

LAB # 6

Deadlock in concurrency

Lab Task:-

- Create three threads by implementing thread synchronization block through 3 locks. (Hint: Apply un-sequenced lock to analyze deadlock and solve it through provided solution:

Code:- (Deadlock Causes)

```

DeadLockTask.java ×
1  public class DeadLockTask {
2      private static final Object lockA = new Object();  2 usages
3      private static final Object lockB = new Object();  2 usages
4      private static final Object lockC = new Object();  2 usages
5      public static void main(String[] args) {
6          System.out.println("Name: Rayyan Wamiq | Roll no.: 2023F-BSE-095");
7          Thread t1 = new Thread(() -> {
8              synchronized (lockA) {
9                  System.out.println("T1 acquired Lock A");
10                 sleep(ms: 100);
11                 synchronized (lockB) {
12                     System.out.println("T1 acquired Lock B");
13                 }
14             });
15         Thread t2 = new Thread(() -> {
16             synchronized (lockB) {
17                 System.out.println("T2 acquired Lock B");
18                 sleep(ms: 100);
19                 synchronized (lockC) {
20                     System.out.println("T2 acquired Lock C");
21                 }
22             });
23         Thread t3 = new Thread(() -> {
24             synchronized (lockC) {
25                 System.out.println("T3 acquired Lock C");
26                 sleep(ms: 100);
27                 synchronized (lockA) {
28                     System.out.println("T3 acquired Lock A");
29                 }
30             });
31         });
32     });
33     t1.start();
34     t2.start();
35     t3.start();
36   }
37   private static void sleep(long ms) {  3 usages
38       try { Thread.sleep(ms); } 
39       catch (InterruptedException ignored) {}
40   }
41 }

```

Output:-

```
C:\Users\ LAPCOM\.jdks\openjdk-25\bin\java.exe
Name: Rayyan Wamiq | Roll no.: 2023F-BSE-095
T1 acquired Lock A
T2 acquired Lock B
T3 acquired Lock C
```

Code:- (Deadlock Solution)

```
DeadlockSolved.java
1  public class DeadlockSolved {
2      private static final Object lockA = new Object(); 1 usage
3      private static final Object lockB = new Object(); 1 usage
4      private static final Object lockC = new Object(); 1 usage
5  public static void main(String[] args) {
6      System.out.println("Name: Rayyan Wamiq | Roll no.: 2023F-BSE-095");
7      Thread t1 = new Thread(() -> acquireLocksInOrder(threadName: "T1"));
8      Thread t2 = new Thread(() -> acquireLocksInOrder(threadName: "T2"));
9      Thread t3 = new Thread(() -> acquireLocksInOrder(threadName: "T3"));
10     t1.start();
11     t2.start();
12     t3.start();
13 }
14 private static void acquireLocksInOrder(String threadName) { 3 usages
15     synchronized (lockA) {
16         System.out.println(threadName + " acquired Lock A");
17         sleep(ms: 50);
18         synchronized (lockB) {
19             System.out.println(threadName + " acquired Lock B");
20             sleep(ms: 50);
21
22             synchronized (lockC) {
23                 System.out.println(threadName + " acquired Lock C");
24             }
25         }
26     }
27     private static void sleep(long ms) { 2 usages
28         try { Thread.sleep(ms); }
29         catch (InterruptedException ignored) {}}
30 }
```

Output:-

```
C:\Users\ LAPCOM\.jdks\openjdk-25\bin\java.exe
Name: Rayyan Wamiq | Roll no.: 2023F-BSE-095
T1 acquired Lock A
T1 acquired Lock B
T1 acquired Lock C
T2 acquired Lock A
T2 acquired Lock B
T2 acquired Lock C
T3 acquired Lock A
T3 acquired Lock B
T3 acquired Lock C

Process finished with exit code 0
```