{

System.out.println("Area of rectangle: "+l\*b);

}

void area(int s)

{

System.out.println("Area of square: "+s\*s);

}

void area(double r)

{

System.out.println("Area of circle: "+3.14\*r\*r);

}

}

class Calculate {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in); Area obj=new Area();

System.out.println("Enter length of a rectangle:"); int l=sc.nextInt();

System.out.println("Enter breadth of a rectangle:"); int b=sc.nextInt();

obj.area(l, b);

System.out.println("Enter length of a square:"); int s=sc.nextInt();

obj.area(s);

System.out.println("Enter radius of a circle:"); double r=sc.nextDouble();

obj.area(r);

}

}

# Output:

Enter length of a rectangle: 6

Enter breadth of a rectangle: 8

Area of rectangle: 48 Enter length of a square: 16

Area of square: 256 Enter radius of a circle: 13

Area of circle: 530.66

**Name:** Rishi Rawat **Course:** MCA **Section:** B

**University Roll No:** 2401219

**Problem Statement:** Write a Java program to insert an element in an array. Use Scanner class for inputs.

**Objective:** To develop a Java program that insert an element in an array using Scanner class.

## Code:

import java.util.Scanner; class ArrayInsertion {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in); int[] arr=new int[100];

System.out.print("Enter the size of array: "); int size=sc.nextInt();

int n=-1;

System.out.println("1: Insert\n2: Display\n3: End Program"); while(true)

{

System.out.print("\nEnter choice: "); int choice=sc.nextInt();

switch (choice) { case 1:

System.out.print("Enter index: "); int index=sc.nextInt();

if (index<0 || index>size-1 || index>n+1)

{

System.out.println("Cannot insert number.\n");

}

else if (index==n) { System.out.print("Enter number: "); arr[index]=sc.nextInt(); System.out.println("Number inserted."); n++;

}

else {

for(int i=n; i>=index; i--)

{

arr[i+1]=arr[i];

}

System.out.print("Enter number: "); arr[index]=sc.nextInt(); System.out.println("Number inserted."); n++;

}

break; case 2:

System.out.println("Array Elements:"); for(int i=0; i<=n; i++)

{

System.out.print(arr[i]+" ");

}

break; case 3:

System.out.println("Program Terminated."); return;

default:

System.out.println("Invalid choice.\n"); break;

}

}

}

}

# Output:

Enter the size of array: 7 1: Insert

2: Display

3: End Program

Enter choice: 1

Enter index: 0

Enter number: 8 Number inserted.

Enter choice: 1

Enter index: 1

Enter number: 6 Number inserted.

Enter choice: 1

Enter index: 2

Enter number: 5 Number inserted.

Enter choice: 1

Enter index: 3

Enter number: 9 Number inserted.

Enter choice: 2 Array Elements: 8 6 5 9

Enter choice: 3 Program Terminated.

**Problem Statement:** Write a Java program to create a class Author with the following information: -

## Member variables:

Name (String), email (String) and gender (char). Parameterized Constructor: To initialize the variables.

Create a class Book with the following information: -

## Member variables:

Name (String), author (of the class Author you have just created), price (double) and qtyInStock (int).

Parameterized Constructor: To initialize the variables.

Getters and Setters method for all the member variables.

In the main method create a book object and print all details of the book.

**Objective:** To develop a Java program that demonstrates object composition by creating two classes, Author and Book**.** This program aims to illustrate the principles of encapsulation, constructors, and getter-setter methods while managing relationships between objects in Java.

## Code:

class Author {

private String name; private String email; private char gender;

Author(String name, String email, char gender) { this.name = name;

this.email = email; this.gender = gender;

}

public String getName() { return name;

}

public String getEmail() { return email;

}

public char getGender() { return gender;

}

public String toString() {

return "Author: "+name+" ("+gender+"), Email: "+email;

}

}

int qtyInStock;

Book(String name, Author author, double price, int qtyInStock) { this.name = name;

this.author = author; this.price = price; this.qtyInStock = qtyInStock;

}

public String getName() { return name;

}

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

}

public Author getAuthor() { return author;

}

public double getPrice() { return price;

}

public void setPrice(double price) { this.price = price;

}

public int getQtyInStock() { return qtyInStock;

