ASSIGNMENT JAVA DAY15

Harshit Kushmakar| 16896

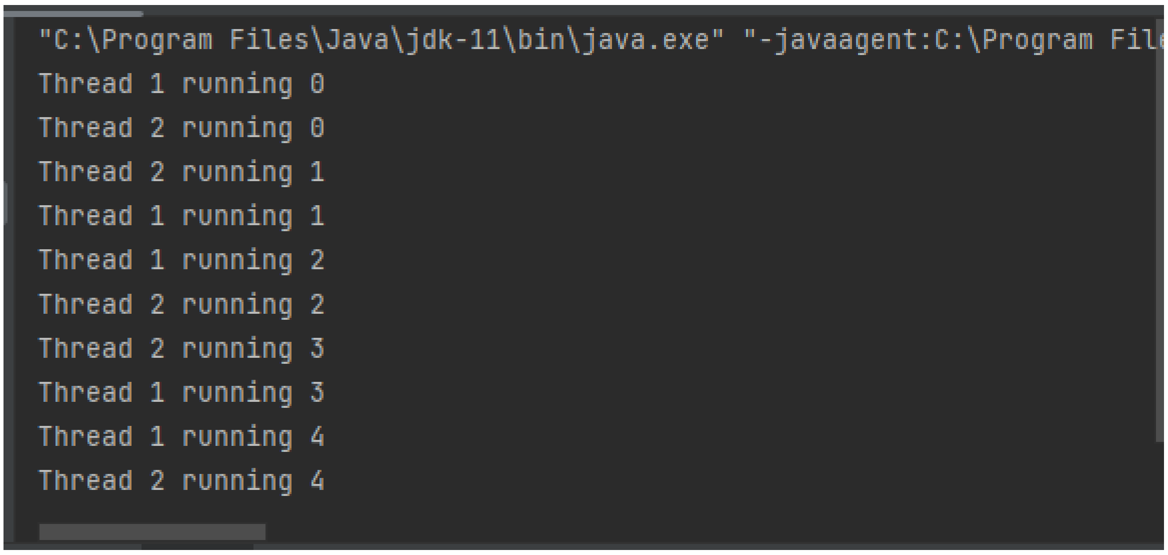
1. Create two threads in java. One using thread class and other using Runnable interface.

```
package assignment15;

public class MyRunnable implements Runnable{
    public void run() {
        for (int i = 0; i < 5; i++) {
            System.out.println(Thread.currentThread().getName() + " running
" + i);
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```
package assignment15;

public class MyThread extends Thread {
    public MyThread(String name) {
        super(name);
    }

    public void run() {
        for (int i = 0; i < 5; i++) {
            System.out.println(getName() + " running " + i);
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```
package assignment15;

public class Main {
    public static void main(String[] args) {
        // Creating a Thread using Thread class
        Thread t1 = new MyThread("Thread 1");
        t1.start();
        // Creating a Thread using Runnable interface
        Runnable r = new MyRunnable();
        Thread t2 = new Thread(r, "Thread 2");
        t2.start();
    }
}
```

OUTPUT:



```
"C:\Program Files\Java\jdk-11\bin\java.exe" "-javaagent:C:\Program Fil
Thread 1 running 0
Thread 2 running 0
Thread 2 running 1
Thread 1 running 1
Thread 1 running 2
Thread 2 running 2
Thread 2 running 3
Thread 1 running 3
Thread 1 running 4
Thread 2 running 4
```

