Unit 6: Multithreading

1. Thread Lifecycle Example (Simple state print)

```
java
CopyEdit
public class ThreadLifecycleExample extends Thread {
    public void run() {
        System.out.println("Thread is running.");
    }

    public static void main(String[] args) {
        ThreadLifecycleExample t1 = new ThreadLifecycleExample();
        System.out.println("Thread state before start: " + t1.getState());
// NEW
        t1.start();
        System.out.println("Thread state after start: " + t1.getState());
// RUNNABLE or TERMINATED depending on timing
    }
}
```

// Shows NEW state before start, RUNNABLE after start.

2. Creating Multiple Threads Example

```
java
CopyEdit
class MyThread extends Thread {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println(getName() + " is running: " + i);
            try {
                Thread.sleep(500);  // Sleep for 0.5 seconds
            } catch (InterruptedException e) {
                System.out.println("Thread interrupted");
            }
        }
    }
}

public class MultipleThreadsExample {
    public static void main(String[] args) {
        MyThread t1 = new MyThread();
        MyThread t2 = new MyThread();

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

// Creates and runs two threads concurrently.

3. Thread Priorities Example

```
java
CopyEdit
public class ThreadPriorityExample extends Thread {
    public void run() {
        System.out.println(getName() + " with priority " + getPriority() +
" is running.");
    }

    public static void main(String[] args) {
        ThreadPriorityExample t1 = new ThreadPriorityExample();
        ThreadPriorityExample t2 = new ThreadPriorityExample();

        t1.setName("Thread 1");
        t2.setName("Thread 2");

        t1.setPriority(Thread.MIN_PRIORITY);  // Priority 1 (lowest)
        t2.setPriority(Thread.MAX_PRIORITY);  // Priority 10 (highest)

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

// Thread priorities hint scheduler which thread runs first.

---

4. Synchronization Example (to avoid race condition)

```
java
CopyEdit
class Counter {
    private int count = 0;

    // synchronized to prevent race condition
    public synchronized void increment() {
        count++;
    }

    public int getCount() {
        return count;
    }
}

public class SynchronizationExample {
    public static void main(String[] args) throws InterruptedException {
        Counter counter = new Counter();

        Runnable task = () -> {
            for (int i = 0; i < 1000; i++) {
                counter.increment();
            }
        };
        Thread t1 = new Thread(task);
        Thread t2 = new Thread(task);

        t1.start();
        t2.start();

        t1.join();
        t2.join();

        System.out.println("Final count: " + counter.getCount()); // Should
be 2000
    }
}
```

// `synchronized` ensures one thread increments at a time, avoiding incorrect counts.