}

public void setQtyInStock(int qtyInStock) { this.qtyInStock = qtyInStock;

}

public String toString() {

return "Book: "+name+"\n"+author+"\nPrice: $"+price+"\nStock: "+qtyInStock;

}

}

public class BookStore {

public static void main(String[] args) {

Author author = new Author("J.K. Rowling", ["jkrowling8@gamil.com",](mailto:jkrowling8@gamil.com) 'F'); Book book = new Book("Harry Potter", author, 29.99, 100); System.out.println(book);

}

}

# Output:

Book: Harry Potter

Author: J.K. Rowling (F), Email: [jkrowling8@gamil.com](mailto:jkrowling8@gamil.com) Price: $29.99

Stock: 100

**Problem Statement:** Write a Java program to create a class named Animal which includes method like eat() and sleep(). Create a child class of Animal named Bird and override the parent class methods. Add a new method named fly(). Create instances of Animal class and Bird class and invoke the all the methods using objects.

**Objective:** To develop a Java program that demonstrates inheritance and method overriding by creating a parent class named Animal and a child class named Bird.

## Code:

class Animal

{

public void eat()

{

System.out.println("Animal is eating.");

}

public void sleep()

{

System.out.println("Animal is sleeping.");

}

}

class Bird extends Animal

{

public void eat()

{

System.out.println("Bird is pecking at food.");

}

public void sleep()

{

System.out.println("Bird is sleeping on a perch.");

}

public void fly()

{

System.out.println("Bird is flying in the sky.");

}

}

public class TestAnimal

{

public static void main(String[] args)

{

Animal animal = new Animal();

System.out.println("Animal Object:"); animal.eat();

animal.sleep(); System.out.println(); Bird bird = new Bird();

System.out.println("Bird Object:"); bird.eat();

bird.sleep();

bird.fly();

}

}

# Output:

Animal Object:

Animal is eating.

Animal is sleeping.

Bird Object:

Bird is pecking at food. Bird is sleeping on a perch. Bird is flying in the sky.

**Problem Statement:** Write a Java program to create a jacked array and calculate the sum of even elements and odd elements.

**Objective:** To develop a Java program that creates a jagged array (a 2D array with variable column sizes) and calculates the sum of even and odd elements.

## Code:

import java.util.Scanner; public class JaggedArray {

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

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

int[][] jaggedArray=new int[rows][]; for (int i=0; i<rows; i++) {

System.out.print("Enter the number of elements in row "+(i+1)+": "); int cols=sc.nextInt();

jaggedArray[i]=new int[cols];

System.out.println("Enter "+cols+" elements for row "+(i + 1)+":"); for (int j=0; j<cols; j++) {

jaggedArray[i][j]=sc.nextInt();

}

}

int evenSum=0,oddSum=0; for (int i=0; i<rows; i++) {

for (int j=0; j<jaggedArray[i].length; j++) { if (jaggedArray[i][j]%2==0) {

evenSum+=jaggedArray[i][j];

}

else {

oddSum+=jaggedArray[i][j];

}

}

}

System.out.println("\nSum of even elements: "+evenSum); System.out.println("Sum of odd elements: "+oddSum); sc.close();

}

}

# Output:

Enter the number of rows: 3

Enter the number of elements in row 1: 3 Enter 3 elements for row 1:

7 5 6

Enter the number of elements in row 2: 2 Enter 2 elements for row 2:

4 2

Enter the number of elements in row 3: 4 Enter 4 elements for row 3:

8 3 1 9

Sum of even elements: 20 Sum of odd elements: 25

**Problem Statement:** Write a Java program to merge data of two files into one file.

**Objective:** To develop a Java program that merges two files into one files.

## Code:

import java.io.\*;

public class MergeFileData {

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

PrintWriter pw=new PrintWriter("third.txt");

BufferedReader br=new BufferedReader(new FileReader("one.txt")); String s1=br.readLine();

while(s1!=null) { pw.println(s1); pw.flush(); s1=br.readLine();

}

br=new BufferedReader(new FileReader("two.txt")); s1=br.readLine();

while(s1!=null) { pw.println(s1); pw.flush(); s1=br.readLine();

}

System.out.println("Data merged and stored.");

}

catch(IOException e) { e.printStackTrace();

}

}

}

# Output:

Data merged and stored.

one.txt:

Content of first file.

two.txt

Content of second file.

third.txt

Content of first file. Content of second file.