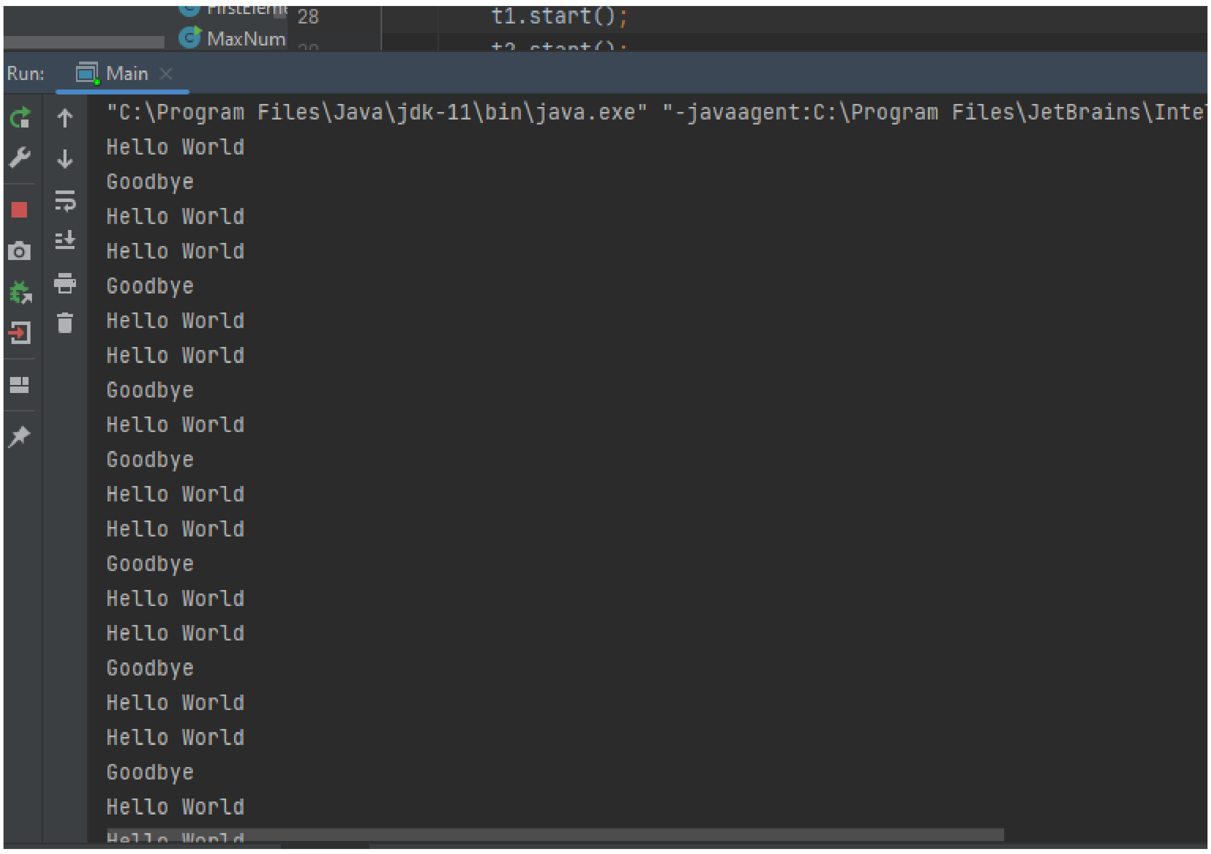
2. Create two threads, one will print “hello world” every second and other will print “Bye” every 2 seconds.

```
package assignment15;

public class GoodByeThread {
    class GoodbyeThread implements Runnable {
        public void run() {
            while (true) {
                System.out.println("Goodbye");
                try {
                    Thread.sleep(2000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }
    }
}
```

```
package assignment15;

public class HelloWorldThread implements Runnable {
    public void run() {
```

```java
        while (true) {
            System.out.println("Hello World");
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```java
package assignment15;

import assignment15.GoodByeThread;

public class Main {
    public static void main(String[] args) {
        // Creating two threads
        Thread t1 = new Thread(new HelloWorldThread());
        Thread t2 = new Thread(new GoodByeThread.GoodbyeThread());

        // Starting the threads
        t1.start();
        t2.start();
    }
}
```



```
t1.start();
```

```
"C:\Program Files\Java\jdk-11\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\Inte
Hello World
Goodbye
Hello World
Hello World
Goodbye
Hello World
Hello World
Goodbye
Hello World
Goodbye
Hello World
Hello World
Goodbye
Hello World
Hello World
Goodbye
Hello World
Hello World
Goodbye
Hello World
```

3. Write a Java program to create 5 threads to print 1-10 in a loop in every thread. The output must look like as displayed below:

