NAME: TANYA CHANCHALANI

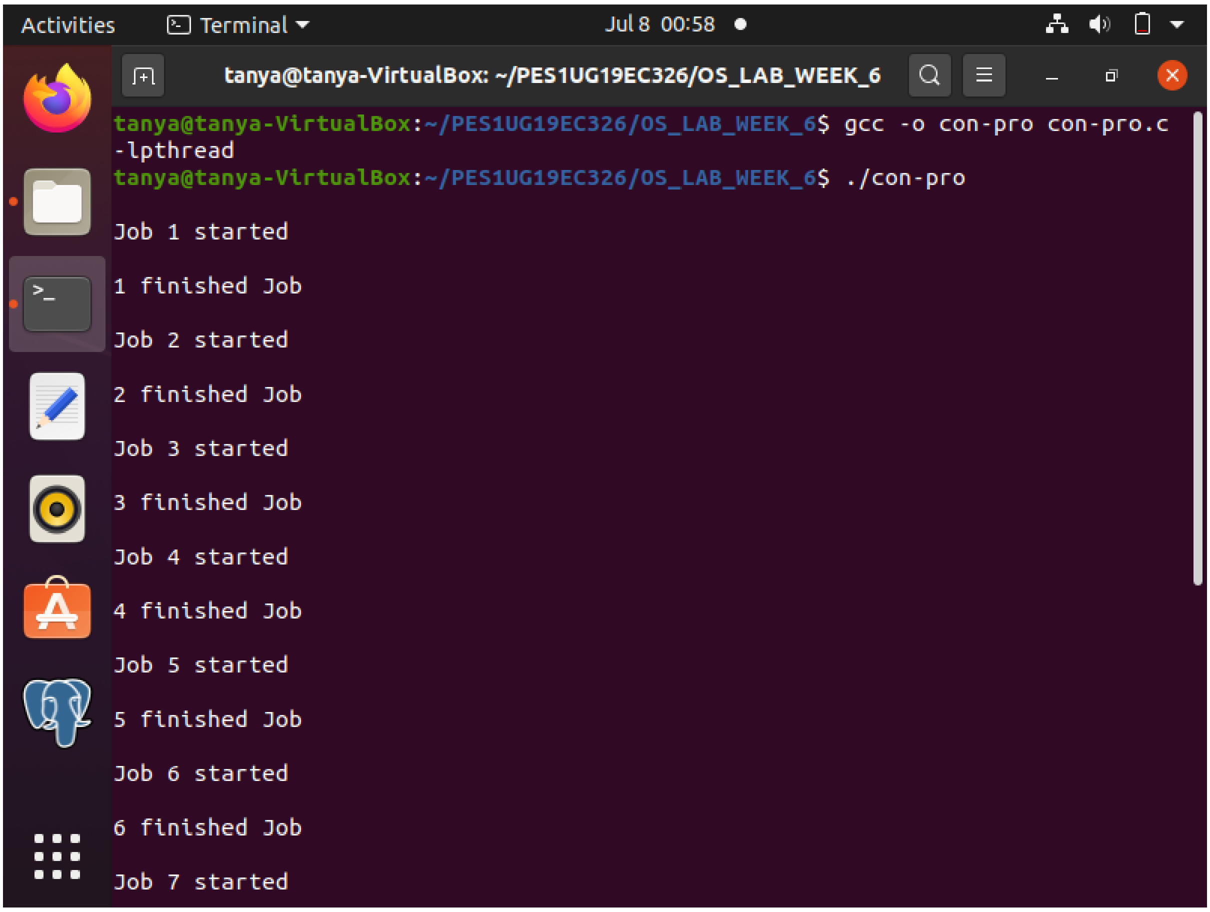
SRN: PES1UG19EC326

OSLAB_WEEK: 6

BRANCH: ECE

a)//using bounded buffer

```
#include<stdio.h>
#include<unistd.h>
#include<pthread.h>
#include<stdlib.h>
#include<stdbool.h>
#define BUFFER_SIZE 78
int *buffer;
int counter=0;
pthread_mutex_t full = PTHREAD_MUTEX_INITIALIZER;
pthread_mutex_t emt = PTHREAD_MUTEX_INITIALIZER;
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
int start=0,end=0;
void *consumer()
{
  while (true)
  {
    pthread_mutex_lock(&full);
    pthread_mutex_lock(&mutex);
    int consumed = buffer[end];
    printf("\n%d finished Job\n", consumed);
    sleep(1);
    end = (end + 1) % BUFFER_SIZE;
    pthread_mutex_unlock(&mutex);
    pthread_mutex_unlock(&emt);
  }
}
void *producer()
{
    int counter = 0;
    while (true)
    {
        pthread_mutex_lock(&emt);
        pthread_mutex_lock(&mutex);
        counter += 1;
        printf("\nJob %d started\n", counter);
        buffer[start] = counter;