**Problem Statement:** Write a Java program that performs the following tasks: Prompts the user to enter five lines of text from the console. Saves each line into a file called output.txt using the PrintWriter class. After writing the lines to the file, reads the content back from output.txt using the BufferedReader class. Finally, prints each line to the console with line numbers (e.g., “Line 1: <content>”).

**Objective:** The objective of this program is to develop a Java application that interacts with the user to input five lines of text, writes the input to a file named output.txt using the PrintWriter class, then reads the file content back using the BufferedReader class, and displays each line on the console prefixed with its corresponding line number.

## Code:

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

public class LineWriteRead {

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

PrintWriter pw=new PrintWriter("output.txt"); System.out.println("Enter 5 lines of text:"); for(int i=1; i<=5; i++) {

String str=sc.nextLine(); pw.println(str); pw.flush();

}

BufferedReader br=new BufferedReader(new FileReader("output.txt")); String str=br.readLine();

int i=1; while(str!=null) {

System.out.println("Line "+i+": "+str); str=br.readLine();

i++;

}

}

catch(IOException e) { e.printStackTrace();

}

}

}

# Output:

Enter 5 lines of text:

Rishi

Java Programming Apache Tomcat Eclipse IDE Assignment

Line 1: Rishi

Line 2: Java Programming Line 3: Apache Tomcat Line 4: Eclipse IDE

Line 5: Assignment

**Problem Statement:** Write a Java program that: Prompts the user to enter a sentence from the console. Writes the sentence into a file named sentence.txt using PrintWriter class. Reads the sentence back from sentence.txt using BufferedReader. Prints the reversed sentence to the console.

**Objective:** To create a Java application that prompts the user to input a sentence, writes the sentence into a file named sentence.txt using the PrintWriter class, reads the sentence back from the file using the BufferedReader class, and then displays the reversed sentence on the console.

## Code:

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

public class LineReverseFile {

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

PrintWriter pw=new PrintWriter("sentence.txt"); System.out.println("Enter a sentence:");

String str=sc.nextLine(); pw.println(str); pw.flush();

pw.close();

BufferedReader br=new BufferedReader(new FileReader("sentence.txt")); String str1=br.readLine();

StringBuffer sb=new StringBuffer(); if(str1!=null) {

String[] words=str1.trim().split("\\s+"); StringBuilder rev=new StringBuilder(); for(int i=words.length-1; i>=0; i--) {

rev.append(words[i]); if(i!=0) {

rev.append(" ");

} }

System.out.println("\nReversed sentence:"); System.out.println(rev.toString()); } }

catch(IOException e) { e.printStackTrace();

}

} }

# Output:

Enter a sentence: My name is Rishi

Reversed sentence: Rishi is name My

**Name:** Rishi Rawat **Course:** MCA **Section:** B

**University Roll No:** 2401219

**Problem Statement:** Write a Java program to implement Object Serialization and Deserialization.

**Objective:** To create a Java application that demonstrates Object Serialization and Deserialization.

## Code:

import java.io.\*;

class Product implements Serializable { int id;

String name;

Product(int id, String name) { this.id=id; this.name=name;

}

public void display() {

System.out.println("ID: "+id+", Name: "+name);

}

}

class ObjectSerialization {

public static void main(String[] args) throws Exception { Product p1=new Product(101,"Fan"); FileOutputStream fos=new FileOutputStream("f.ser");

ObjectOutputStream oos=new ObjectOutputStream(fos); oos.writeObject(p1);

System.out.println("Object Serialization done."); FileInputStream fis=new FileInputStream("f.ser"); ObjectInputStream ois=new ObjectInputStream(fis); Product p2=(Product)ois.readObject(); System.out.println(p2.id); System.out.println(p2.name);

}

}

# Output:

Object Serialization done. 101

Fan

**Problem Statement:** Write a Java program to write integers ranges from 1 to 999 line by line in number.txt file. Reads each integer and check whether it is prime or not. If number is prime then write this integer into prime.txt. Now read prime.txt and print the prime numbers.

