**1.   Write a Java program to perform a runnable interface, take two threads t1 and t2 and fetch the names of the thread using getName() method.**

**CODE:**

import java.util.\*;

class MyRunnable implements Runnable {

public void run() {

System.out.println("Thread Name: " + Thread.currentThread().getName());

}

}

public class Main {

public static void main(String[] args) {

MyRunnable myRunnable = new MyRunnable();

Thread t1 = new Thread(myRunnable);

Thread t2 = new Thread(myRunnable);

t1.start();

t2.start();

System.out.println("Thread t1 name: " + t1.getName());

System.out.println("Thread t2 name: " + t2.getName());

}

}

**2.Given an integer N, the task is to write program to print the first N natural numbers in increasing order using two threads.**

***Input: N = 10  
Output: 1 2 3 4 5 6 7 8 9 10***

***Input: N = 18  
Output: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18***

**CODE:**

import java.util.Scanner;

class PrintNumbers implements Runnable {

private int start;

private int end;

public PrintNumbers(int start, int end) {

this.start = start;

this.end = end;

}

public void run() {

for (int i = start; i <= end; i++) {

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

}

}

}

public class Main{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the value of N: ");

int N = scanner.nextInt();

int mid = N / 2;

PrintNumbers printFirstHalf = new PrintNumbers(1, mid);

PrintNumbers printSecondHalf = new PrintNumbers(mid + 1, N);

Thread t1 = new Thread(printFirstHalf, "Thread 1");

Thread t2 = new Thread(printSecondHalf, "Thread 2");

t1.start();

t2.start();

try {

t1.join();

t2.join();

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

**3.  Write a two-threaded program, where one thread finds all prime numbers (in 0 to 10) and another thread finds all palindrome numbers (in 10 to 50). Schedule these threads in a sequential manner to get the results.**

**Palindrome numbers from 10 to 50 : 11 22 33 44**

**Prime numbers from 0 to 10 : 2 3 5 7**

**CODE:**

import java.util.\*;

class PrimeNumberFinder implements Runnable {

public void run() {

System.out.println("Thread 1 - Finding Prime Numbers (0 to 10):");

for (int i = 0; i <= 10; i++) {

if (isPrime(i)) {

System.out.println("Prime Number: " + i);

}

}

}

private 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;

}

}

class PalindromeNumberFinder implements Runnable {

public void run() {

System.out.println("Thread 2 - Finding Palindrome Numbers (10 to 50):");

for (int i = 10; i <= 50; i++) {

if (isPalindrome(i)) {

System.out.println("Palindrome Number: " + i);

}

}

}

private boolean isPalindrome(int num) {

int originalNum = num;

int reversedNum = 0;

while (num > 0) {

int digit = num % 10;

reversedNum = reversedNum \* 10 + digit;

num /= 10;

}

return originalNum == reversedNum;

}

}

public class Main {

public static void main(String[] args) {

PrimeNumberFinder primeNumberFinder = new PrimeNumberFinder();

PalindromeNumberFinder palindromeNumberFinder = new PalindromeNumberFinder();

Thread t1 = new Thread(primeNumberFinder);

Thread t2 = new Thread(palindromeNumberFinder);

t1.start();

try {

t1.join();

} catch (InterruptedException e) {

e.printStackTrace();

}

t2.start();

}

}