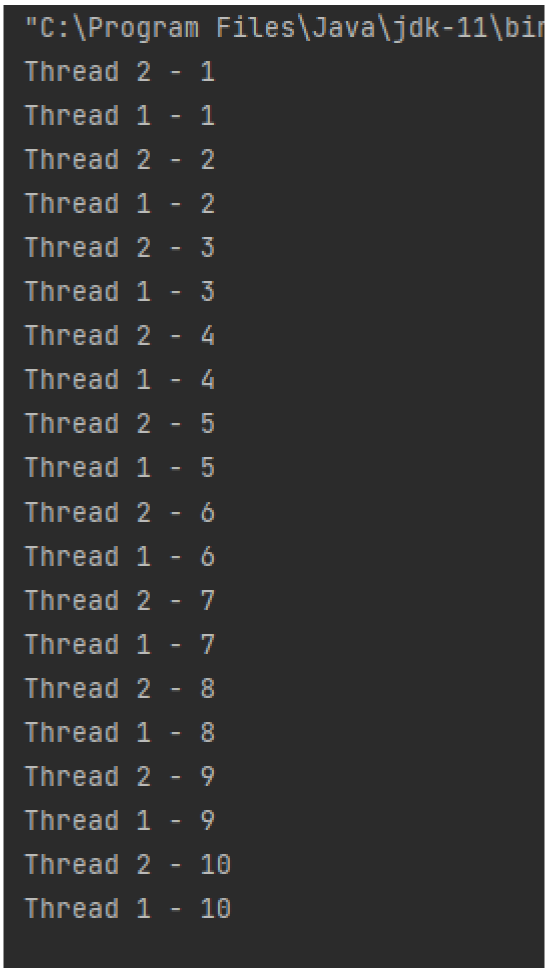
<Thead Name> – 1

<Thread Name> - 2 and so on

Each thread will print in the same manner. Check the output sequence. Is it in sequence or random order. Record your findings with the reason in form of comments.

```java
package assignment15;

public class NumberPrinter implements Runnable{

    private int threadNumber;

    public NumberPrinter(int threadNumber) {
        this.threadNumber = threadNumber;
    }

    public void run() {
        for (int i = 1; i <= 10; i++) {
            System.out.println("Thread " + threadNumber + " - " + i);
        }
    }
}
```

```java
package assignment15;

public class Main {
    public static void main(String[] args) {
        // Create thread 1 with highest priority
        Thread t1 = new Thread( new NumberPrinter(1));
        t1.setPriority(Thread.MAX_PRIORITY);
        t1.start();

        // Create thread 2 with normal priority
        Thread t2 = new Thread( new NumberPrinter(2));
        t2.start();
    }
}
```



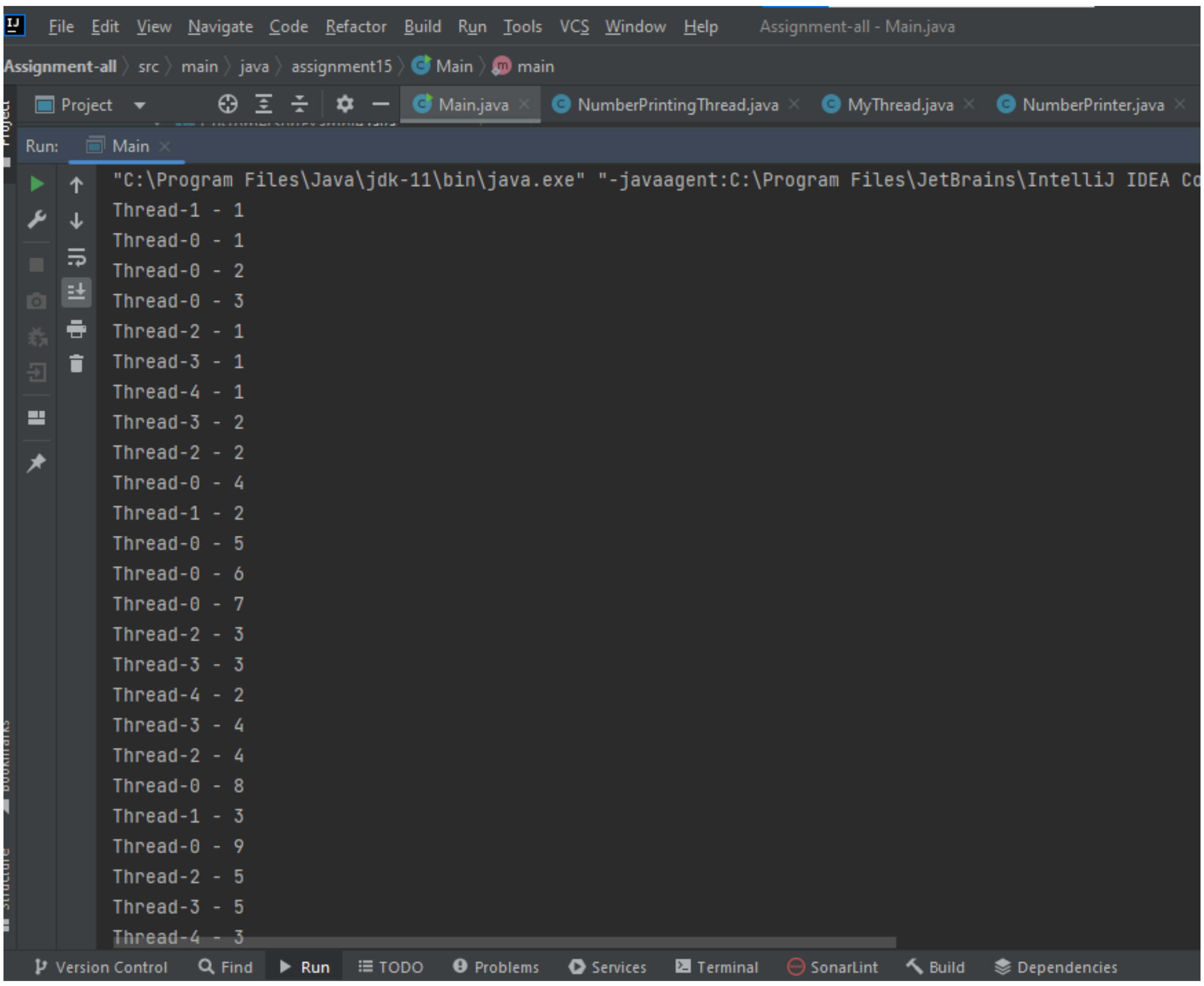
```
"C:\Program Files\Java\jdk-11\bi
Thread 2 - 1
Thread 1 - 1
Thread 2 - 2
Thread 1 - 2
Thread 2 - 3
Thread 1 - 3
Thread 2 - 4
Thread 1 - 4
Thread 2 - 5
Thread 1 - 5
Thread 2 - 6
Thread 1 - 6
Thread 2 - 7
Thread 1 - 7
Thread 2 - 8
Thread 1 - 8
Thread 2 - 9
Thread 1 - 9
Thread 2 - 10
Thread 1 - 10
```