        pthread_mutex_unlock(&mutex);
        pthread_mutex_unlock(&full);
        start = (start + 1) % BUFFER_SIZE;
    }
}
int main()
{
    pthread_t producer_t, consumer_t;
    void *producer();
    void *consumer();
    buffer = (int *)malloc(sizeof(int) * BUFFER_SIZE);
    pthread_create(&producer_t, NULL, producer, NULL);
    sleep(1);
    pthread_create(&consumer_t, NULL, consumer, NULL);
    pthread_join(producer_t, NULL);
    pthread_join(consumer_t, NULL);
    free(buffer);
    return 0;
}
```



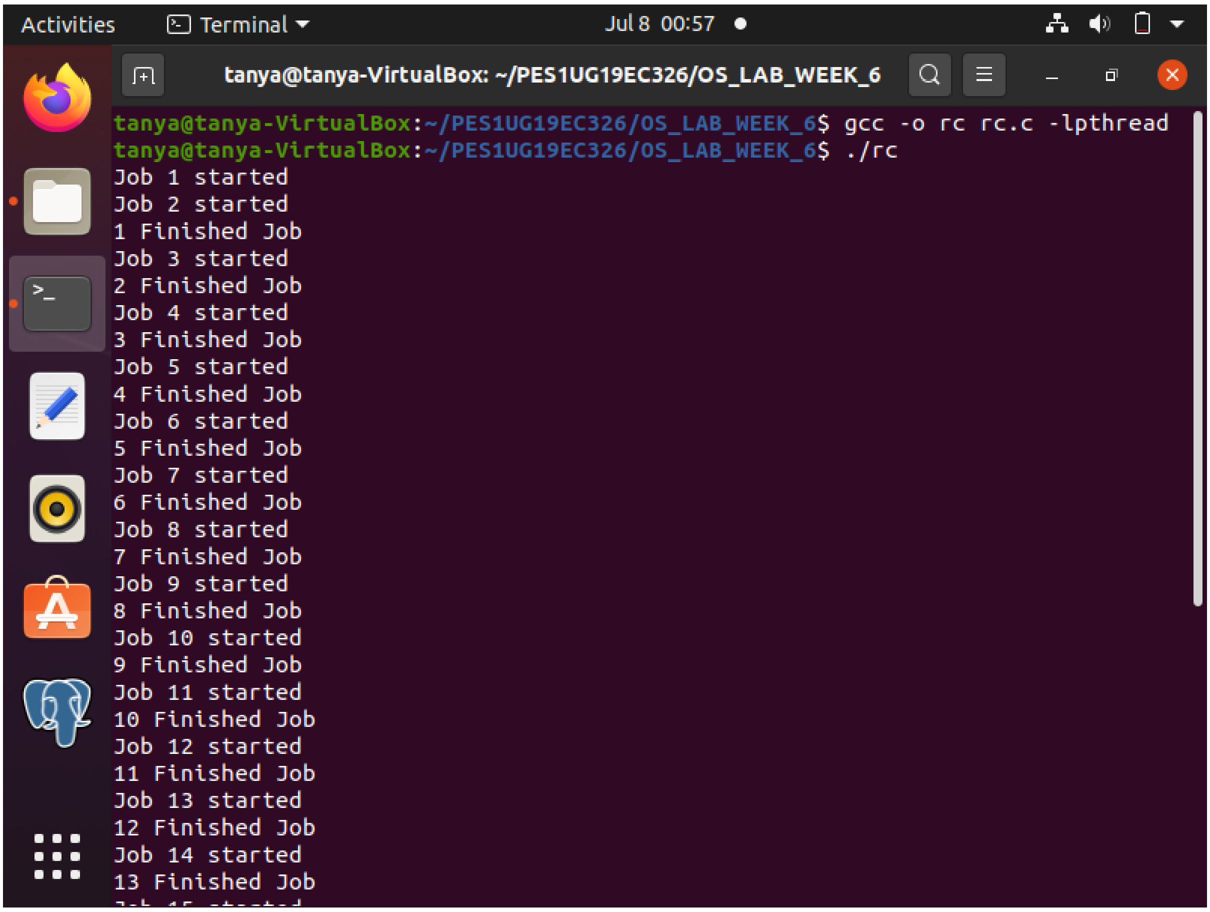
```
tanya@tanya-VirtualBox:~/PES1UG19EC326/OS_LAB_WEEK_6$ gcc -o con-pro con-pro.c
-lpthread
tanya@tanya-VirtualBox:~/PES1UG19EC326/OS_LAB_WEEK_6$ ./con-pro

Job 1 started

1 finished Job

Job 2 started

2 finished Job

Job 3 started

3 finished Job

Job 4 started

4 finished Job

Job 5 started

5 finished Job

Job 6 started

6 finished Job

Job 7 started
```

//the expected value was something else but due to overlapping of operations of both the threads
