**Process Synchronization**

**Task#1**: Program to show the race condition.

Program to create two threads: one to increment the value of a shared variable and second

to decrement the value of shared variable. Both the threads are executed, so the final value

of shared variable should be same as its initial value. But due to race condition it would not

be same.

**Task#2:** Program for process synchronization using locks

Program create two threads: one to increment the value of a shared variable and second to

decrement the value of shared variable. Both the threads make use of locks so that only

one of the threads is executing in its critical section

**Task#3:** Simulate a printer system where multiple users can submit print jobs. Only one print job can be processed at a time, and others must wait in the queue. Use Mutex Lock to manage access to the printer.

:Number of print jobs and the time each job takes.

: Show when each print job is being processed and when the printer becomes available for the next job.