4. In the above question, set the priority of Thread 1 to be highest. See the impact on the output.

```
package assignment15;
public class Main {
    public static void main(String[] args) {
        Thread thread1 = new NumberPrintingThread(1);
        Thread thread2 = new NumberPrintingThread(2);
        Thread thread3 = new NumberPrintingThread(3);
        Thread thread4 = new NumberPrintingThread(4);
        Thread thread5 = new NumberPrintingThread(5);

        // Set the priority of the first thread to the highest
        thread1.setPriority(Thread.MAX_PRIORITY);

        thread1.start();
        thread2.start();
        thread3.start();
        thread4.start();
        thread5.start();
    }
}
```

```
package assignment15;
```

```java
public class NumberPrintingThread extends Thread{

    private int threadNumber;

    public NumberPrintingThread(int threadNumber) {
        this.threadNumber = threadNumber;
    }

    public void run() {
        for (int i = 1; i <= 10; i++) {
            System.out.println(Thread.currentThread().getName() + " - " +
i);
        }
    }
}
```



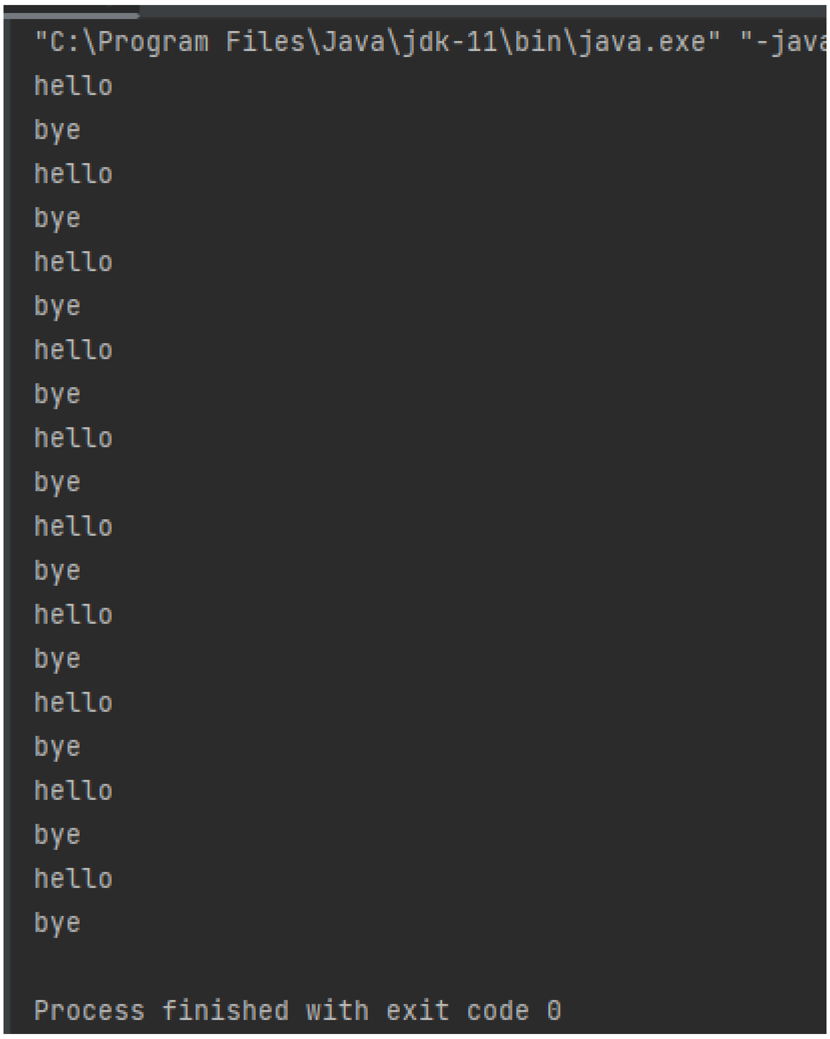
File Edit View Navigate Code Refactor Build Run Tools VCS Window Help Assignment-all - Main.java
Assignment-all src main java assignment15 Main main
Project Main.java NumberPrintingThread.java MyThread.java NumberPrinter.java
Run: Main
Version Control Find Run TODO Problems Services Terminal SonarLint Build Dependencies

```
"C:\Program Files\Java\jdk-11\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Co
Thread-1 - 1
Thread-0 - 1
Thread-0 - 2
Thread-0 - 3
Thread-2 - 1
Thread-3 - 1
Thread-4 - 1
Thread-3 - 2
Thread-2 - 2
Thread-0 - 4
Thread-1 - 2
Thread-0 - 5
Thread-0 - 6
Thread-0 - 7
Thread-2 - 3
Thread-3 - 3
Thread-4 - 2
Thread-3 - 4
Thread-2 - 4
Thread-0 - 8
Thread-1 - 3
Thread-0 - 9
Thread-2 - 5
Thread-3 - 5
Thread-4 - 3
```