//the value is shown that of the last execution made by the last thread - race condition

b)//using bounded buffer

```
#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include <unistd.h>

#include <pthread.h>

#define BUFFER_SIZE 90

int start = 0, end = 0;

int *buffer;

int counter = 0;

void *producer()

{
    while (true)
    {
        counter += 1;
        printf("Job %d started\n", counter);
        sleep(1);
        while (((start + 1) % BUFFER_SIZE) == end)
            ;
        buffer[start] = counter;
        start = (start + 1) % BUFFER_SIZE;
    }
}
void *consumer()
{
    while (true)
    {
        while (start == end)
            ;
        int finished = buffer[end];
        printf("%d Finished Job\n", finished);
        sleep(1);
        end = (end + 1) % BUFFER_SIZE;
    }
}
int main()
{

    void *producer();
    void *consumer();
    buffer = (int *)malloc(sizeof(int) * BUFFER_SIZE);
    pthread_t producer_t, consumer_t;
    pthread_create(&producer_t, NULL, producer, NULL);
```

```
  pthread_create(&consumer_t, NULL, consumer, NULL);

  pthread_join(producer_t, NULL);

  pthread_join(consumer_t, NULL);

  free(buffer);

  return 0;

}
```



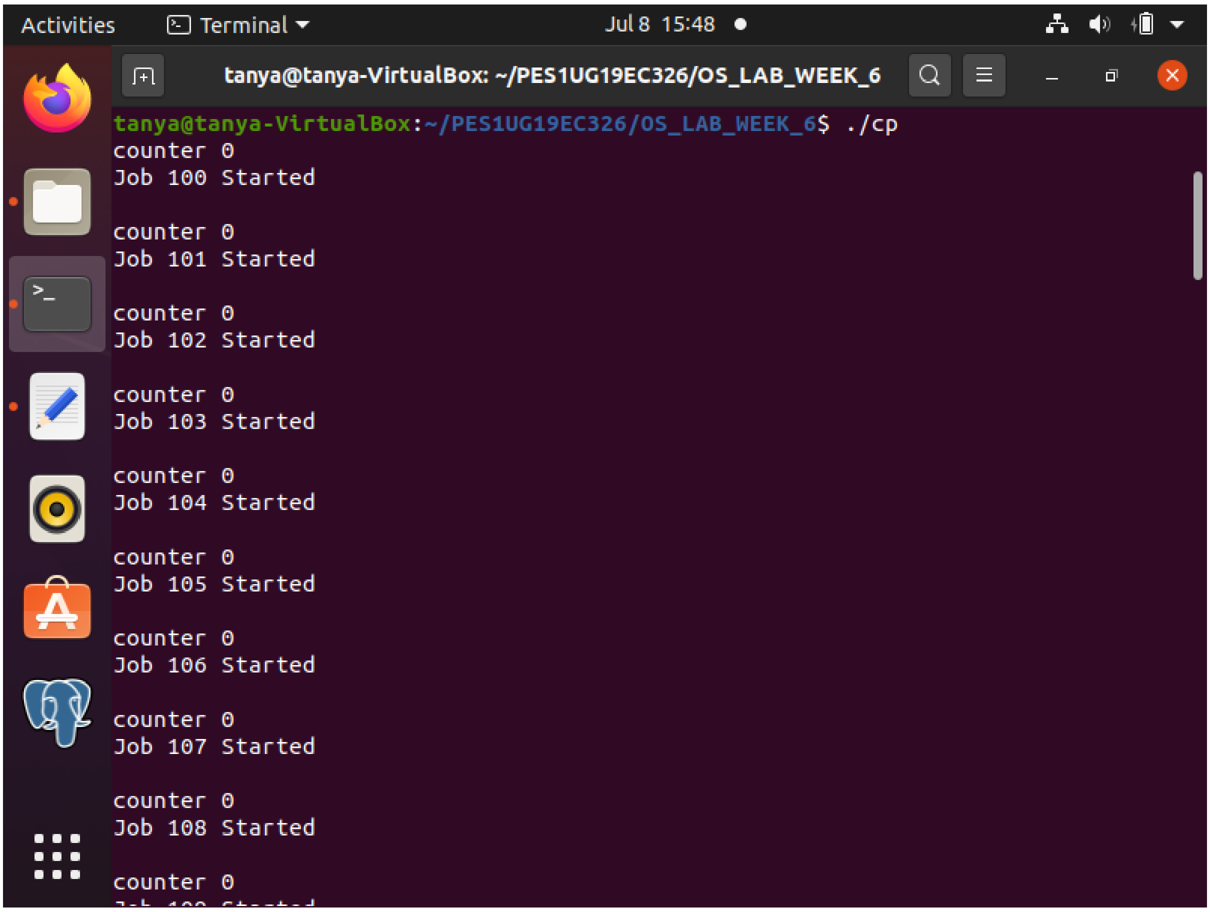
```
tanya@tanya-VirtualBox:~/PES1UG19EC326/OS_LAB_WEEK_6$ gcc -o rc rc.c -lpthread
tanya@tanya-VirtualBox:~/PES1UG19EC326/OS_LAB_WEEK_6$ ./rc
Job 1 started
Job 2 started
1 Finished Job
Job 3 started
2 Finished Job
Job 4 started
3 Finished Job
Job 5 started
4 Finished Job
Job 6 started
5 Finished Job
Job 7 started
6 Finished Job
Job 8 started
7 Finished Job
Job 9 started
8 Finished Job
Job 10 started
9 Finished Job
Job 11 started
10 Finished Job
Job 12 started
11 Finished Job
Job 13 started
12 Finished Job
Job 14 started
13 Finished Job
Job 15 started
```

<u>Practice programs:-</u>

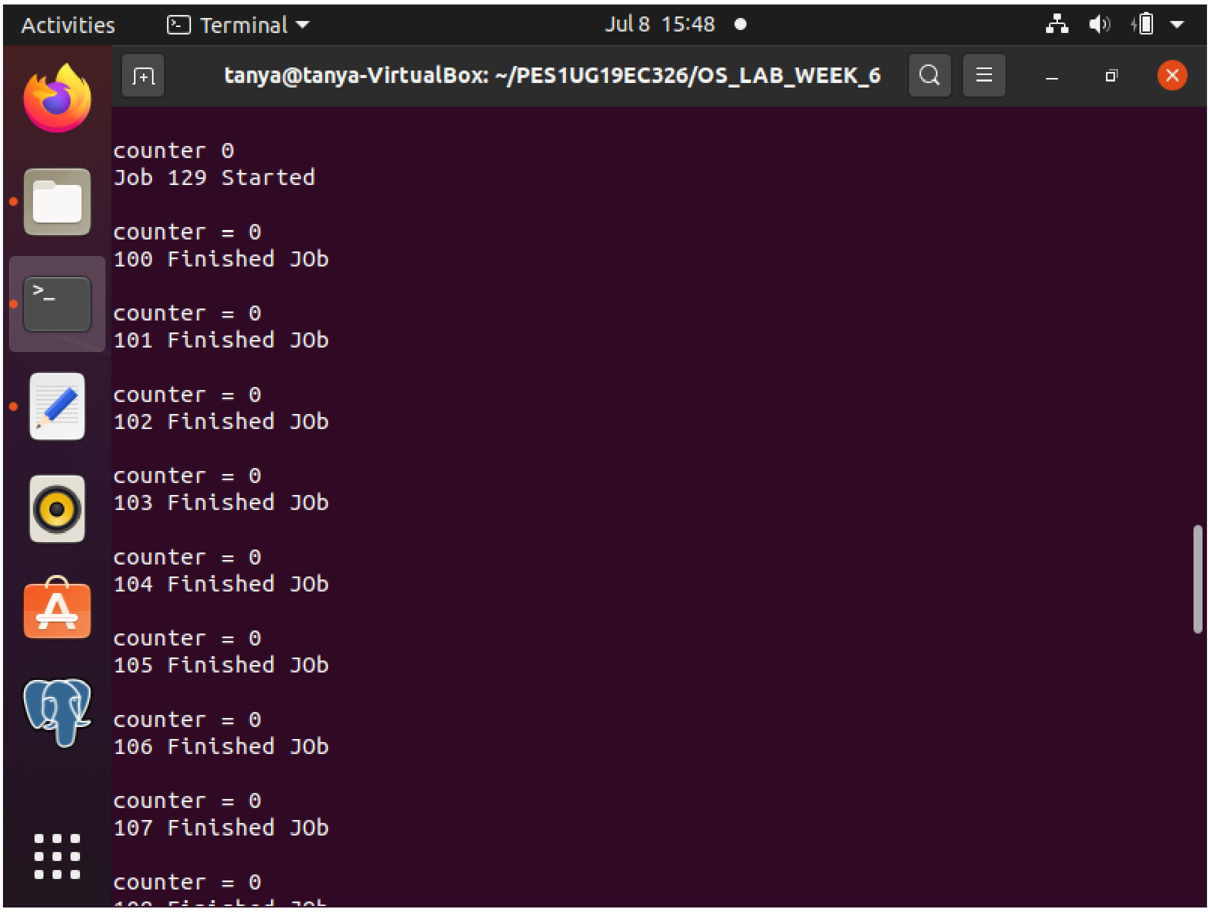
c) //showing with a counter and using binary semaphore

```
#include<stdio.h>

#include<pthread.h>

#include<unistd.h>

#include<stdlib.h>

#define BUFFER_SIZE 30

