NAME: VARSHA D KULKARNI SRN: PES1UG19EC339

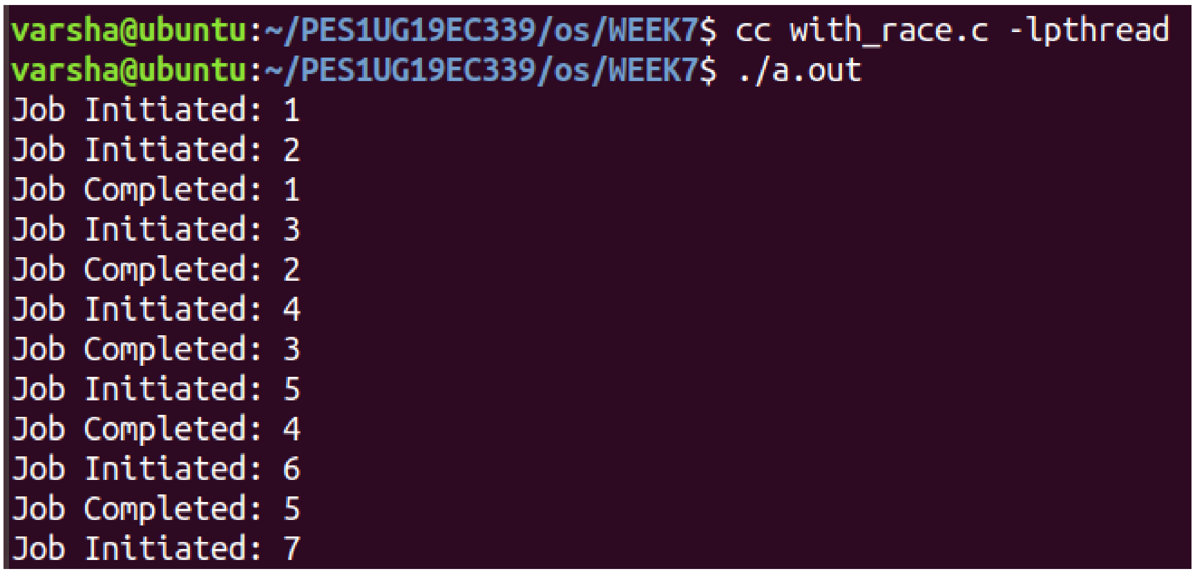
OS-LAB-WEEK-7

PROGRAM 1: Write a C Program to simulate race condition in Producer Consumer Problem Implement a main program that creates two threads.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <unistd.h>
#include <pthread.h>
#define BUFFER_SIZE 100

void *producer();
void *consumer();

int front = 0, rear = 0;
int item = 0;
int *buffer;

int main()
{
   buffer = (int *)malloc(sizeof(int) * BUFFER_SIZE);
   pthread_t producer_thread, consumer_thread;
   pthread_create(&producer_thread, NULL, producer, NULL);
   pthread_create(&consumer_thread, NULL, consumer, NULL);
   pthread_join(producer_thread, NULL);
   pthread_join(consumer_thread, NULL);
   free(buffer);
   return 0;
}

void *producer()
{
   while (true)
   {
      item += 1;
      printf("Job Initiated: %d\n", item);
      sleep(1);
      while (((front + 1) % BUFFER_SIZE) == rear)
         ;
      buffer[front] = item;
      front = (front + 1) % BUFFER_SIZE;
   }
}