5. Create two threads, one will print “hello” and other will print Bye for 10 times. The order of printing the message should be one after another i.e “hello” “bye” “hello” “bye”

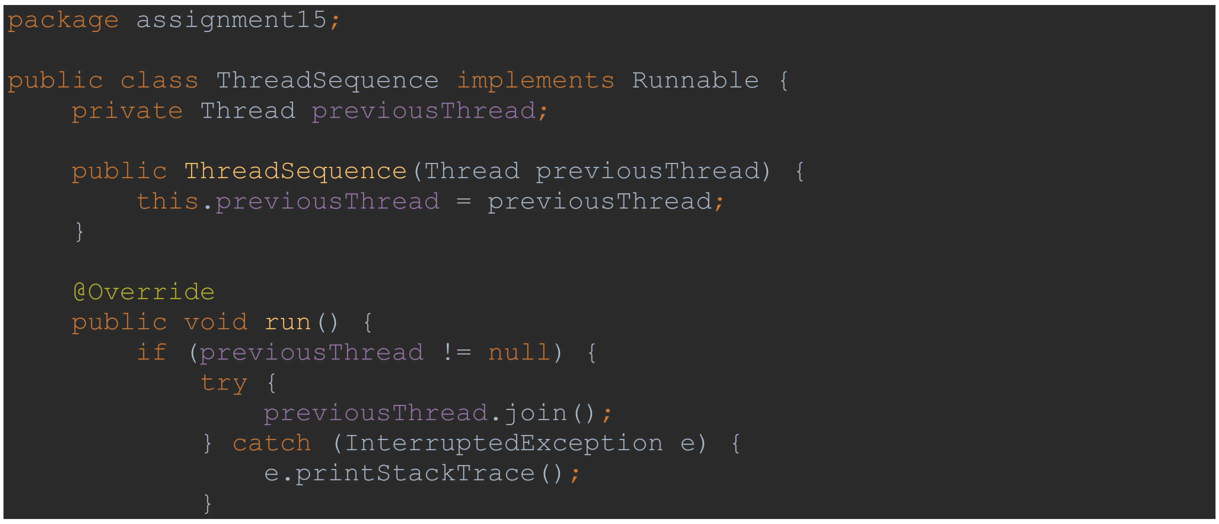
```java
package assignment15;

public class HelloByeThread extends Thread {
    private String message;
    private Object lock;
    private int count;

    public HelloByeThread(String message, Object lock, int count) {
        this.message = message;
        this.lock = lock;
        this.count = count;
    }

    public void run() {
        for (int i = 0; i < count; i++) {
            synchronized (lock) {
                System.out.println(message);
                lock.notify();
                try {
                    if (i < count - 1) {
                        lock.wait();
                    }
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }
    }
```

```java
package assignment15;

public class Main {
 public static void main(String[] args) {
     Object lock = new Object();
     Thread helloThread = new HelloByeThread("hello", lock, 10);
     Thread byeThread = new HelloByeThread("bye", lock, 10);

     helloThread.start();
     byeThread.start();
 }
}
```

