1. class MyRunnable implements Runnable {

public void run() {

// Code to be executed by the thread

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

}

}

public class ThreadExample {

public static void main(String[] args) {

// Creating instances of the MyRunnable class

MyRunnable myRunnable = new MyRunnable();

// Creating threads and associating with the MyRunnable instances

Thread t1 = new Thread(myRunnable, "Thread-1");

Thread t2 = new Thread(myRunnable, "Thread-2");

// Starting the threads

t1.start();

t2.start();

}

}

2)

class PrintNumbers {

private int n;

private volatile int currentNumber;

public PrintNumbers(int n) {

this.n = n;

this.currentNumber = 1;

}

public synchronized void printOdd() {

while (currentNumber <= n) {

if (currentNumber % 2 != 0) {

System.out.print(currentNumber + " ");

currentNumber++;

notify();

} else {

try {

wait();

} catch (InterruptedException e) {

Thread.currentThread().interrupt();

}

}

}

}

public synchronized void printEven() {

while (currentNumber <= n) {

if (currentNumber % 2 == 0) {

System.out.print(currentNumber + " ");

currentNumber++;

notify();

} else {

try {

wait();

} catch (InterruptedException e) {

Thread.currentThread().interrupt();

}

}

}

}

}

public class Main {

public static void main(String[] args) {

int n = 18;

PrintNumbers printNumbers = new PrintNumbers(n);

Thread oddThread = new Thread(() -> printNumbers.printOdd());

Thread evenThread = new Thread(() -> printNumbers.printEven());

oddThread.start();

evenThread.start();

}

}

3)

class PrimeThread extends Thread {

public void run() {

System.out.print("Prime numbers from 0 to 10: ");

for (int i = 2; i <= 10; i++) {

if (isPrime(i)) {

System.out.print(i + " ");

}

}

System.out.println();

}

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 PalindromeThread extends Thread {

public void run() {

System.out.print("Palindrome numbers from 10 to 50: ");

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

if (isPalindrome(i)) {

System.out.print(i + " ");

}

}

System.out.println();

}

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) {

PrimeThread primeThread = new PrimeThread();

PalindromeThread palindromeThread = new PalindromeThread();

primeThread.start();

try {

primeThread.join(); // Wait for primeThread to finish before starting palindromeThread

} catch (InterruptedException e) {

e.printStackTrace();

}

palindromeThread.start();

}

}