Experiment NO - 14

→ Aim → Aim of this program is to demonstrate the basic principles of multithreading in Java by creating multithreads that execute concurrently and implement it.

→ Resource Required → Notepad, JDK 1.8, wordpad, Pentium IV, Printer

→ Theory →

Multithreading is a programming concept that allows multiple threads to run concurrently within a single program.

1. Thread Class → Threads can be created by extending the thread class. The run() method contains the code that is executed when the thread starts.

2. Thread Lifecycle → A thread in Java goes through several states: New, Runnable, Blocked, Waiting, Timed waiting and Terminated. When a thread is created, it starts in the New state. Once the start() method is called, it enters the Runnable state, where the JVM (Java Virtual Machine) can

Schedule it for execution. Depending on the availability of system resources and scheduling, a thread may move between these states, allowing for efficient management of concurrent tasks.

SYNTAX for creating threads →

```
class MyThread extends Thread {
    public void run() {
        // Thread code goes here
    }
}
```

To start a thread, you call the start() method on an instance of the Thread class. This will invoke the run() method in a separate call stack.

SYNTAX →

```
MyThread thread = new MyThread();
thread.start(); // For extending Thread.
```

3. Concurrency → By running multiple threads, a program can perform several operations simultaneously, improving performance especially on multi-core processors. Concurrency is particularly beneficial for I/O-bound tasks, where threads can handle waiting operations while others continue executing.

4. Thread Identification - Each thread has a unique identifier (ID), which can be accessed using Thread.currentThread().getId(). This helps in add new lines in each point.

Conclusion - The program successfully demonstrates the creation and execution of multiple threads in Java. By creating eight instances of the MultithreadingDemo class and starting each thread, the program showcases the concurrent execution of the run() method. Each thread independently prints its ID highlighting the non blocking nature of multithreading.