

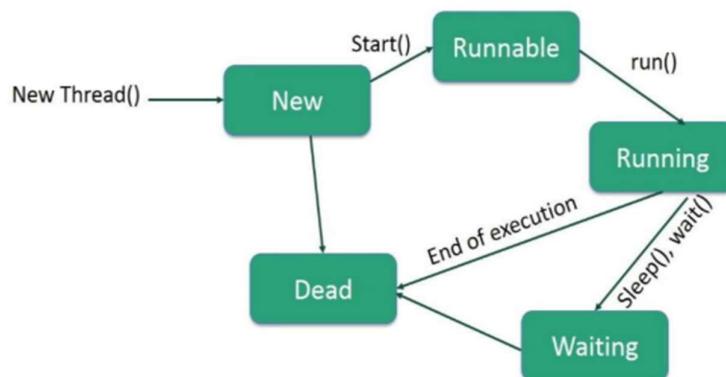
## Experiment 2

**Aim:** Write a program to implement the Multiple Thread application using java.

**Theory:** Java is a multi-threaded programming language which means we can develop multi-threaded programs using Java. A multi-threaded program contains two or more parts that can run concurrently and each part can handle a different task at the same time making optimal use of the available resources specially when your computer has multiple CPUs. By definition, multitasking is when multiple processes share common processing resources such as a CPU. Multi-threading extends the idea of multitasking into applications where you can subdivide specific operations within a single application into individual threads. Each of the threads can run in parallel. The OS divides processing time not only among different applications, but also among each thread within an application. Multi-threading enables you to write in a way where multiple activities can proceed concurrently in the same program.

### Life Cycle of a Thread

A thread goes through various stages in its life cycle. For example, a thread is born, started, runs, and then dies. The following diagram shows the complete life cycle of a thread.



Following are the stages of the life cycle –

- **New** – A new thread begins its life cycle in the new state. It remains in this state until the program starts the thread. It is also referred to as a born thread.
- **Runnable** – After a newly born thread is started, the thread becomes runnable. A thread in this state is considered to be executing its task.
- **Waiting** – Sometimes, a thread transitions to the waiting state while the thread waits for another thread to perform a task. A thread transitions back to the runnable state only when another thread signals the waiting thread to continue executing.
- **Timed Waiting** – A runnable thread can enter the timed waiting state for a specified interval of time. A thread in this state transitions back to the runnable state when that time interval expires or when the event it is waiting for occurs.
- **Terminated (Dead)** – A runnable thread enters the terminated state when it completes its task or otherwise terminates.

## Thread Priorities

Every Java thread has a priority that helps the operating system determine the order in which threads are scheduled.

Java thread priorities are in the range between `MIN_PRIORITY` (a constant of 1) and `MAX_PRIORITY` (a constant of 10). By default, every thread is given priority `NORM_PRIORITY` (a constant of 5).

Threads with higher priority are more important to a program and should be allocated processor time before lower-priority threads. However, thread priorities cannot guarantee the order in which threads execute and are very much platform dependent.

## Program

```
// Using Thread class
class MyThread extends Thread {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println(Thread.currentThread().getName() + " - Value: " + i);
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
    }
}
```

```
// Using Runnable interface
class MyRunnable implements Runnable {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println(Thread.currentThread().getName() + " - Value: " + i);
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
    }
}
```

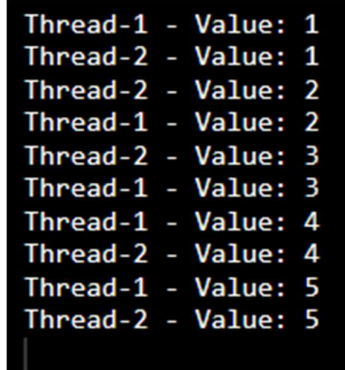
```
public class MultiThreadDemo {
    public static void main(String[] args) {
        MyThread thread1 = new MyThread();
```

```
Thread thread2 = new Thread(new MyRunnable());

thread1.setName("Thread-1");
thread2.setName("Thread-2");

thread1.start();
thread2.start();
}
}
```

### Output



```
Thread-1 - Value: 1
Thread-2 - Value: 1
Thread-2 - Value: 2
Thread-1 - Value: 2
Thread-2 - Value: 3
Thread-1 - Value: 3
Thread-1 - Value: 4
Thread-2 - Value: 4
Thread-1 - Value: 5
Thread-2 - Value: 5
|
```

### Conclusion:

Thus, we had Implemented the Multiple Thread application using java.