struct pseudo_sem {

  int value;
```

```
    pthread_mutex_t mutex;
};
struct pseudo_sem s = { 1, PTHREAD_MUTEX_INITIALIZER };
struct pseudo_sem full = { 0, PTHREAD_MUTEX_INITIALIZER };
struct pseudo_sem empt = { BUFFER_SIZE, PTHREAD_MUTEX_INITIALIZER };
int arr[BUFFER_SIZE];
int value = 100;
void wait(struct pseudo_sem *s)
{
    pthread_mutex_lock(&s->mutex);
    while(s->value <= 0) {
        pthread_mutex_unlock(&s->mutex);
        usleep(100);
        pthread_mutex_lock(&s->mutex);
    }
    s->value--;
    pthread_mutex_unlock(&s->mutex);
}

void signal(struct pseudo_sem* s)
{
    pthread_mutex_lock(&s->mutex);
    s->value++;
    pthread_mutex_unlock(&s->mutex);
}
void *producer(void* param)
{
    for(int i=0;i<BUFFER_SIZE;i++)
    {
        int new_item = value;
        value++;
        wait(&empt);
        wait(&s);

        printf("counter %d\n",s);
        //printf("Producer inside critical section\n");
        printf("Job %d Started\n\n",new_item);
        arr[i] = new_item;