**Objective:** To develop a Java application that writes integers from 1 to 999 line by line into a file named number.txt, reads each integer from the file, checks whether it is a prime number, writes all prime numbers into a separate file called prime.txt, and finally reads and displays the prime numbers from prime.txt on the console.

## Code:

import java.io.\*;

public class PrimeNumberFile {

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

PrintWriter numberWrite=new PrintWriter("number.txt"); for(int i=0; i<=999; i++) {

numberWrite.println(i);

}

numberWrite.flush();

System.out.println("Numbers from 1 to 999 written in number.txt"); BufferedReader numberRead=new BufferedReader(new FileReader("number.txt")); PrintWriter primeWrite=new PrintWriter("prime.txt");

String line; int prime=0;

while((line=numberRead.readLine())!=null) { int num=Integer.parseInt(line.trim()); for(int i=2; i<num; i++) {

if(num%i==0) { prime=1;

}

}

if(prime==0) { primeWrite.println(num+” ”);

}

prime=0;

}

numberRead.close(); primeWrite.close();

System.out.println("Prime numbers written to prime.txt");

BufferedReader primeRead=new BufferedReader(new FileReader("prime.txt"));

System.out.println("Prime numbers are:"); while((line=primeRead.readLine())!=null) {

System.out.println(line);

}

primeRead.close();

}

catch(IOException e) { e.printStackTrace();

}

}

}

# Output:

Numbers from 1 to 999 written in number.txt Prime numbers written to prime.txt

Prime numbers are:

0 1 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109

113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211 223 227 229

233 239 241 251 257 263 269 271 277 281 283 293 307 311 313 317 331 337 347 349 353

359 367 373 379 383 389 397 401 409 419 421 431 433 439 443 449 457 461 463 467 479

487 491 499 503 509 521 523 541 547 557 563 569 571 577 587 593 599 601 607 613 617

619 631 641 643 647 653 659 661 673 677 683 691 701 709 719 727 733 739 743 751 757

761 769 773 787 797 809 811 821 823 827 829 839 853 857 859 863 877 881 883 887 907

911 919 929 937 941 947 953 967 971 977 983 991 997

**Problem Statement:** Write a Java program to create two threads, one thread will print odd numbers with thread name and another thread will print only even numbers with thread name between 1 to 10.

**Objective:** To create a Java application that utilizes multithreading by defining two separate threads: one thread to print odd numbers and another thread to print even numbers between 1 and 10, each displaying the thread name along with the number during execution.

## Code:

class OddThread extends Thread { public void run() {

for (int i=1; i<=10; i+=2) { System.out.println(Thread.currentThread().getName()+" - Odd number: "+i); try {

Thread.sleep(100);

}

catch (InterruptedException e) { System.out.println(e.getMessage());

}

}

}

}

class EvenThread extends Thread { public void run() {

for (int i=2; i<=10; i+=2) { System.out.println(Thread.currentThread().getName()+" - Even number: "+i); try {

Thread.sleep(100);

}

catch (InterruptedException e) { System.out.println(e.getMessage());

}

}

}

}

public class OddEvenThread {

public static void main(String[] args) { OddThread oddThread=new OddThread(); EvenThread evenThread=new EvenThread(); oddThread.setName("OddThread");

evenThread.setName("EvenThread"); oddThread.start(); evenThread.start();

}

}

# Output:

OddThread - Odd number: 1 EvenThread - Even number: 2 EvenThread - Even number: 4 OddThread - Odd number: 3 OddThread - Odd number: 5 EvenThread - Even number: 6 OddThread - Odd number: 7 EvenThread - Even number: 8 OddThread - Odd number: 9 EvenThread - Even number: 10

**Problem Statement:** Write a Java program to create two threads t1 and t2. Print the priority of t1, t2 and main threads. Set the priority of t2 thread maximum and t1 thread minimum. The job of t1 thread is to print prime numbers between 1 to 100 and the job for t2 thread is to print palindrome numbers between 111 to 999.

**Objective:** To develop a Java application that demonstrates thread prioritization by creating two threads: one (t1) to print prime numbers between 1 and 100 and another (t2) to print palindrome numbers between 111 and 999. The program sets and displays the priorities of the main thread, t1, and t2, assigning minimum priority to t1 and maximum priority to t2, and then executes both threads.