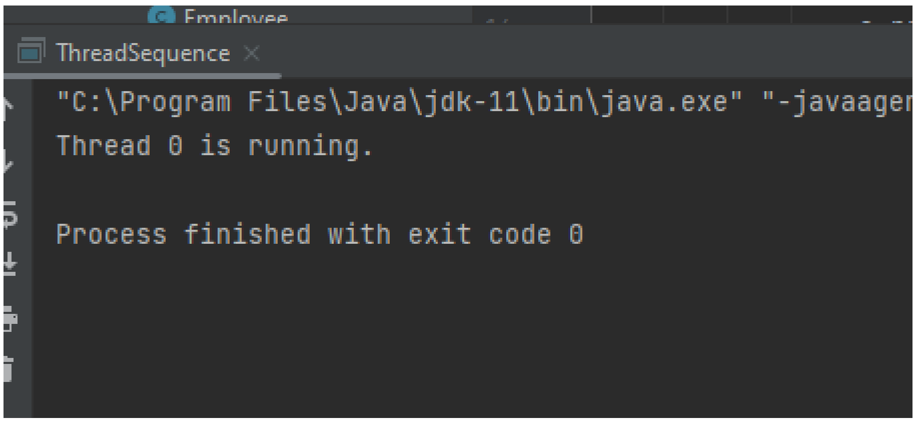


```
"C:\Program Files\Java\jdk-11\bin\java.exe" "-java
hello
bye
hello
bye
hello
bye
hello
bye
hello
bye
hello
bye
hello
bye
hello
bye
hello
bye
hello
bye

Process finished with exit code 0
```

6. Write a java program that creates a number of threads and each thread must start after thecompletion of previous thread except the first one.



```java
package assignment15;

public class ThreadSequence implements Runnable {
    private Thread previousThread;

    public ThreadSequence(Thread previousThread) {
        this.previousThread = previousThread;
    }

    @Override
    public void run() {
        if (previousThread != null) {
            try {
                previousThread.join();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        System.out.println(Thread.currentThread().getName() + " is 
running.");
    }

    public static void main(String[] args) {
        int numThreads = 5;
        Thread[] threads = new Thread[numThreads];

        for (int i = 0; i < numThreads; i++) {
            ThreadSequence runnable = new ThreadSequence(i == 0 ? null : 
threads[i-1]);
            threads[i] = new Thread(runnable, "Thread " + i);
        }

        threads[0].start();
    }
}
```

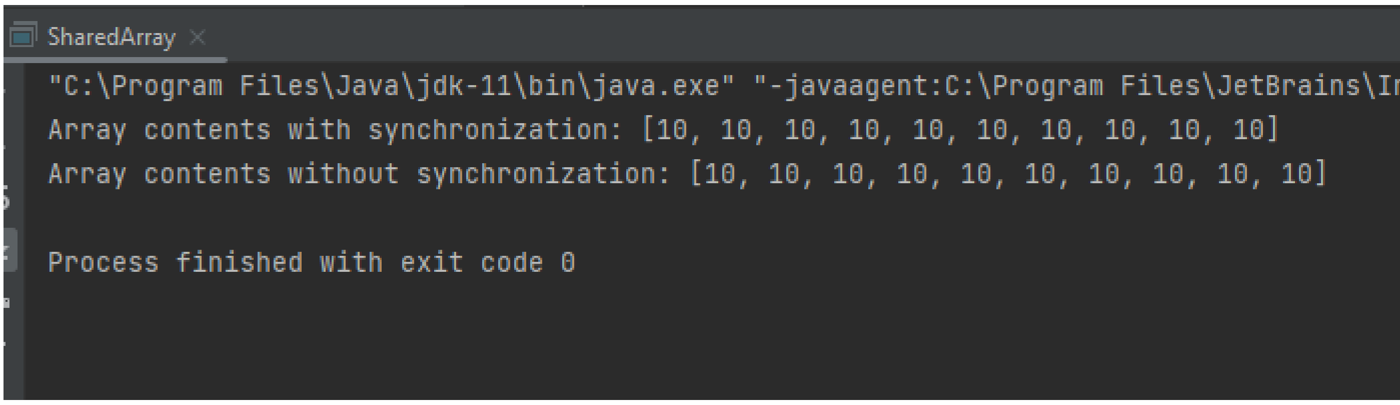


```
"C:\Program Files\Java\jdk-11\bin\java.exe" "-javaager
Thread 0 is running.

Process finished with exit code 0
```

