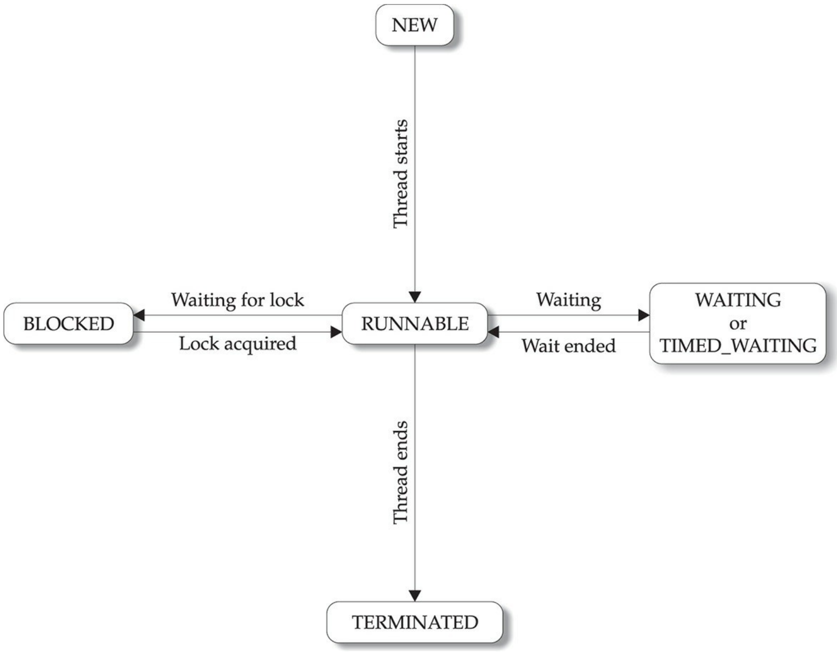
Lab Program 12. : Write a Java program to illustrate five states of the life cycle of the thread (New, Runnable, Running, Non-Runnable (Blocked/Waiting), Terminated) with an example program.



A thread can be in one of the five states.The life cycle of the thread in java is controlled by JVM. The java thread states are as follows:

1. New
2. Runnable
3. Running
4. Non-Runnable (Blocked/Waiting)
5. Terminated

Java Code

```
package Threads;


class ThreadStateExample extends Thread {

    @Override
    public void run() {
        // Thread is in running state when it enters here.
        System.out.println("Thread is running.");

        try {
            // Making the thread sleep to put it in Non-Runnable (Waiting) state.
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted.");
        }

        System.out.println("Thread is finished running.");
    }

    public static void main(String[] args) {
        // Thread is in New state when created but not started.
        ThreadStateExample thread = new ThreadStateExample();
        System.out.println("Thread state after creation: " + thread.getState());  // NEW

        // Thread is now in Runnable state after start() method is invoked.
        thread.start();
        System.out.println("Thread state after calling start: " + thread.getState());  // RUNNABLE

        try {
            // The main thread sleeps so we can see the RUNNING and WAITING state of the thread.
            Thread.sleep(500);  // Allows time for the thread to run
            System.out.println("Thread state while sleeping: " + thread.getState());  // TIMED_WAITING
(non-runnable state)

            // Wait for thread to die (terminated state).
            thread.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        // Thread is in Terminated state after finishing execution.
        System.out.println("Thread state after execution: " + thread.getState());  // TERMINATED
    }
}
```

o/p

```
Thread state after creation: NEW
Thread state after calling start: RUNNABLE
Thread is running.
Thread state while sleeping: TIMED_WAITING
Thread is finished running.
Thread state after execution: TERMINATED
```