## Code:

class PrimeThread extends Thread { public void run() {

System.out.println(Thread.currentThread().getName()+" started"); for (int i=1; i<=100; i++) {

if (isPrime(i)) {

System.out.println(Thread.currentThread().getName()+" - Prime: "+i);

}

}

}

private boolean isPrime(int num) { if (num<=1) return false;

if (num==2) return true;

if (num % 2==0) return false;

for (int i=3; i<=Math.sqrt(num); i+=2) { if (num%i==0) return false;

}

return true;

}

}

class PalindromeThread extends Thread { public void run() {

System.out.println(Thread.currentThread().getName()+" started"); for (int i=111; i<=999; i++) {

if (isPalindrome(i)) {

System.out.println(Thread.currentThread().getName()+" - Palindrome: "+i);

}

}

}

int original=num; int reverse=0; while (num!=0) {

int digit=num%10; reverse=reverse\*10+digit; num/=10;

}

return original==reverse;

}

}

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

PrimeThread t1=new PrimeThread(); PalindromeThread t2=new PalindromeThread();

System.out.println("Main thread priority: "+Thread.currentThread().getPriority()); System.out.println("Default t1 priority: "+t1.getPriority()); System.out.println("Default t2 priority: "+t2.getPriority()); t1.setPriority(Thread.MIN\_PRIORITY); t2.setPriority(Thread.MAX\_PRIORITY);

System.out.println("Updated t1 priority: "+t1.getPriority()); System.out.println("Updated t2 priority: "+t2.getPriority()); t1.setName("PrimeThread"); t2.setName("PalindromeThread");

t1.start();

t2.start();

}

}

**Course:** MCA

## Section: B

**University Roll No:** 2401219

# Output:

Main thread priority: 5 Default t1 priority: 5 Default t2 priority: 5 Updated t1 priority: 1 Updated t2 priority: 10 PalindromeThread started PrimeThread started

PalindromeThread - Palindrome: 111 PrimeThread - Prime: 2 PalindromeThread - Palindrome: 121 PrimeThread - Prime: 3 PalindromeThread - Palindrome: 131 PrimeThread - Prime: 5 PalindromeThread - Palindrome: 141 PrimeThread - Prime: 7 PalindromeThread - Palindrome: 151 PrimeThread - Prime: 11 PalindromeThread - Palindrome: 161 PrimeThread - Prime: 13 PalindromeThread - Palindrome: 171 PrimeThread - Prime: 17 PalindromeThread - Palindrome: 181 PrimeThread - Prime: 19 PalindromeThread - Palindrome: 191 PrimeThread - Prime: 23 PalindromeThread - Palindrome: 202 PrimeThread - Prime: 29 PalindromeThread - Palindrome: 212 PrimeThread - Prime: 31 PalindromeThread - Palindrome: 222 PrimeThread - Prime: 37 PalindromeThread - Palindrome: 232 PrimeThread - Prime: 41 PalindromeThread - Palindrome: 242 PrimeThread - Prime: 43 PalindromeThread - Palindrome: 252 PrimeThread - Prime: 47 PalindromeThread - Palindrome: 262 PrimeThread - Prime: 53

PrimeThread - Prime: 59 PalindromeThread - Palindrome: 282 PrimeThread - Prime: 61 PalindromeThread - Palindrome: 292 PrimeThread - Prime: 67 PalindromeThread - Palindrome: 303 PrimeThread - Prime: 71 PalindromeThread - Palindrome: 313 PrimeThread - Prime: 73 PrimeThread - Prime: 79 PalindromeThread - Palindrome: 323 PrimeThread - Prime: 83 PalindromeThread - Palindrome: 333 PrimeThread - Prime: 89 PalindromeThread - Palindrome: 343 PrimeThread - Prime: 97 PalindromeThread - Palindrome: 353 PalindromeThread - Palindrome: 363 PalindromeThread - Palindrome: 373 PalindromeThread - Palindrome: 383 PalindromeThread - Palindrome: 393 PalindromeThread - Palindrome: 404 PalindromeThread - Palindrome: 414 PalindromeThread - Palindrome: 424 PalindromeThread - Palindrome: 434 PalindromeThread - Palindrome: 444 PalindromeThread - Palindrome: 454 PalindromeThread - Palindrome: 464 PalindromeThread - Palindrome: 474 PalindromeThread - Palindrome: 484 PalindromeThread - Palindrome: 494 PalindromeThread - Palindrome: 505 PalindromeThread - Palindrome: 515 PalindromeThread - Palindrome: 525 PalindromeThread - Palindrome: 535 PalindromeThread - Palindrome: 545 PalindromeThread - Palindrome: 555 PalindromeThread - Palindrome: 565 PalindromeThread - Palindrome: 575 PalindromeThread - Palindrome: 585 PalindromeThread - Palindrome: 595 PalindromeThread - Palindrome: 606 PalindromeThread - Palindrome: 616 PalindromeThread - Palindrome: 626 PalindromeThread - Palindrome: 636

