FACULTY OF COMPUTER SCIENCE AND ENGINEERING

Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Topi

Lab Duration: 3 hrs. CS417 Parallel Processing Lab Marks: 10

Lab No: 05 Instructor: Ms. Anmol Sanaullah Khan Dated: 11/10/2024

Before performing tasks, keep in mind the following rules:

1. **CHEATING IS NOT ALLOWED. Looking at someone’s else screen is also cheating.**
2. **Mobile phone and internet usage are not allowed.**
3. **If you have any queries related to the task, you can ask instructors only. Never talk to each other until you are allowed.**
4. **Do not answer any query until you are asked.**
5. **Perform all the tasks.**
6. **Avoiding any of the above rules will lead to marks deduction.**

**Task 1 – Producer-Consumer Problem**

Write a program to simulate the producer-consumer problem efficiently using conditional variables.

* The producer generates a number (e.g., a random integer) and places it in a shared variable.
* The consumer waits until the producer has produced a number and then consumes it.
* After consuming, the consumer waits until the producer produces another number.
* Synchronization is done using a conditionalvariable so that the consumer only consumes after the producer has produced.

**Expected Outcome:**

Consumer: Waiting for the producer to produce...

Producer: Produced data: 42

Consumer: Consumed data: 42

Consumer: Waiting for the producer to produce...

Producer: Produced data: 73

Consumer: Consumed data: 73

...

**Task 2 – Traffic Light Simulation**

Imagine a trafficlight system controlling two roads: Road A and Road B. Each road has cars waiting to pass, but only one road can be green at any given time. You need to simulate this using threads and conditional variable.

* Two threads represent two roads (A and B).
* A third thread simulates the traffic light controller, which changes the light every 5 seconds.
* When Road A is green, cars on Road A can pass, and cars on Road B must wait (and vice versa).
* Use a conditional variable to signal cars on which road should go.
* The cars on the roads must wait until the traffic light controller gives them the signal to go.

**Expected Output:**

Traffic Light: Green for Road A, Red for Road B

Road A: Green light! Cars passing...

Road B: Red light. Waiting...

Road A: Green light! Cars passing...

Traffic Light: Green for Road B, Red for Road A

Road B: Green light! Cars passing...

Road A: Red light. Waiting...

Road B: Green light! Cars passing...

Task 3 – Parallel Array Sum using OpenMP

Write a program that calculates the sum of an array using OpenMP. Divide the array between multiple threads to parallelize the summation.

Steps:

* Initialize an array of 100 elements with random values.
* Use OpenMP to parallelize the loop that sums the elements.
* Use the reduction clause to ensure the sum is correctly computed.

Hint: Use #pragma omp parallel for reduction(+:sum)