OOAD

Lab 8 **-Adithya M, Section K**
-PES1UG20CS621

Multithreading:

- Multithreading is the ability of a program to execute multiple threads concurrently.
- A thread is a separate path of execution within a program, and each thread can execute its own set of instructions simultaneously with other threads.
- Multithreading allows a program to perform several tasks concurrently and can improve the performance of the program.
- A thread is started by creating an instance of the Thread class and passing the Runnable object to its constructor.
- A thread can be created by implementing the Runnable interface and overriding the run() method.
- The start() method of the Thread class is then called to start the thread.
- The join() method is used when you want one thread to wait for the completion of another thread.
- The interrupt() method of thread class is used to interrupt the thread. If any thread is in sleeping or waiting state (i.e. sleep() or wait() is invoked) then using the interrupt() method, we can interrupt the thread execution by throwing InterruptedException.

Code:

```
import java.util.*;
import java.net.*;

public class RaceSimulation implements Runnable {
    public int id;
    public int distance;

    public RaceSimulation(int id) {
        this.id = id;
        this.distance = 0;
    }

    @Override
    public void run() {
        Random random = new Random();
        while (distance < 1000) {
            int distanceCovered = random.nextInt() % 5 + 5;
            distance += distanceCovered;
            System.out.println(id + " " + distance + "m");
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        System.out.println(id + " has finished");
    }

    public static void main(String[] args) throws InterruptedException, UnknownHostException {
        System.out.println("User name is: " + System.getProperty("user.name"));
        InetAddress localHost = InetAddress.getLocalHost();
        System.out.println("IP Address is :" + localHost.getHostAddress());

        System.out.println("Enter number of runners");
        Scanner sc = new Scanner(System.in);
        int num = sc.nextInt();
        List<RaceSimulation> runners = new ArrayList<>();

        for (int i = 1; i <= num; i++) {
            runners.add(new RaceSimulation(i));
        }

        List<Thread> threads = new ArrayList<>();
        for (RaceSimulation runner : runners) {
            Thread thread = new Thread(runner);
            threads.add(thread);
            thread.start();
            if(runner.id == 3)
                thread.interrupt();
        }

        for (Thread thread : threads) {
            thread.join();
        }

        runners.sort((o1, o2) -> Integer.compare(o2.distance, o1.distance));
        System.out.println("Top 3");
```

```
        for (int i = 0; i < 3 && i < runners.size(); i++) {
            System.out.println((i + 1) + ". " + runners.get(i).distance + " " +
                    runners.get(i).distance + " meters");
        }
        sc.close();
    }
}
```