HIGH PERFORMANCE COMPUTING

N.Soukya

2303A51401

Batch-24

Assignment-4

Barrier Synchronization (Two-Phase Computation)

Scenario

Perform a computation in two phases ensuring correct synchronization between phases.

Objective

To demonstrate barrier synchronization in parallel execution.

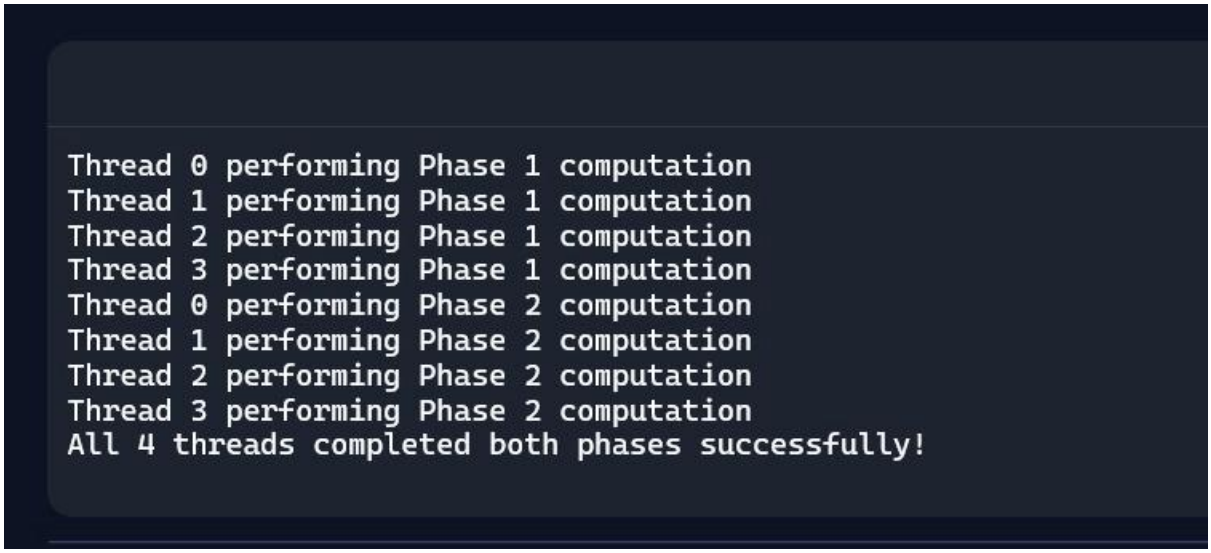
Tasks

1. Perform computation in Phase 1.
2. Synchronize all threads.
3. Perform computation in Phase 2.
4. Display completion message.

Code :

```
#include <stdio.h>
#include <omp.h>
int main()
{
   int num_threads;
   #pragma omp parallel
   {
      int tid = omp_get_thread_num();
      num_threads = omp_get_num_threads();
      // Phase 1 computation
      printf("Thread %d performing Phase 1 computation\n", tid);
      // Barrier synchronization
```

```
        #pragma omp barrier
        // Phase 2 computation
        printf("Thread %d performing Phase 2 computation\n", tid);
        // Barrier again (optional, ensures all threads finish Phase 2 before completion message)
        #pragma omp barrier
        // Only one thread prints completion message
        #pragma omp single
        {
            printf("All %d threads completed both phases successfully!\n", num_threads);
        }
    }
    return 0;
}
```



Learning outcomes :

- Understood barrier synchronization in parallel computing.
- Implemented multi-phase parallel programs.
- correct execution order using synchronization techniques.
- understood role of barriers in OpenMP-style parallel execution.