7. Write a java program taking array as a shared resource and which is access by multiple threads with and without synchronization it.

```
package assignment15;

public class ThreadSequence implements Runnable {
    private Thread previousThread;

    public ThreadSequence(Thread previousThread) {
        this.previousThread = previousThread;
    }

    @Override
    public void run() {
        if (previousThread != null) {
            try {
                previousThread.join();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        System.out.println(Thread.currentThread().getName() + " is 
running.");
```

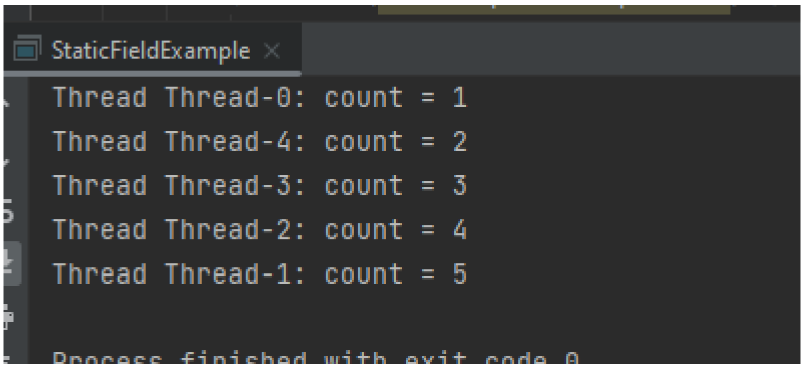
```
    }

    public static void main(String[] args) {
        int numThreads = 5;
        Thread[] threads = new Thread[numThreads];

        for (int i = 0; i < numThreads; i++) {
            ThreadSequence runnable = new ThreadSequence(i == 0 ? null :
threads[i-1]);
            threads[i] = new Thread(runnable, "Thread " + i);
        }

        threads[0].start();
    }
}
```


SharedArray

```
"C:\Program Files\Java\jdk-11\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\I
Array contents with synchronization: [10, 10, 10, 10, 10, 10, 10, 10, 10, 10]
Array contents without synchronization: [10, 10, 10, 10, 10, 10, 10, 10, 10, 10]

Process finished with exit code 0
```

8. Write a java program that creates a number of threads which access the static field and prints the incremented value of static field.

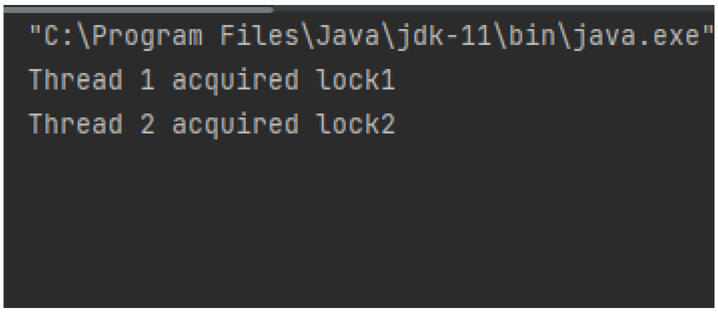
```
package assignment15;

public class StaticFieldExample {

    private static int count = 0;

    public static void main(String[] args) {
        int numThreads = 5;
        Thread[] threads = new Thread[numThreads];

        for (int i = 0; i < numThreads; i++) {
            threads[i] = new Thread(() -> {
                synchronized (StaticFieldExample.class) {
                    System.out.println("Thread " +
Thread.currentThread().getName() + ": count = " + ++count);
                }
            });
            threads[i].start();
        }

        for (Thread thread : threads) {
            try {
                thread.join();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}
```



```
Thread Thread-0: count = 1
Thread Thread-4: count = 2
Thread Thread-3: count = 3
Thread Thread-2: count = 4
Thread Thread-1: count = 5

Process finished with exit code 0
```

9. Write a java program to present a deadlock condition.

```
package assignment15;

public class DeadLockExample {



        private static final Object lock1 = new Object();
        private static final Object lock2 = new Object();

        public static void main(String[] args) {
            Thread t1 = new Thread(() -> {
                synchronized (lock1) {
                    System.out.println("Thread 1 acquired lock1");
                    try {
                        Thread.sleep(1000);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                    synchronized (lock2) {
                        System.out.println("Thread 1 acquired lock2");
                    }
                }
            });

            Thread t2 = new Thread(() -> {
                synchronized (lock2) {
                    System.out.println("Thread 2 acquired lock2");
                    try {
                        Thread.sleep(1000);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                    synchronized (lock1) {
                        System.out.println("Thread 2 acquired lock1");
                    }
                }
            });

            t1.start();
            t2.start();
        }
    }
```



```
"C:\Program Files\Java\jdk-11\bin\java.exe"
Thread 1 acquired lock1
Thread 2 acquired lock2
```

