Task 2: States and Transitions

Create a Java class that simulates a thread going through different lifecycle states: NEW, RUNNABLE, WAITING, TIMED_WAITING, BLOCKED, and TERMINATED. Use methods like sleep(), wait(), notify(), and join() to demonstrate these states..

```
class StateAndTransition implements Runnable {
    private final Object lock = new Object();

    @Override
    public void run() {
        System.out.println(Thread.currentThread().getName() + ": State RUNNABLE");

        // Demonstrate TIMED_WAITING using sleep
        try {
            System.out.println(Thread.currentThread().getName() + ": State TIMED_WAITING (sleeping)");
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            Thread.currentThread().interrupt();
        }

        // Demonstrate WAITING using wait
        synchronized (lock) {
            try {
                System.out.println(Thread.currentThread().getName() + ": State WAITING (waiting for lock)");
                lock.wait();
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
        }

        // Continue running
        System.out.println(Thread.currentThread().getName() + ": State RUNNABLE again");

        // Demonstrate BLOCKED
        synchronized (lock) {
```

```
        System.out.println(Thread.currentThread().getName() + ": Acquired lock, not BLOCKED anymore");
    }

    System.out.println(Thread.currentThread().getName() + ": State TERMINATED (exiting run)");
}

public static void main(String[] args) {
    StateAndTransition lifecycle = new StateAndTransition();

    // State NEW
    Thread thread = new Thread(lifecycle, "Thread-1");
    System.out.println(thread.getName() + ": State NEW");

    // Start the thread
    thread.start();

    // Notify the waiting thread
    try {
        Thread.sleep(500); // Ensure the thread reaches the wait state
    } catch (InterruptedException e) {
        Thread.currentThread().interrupt();
    }

    synchronized (lifecycle.lock) {
        lifecycle.lock.notify();
        System.out.println("Main thread notified " + thread.getName());
    }

    // Wait for the thread to terminate
    try {
        thread.join();
    } catch (InterruptedException e) {
        Thread.currentThread().interrupt();
    }
```

```
        System.out.println(thread.getName() + ": State TERMINATED (confirmed in main)");
    }
}
```