PalindromeThread - Palindrome: 656 PalindromeThread - Palindrome: 666 PalindromeThread - Palindrome: 676 PalindromeThread - Palindrome: 686 PalindromeThread - Palindrome: 696 PalindromeThread - Palindrome: 707 PalindromeThread - Palindrome: 717 PalindromeThread - Palindrome: 727 PalindromeThread - Palindrome: 737 PalindromeThread - Palindrome: 747 PalindromeThread - Palindrome: 757 PalindromeThread - Palindrome: 767 PalindromeThread - Palindrome: 777 PalindromeThread - Palindrome: 787 PalindromeThread - Palindrome: 797 PalindromeThread - Palindrome: 808 PalindromeThread - Palindrome: 818 PalindromeThread - Palindrome: 828 PalindromeThread - Palindrome: 838 PalindromeThread - Palindrome: 848 PalindromeThread - Palindrome: 858 PalindromeThread - Palindrome: 868 PalindromeThread - Palindrome: 878 PalindromeThread - Palindrome: 888 PalindromeThread - Palindrome: 898 PalindromeThread - Palindrome: 909 PalindromeThread - Palindrome: 919 PalindromeThread - Palindrome: 929 PalindromeThread - Palindrome: 939 PalindromeThread - Palindrome: 949 PalindromeThread - Palindrome: 959 PalindromeThread - Palindrome: 969 PalindromeThread - Palindrome: 979 PalindromeThread - Palindrome: 989 PalindromeThread - Palindrome: 999

**Problem Statement:** Write a Java program to create two threads t1 and main thread. Demonstrate the working of join and sleep method. Main thread will wait till t1 completes its execution.

**Objective:** To create a Java application that demonstrates the usage of join() and sleep() methods by creating a separate thread (t1) that performs a counting task with delays. The main thread waits for t1 to complete its execution using the join() method before resuming its own execution.

## Code:

class MyThread extends Thread { public void run() {

for(int i=1; i<=10; i++) { try {

Thread.sleep(1000);

}

catch(InterruptedException e) { e.getMessage();

}

System.out.println("Child t1 Thread");

}

}

}

class SleepJoinThread {

public static void main(String[] args) throws Exception { MyThread t1=new MyThread();

t1.start();

t1.join();

for(int i=1; i<=10; i++) { System.out.println("Main Thread");

}

}

}

# Output:

Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread

**Problem Statement:** Write a Java program to create two threads t1 and main thread. Demonstrate the working of join and sleep method. t1 thread will wait till main completes its execution.

**Objective:** To develop a Java application that demonstrates the use of join() and sleep() methods by creating two threads: the main thread and a child thread (t1). In this program, the t1 thread waits for the main thread to complete its execution using the join() method, while the main thread performs a task with simulated delays using the sleep() method.

## Code:

class MyThread extends Thread { Thread mainThread; MyThread(Thread mainThread) {

this.mainThread=mainThread;

}

public void run() { try {

for(int i=1; i<=10; i++) { mainThread.join(); System.out.println("Child t1 Thread");

}

}

catch(InterruptedException e) { e.getMessage();

}

}

}

class MainThreadSleep {

public static void main(String[] args) throws Exception { Thread mainThread=Thread.currentThread(); MyThread t1=new MyThread(mainThread);

t1.start();

for(int i=1; i<=10; i++) { System.out.println("Main Thread"); Thread.sleep(1000);

}

}

}

# Output:

Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread Main Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread Child t1 Thread