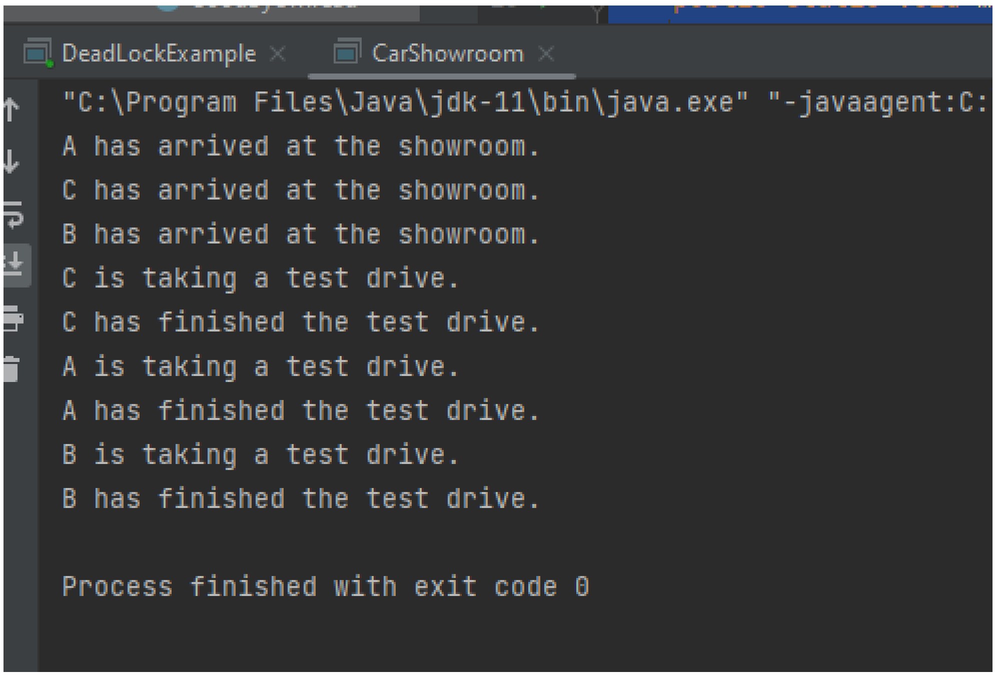
10. Write a java program with two threads in which each thread reads the data from a text file and display the data of each file on Console alternatively such that one line from first input file is printed and then one line from another input file is printed. The names of two input file to be taken from user.

11. 3 customers (A, B, C) visit car showroom. The visitors take a test drive for random time and the other visitors have to wait till a car becomes free. i.e if “A” is taking the drive B and C have to wait. Implement this functionality using threads in java

```
package assignment15;

import java.util.concurrent.Semaphore;

public class CarShowroom {

    private static final Semaphore sem = new Semaphore(1);
    private static int carsAvailable = 1;

    public static void main(String[] args) {
        // Create three customers
```

```java
        Thread a = new Thread(new Customer("A"));
        Thread b = new Thread(new Customer("B"));
        Thread c = new Thread(new Customer("C"));

        // Start the customers
        a.start();
        b.start();
        c.start();
    }

    static class Customer implements Runnable {
        private final String name;

        public Customer(String name) {
            this.name = name;
        }

        @Override
        public void run() {
            System.out.println(name + " has arrived at the showroom.");

            try {
                sem.acquire();
                System.out.println(name + " is taking a test drive.");

                // Simulate the test drive taking a random amount of time
                Thread.sleep((long) (Math.random() * 10000));

                System.out.println(name + " has finished the test drive.");
            } catch (InterruptedException e) {
                e.printStackTrace();
            } finally {
                sem.release();
            }
        }
    }
}
```

OUTPUT:



```
"C:\Program Files\Java\jdk-11\bin\java.exe" "-javaagent:C:
A has arrived at the showroom.
C has arrived at the showroom.
B has arrived at the showroom.
C is taking a test drive.
C has finished the test drive.
A is taking a test drive.
A has finished the test drive.
B is taking a test drive.
B has finished the test drive.

Process finished with exit code 0
```