void *consumer()
{
   while (true)
```

```
    {
        while (front == rear)
            ;
        int consumed = buffer[rear];
        printf("Job Completed: %d\n", consumed);
        sleep(1);
        rear = (rear + 1) % BUFFER_SIZE;
    }
}
```

OUTPUT:



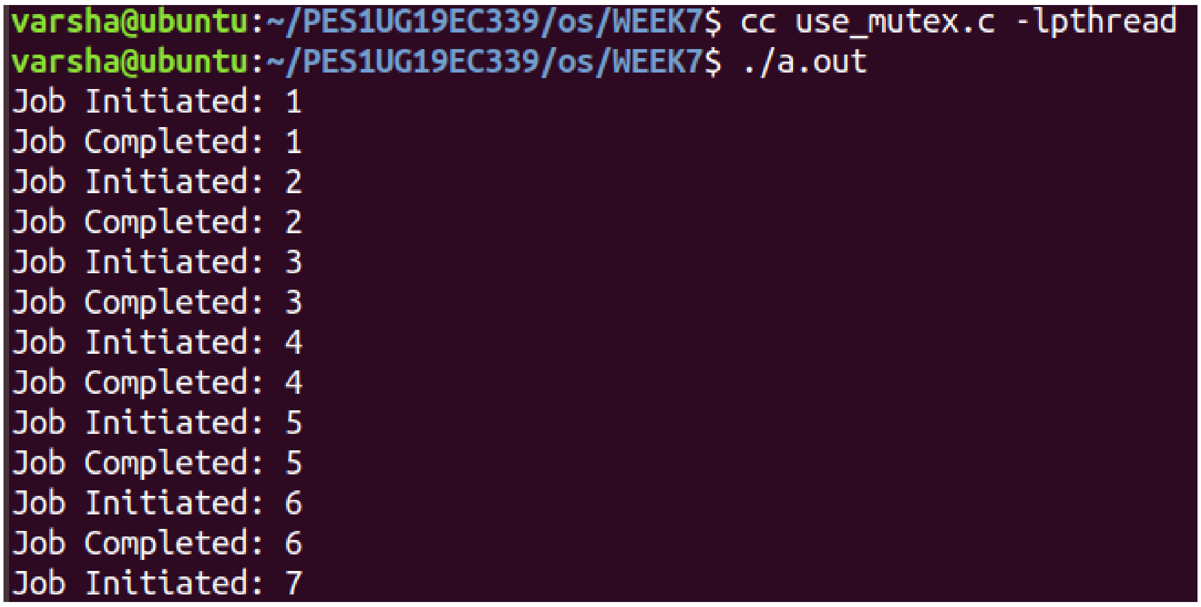
```
varsha@ubuntu:~/PES1UG19EC339/os/WEEK7$ cc with_race.c -lpthread
varsha@ubuntu:~/PES1UG19EC339/os/WEEK7$ ./a.out
Job Initiated: 1
Job Initiated: 2
Job Completed: 1
Job Initiated: 3
Job Completed: 2
Job Initiated: 4
Job Completed: 3
Job Initiated: 5
Job Completed: 4
Job Initiated: 6
Job Completed: 5
Job Initiated: 7
```

PROGRAM 2: Write a C program to implement Producer Consumer problem using Mutex.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <unistd.h>
#include <pthread.h>
#define BUFFER_SIZE 100

void *producer();
void *consumer();

int front = 0, rear = 0;
int *buffer;

pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
pthread_mutex_t empty = PTHREAD_MUTEX_INITIALIZER;
pthread_mutex_t full = PTHREAD_MUTEX_INITIALIZER;
```

```
int main()
{
   buffer = (int *)malloc(sizeof(int) * BUFFER_SIZE);
   pthread_t producer_thread, consumer_thread;
   pthread_create(&producer_thread, NULL, producer, NULL);
   sleep(1);
   pthread_create(&consumer_thread, NULL, consumer, NULL);
   pthread_join(producer_thread, NULL);
   pthread_join(consumer_thread, NULL);
   free(buffer);
   return 0;
}

void *producer()
{
   int item = 0;
   while (true)
   {
      pthread_mutex_lock(&empty);
      pthread_mutex_lock(&mutex);
      item += 1;
      printf("Job Initiated: %d\n", item);
      buffer[front] = item;
      pthread_mutex_unlock(&mutex);
      pthread_mutex_unlock(&full);
      front = (front + 1) % BUFFER_SIZE;
   }
}

void *consumer()
{
   while (true)
   {
      pthread_mutex_lock(&full);
      pthread_mutex_lock(&mutex);
      int consumed = buffer[rear];
      printf("Job Completed: %d\n", consumed);
      sleep(1);
      rear = (rear + 1) % BUFFER_SIZE;
      pthread_mutex_unlock(&mutex);
      pthread_mutex_unlock(&empty);
   }
}
```

OUTPUT



```
varsha@ubuntu:~/PES1UG19EC339/os/WEEK7$ cc use_mutex.c -lpthread
varsha@ubuntu:~/PES1UG19EC339/os/WEEK7$ ./a.out
Job Initiated: 1
Job Completed: 1
Job Initiated: 2
Job Completed: 2
Job Initiated: 3
Job Completed: 3
Job Initiated: 4
Job Completed: 4
Job Initiated: 5
Job Completed: 5
Job Initiated: 6
Job Completed: 6
Job Initiated: 7
```