

# Assignment-4

**Name: K.Amulya**

**H.NO:2303A51242**

**Batch:05**

## Assignment 4: Barrier Synchronization (Two-Phase Computation)

### Scenario:

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

### Tasks:

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

### Code:

```
import threading
import time
num_threads = 4
barrier = threading.Barrier(num_threads)

def worker(thread_id):
    print(f"Thread {thread_id} : Starting Phase 1")
    time.sleep(1 + thread_id) # Simulate work
    print(f"Thread {thread_id} : Finished Phase 1")
    barrier.wait()
    print(f"Thread {thread_id} : Starting Phase 2")
    time.sleep(1)
    print(f"Thread {thread_id} : Finished Phase 2")
    print(f"Thread {thread_id} : Task Completed")

threads = []
for i in range(num_threads):
    t = threading.Thread(target=worker, args=(i,))
    threads.append(t)
    t.start()

for t in threads:
    t.join()

print("All Threads Have Completed Execution")
```

## Output:

```
•• Thread 0 : Starting Phase 1
    Thread 1 : Starting Phase 1
    Thread 2 : Starting Phase 1
    Thread 3 : Starting Phase 1
    Thread 0 : Finished Phase 1
    Thread 1 : Finished Phase 1
    Thread 2 : Finished Phase 1
    Thread 3 : Finished Phase 1
    Thread 3 : Starting Phase 2
    Thread 1 : Starting Phase 2
    Thread 0 : Starting Phase 2
    Thread 2 : Starting Phase 2
    Thread 3 : Finished Phase 2Thread 1 : Finished Phase 2
    Thread 1 : Task Completed
    Thread 2 : Finished Phase 2
    Thread 2 : Task Completed
    Thread 0 : Finished Phase 2
    Thread 0 : Task Completed

    Thread 3 : Task Completed
    All Threads Have Completed Execution
```

## Observation:

- All threads complete Phase 1 before starting Phase 2.
- Barrier ensures proper synchronization.
- No thread enters Phase 2 early.
- Program finishes only after all threads complete.