        signal(&s);
        signal(&full);
    }
    pthread_exit(0);
}
void *consumer(void* param)
{
    for(int i=0;i<BUFFER_SIZE;i++)
    {
        wait(&full);
        wait(&s);

        printf("counter = %d\n",s);
        //printf("Consumer inside critical section\n");
        printf("%d Finished JOb\n\n", arr[i]);

        signal(&s);
        signal(&empt);
    }
    pthread_exit(0);
}
int main()
{
    pthread_t tid_p,tid_c;

    pthread_attr_t attr1,attr2;

    pthread_attr_init(&attr1);

    pthread_attr_init(&attr2);


    pthread_create(&tid_p,&attr1,producer,NULL);

    pthread_create(&tid_c,&attr2,consumer,NULL);


    pthread_join(tid_p,NULL);

    pthread_join(tid_c,NULL);

    return 0;

}
```



```
tanya@tanya-VirtualBox:~/PES1UG19EC326/OS_LAB_WEEK_6$ ./cp
counter 0
Job 100 Started

counter 0
Job 101 Started

counter 0
Job 102 Started

counter 0
Job 103 Started

counter 0
Job 104 Started

counter 0
Job 105 Started

counter 0
Job 106 Started

counter 0
Job 107 Started

counter 0
Job 108 Started

counter 0
```



```
counter 0
Job 129 Started

counter = 0
100 Finished JOb

counter = 0
101 Finished JOb

counter = 0
102 Finished JOb

counter = 0
103 Finished JOb

counter = 0
104 Finished JOb

counter = 0
105 Finished JOb

counter = 0
106 Finished JOb

counter = 0
107 Finished JOb

counter = 0
108 Finished JOb
```

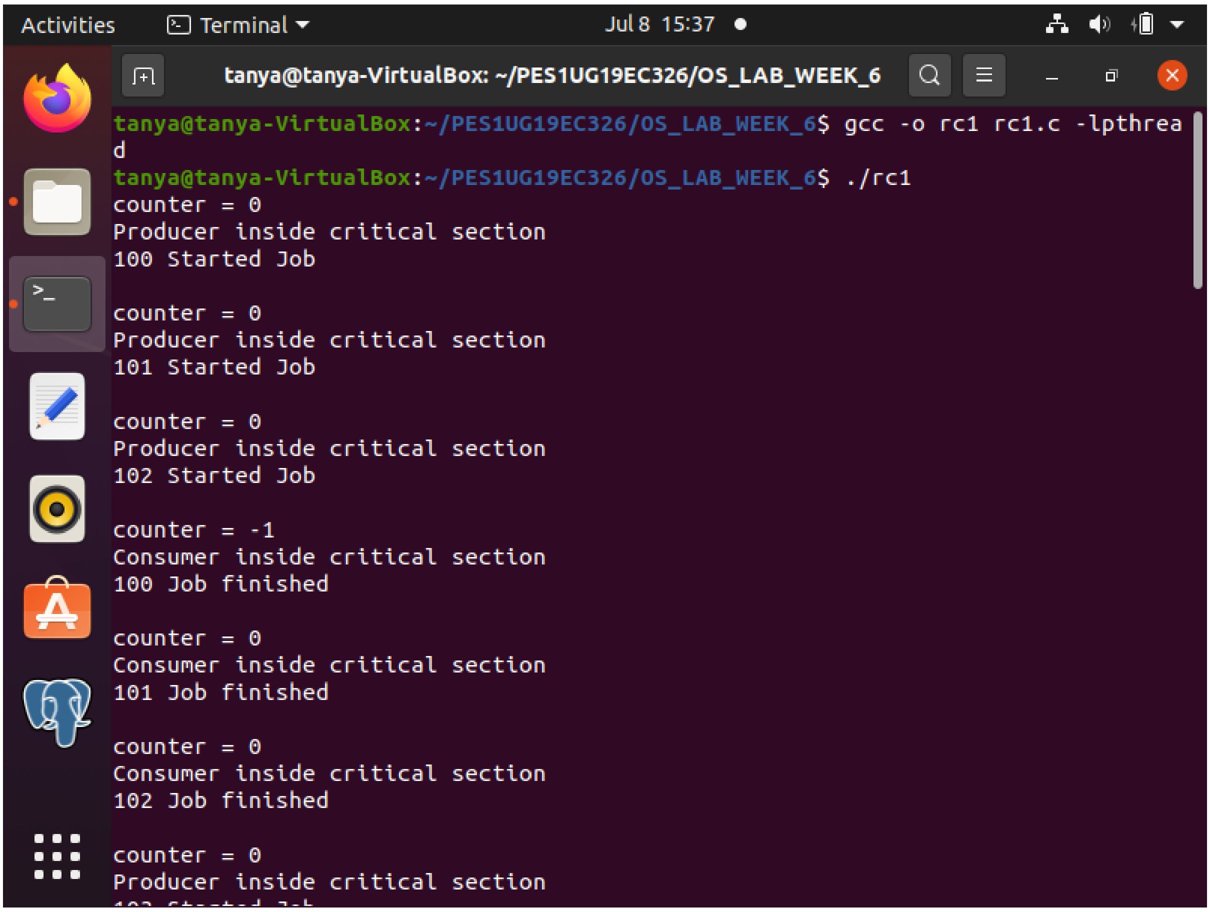
d) //showing race condition using binary semaphore

```
#include<unistd.h>

#include<stdlib.h>

#include<stdio.h>

#include<pthread.h>

#define BUFF_SIZ 15

int s = 1;

int full = 0;

int empt = BUFF_SIZ;

int arr[BUFF_SIZ];

int value = 100;

void wait(int *s)

{

   while(*s <= 0);

   *s = *s - 1;
```

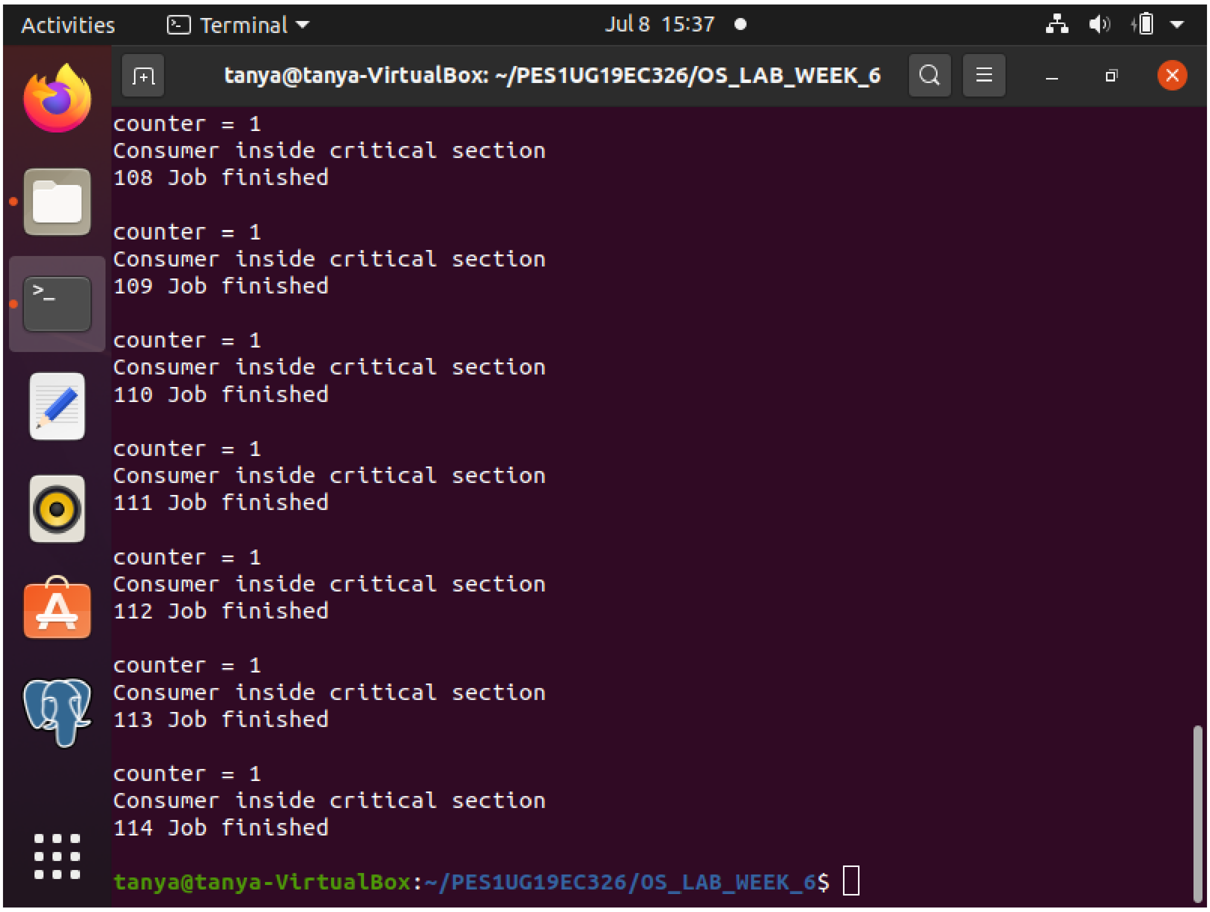
```
}
void signal(int* s)
{
  *s = *s + 1;
}
void *producer(void* param)
{
  for(int i=0;i<BUFF_SIZ;i++)
  {
    int new_item = value;
    value++;
    wait(&empt);
    wait(&s);

    printf("counter = %d\n",s);
    printf("Producer inside critical section\n");
    printf("%d Started Job\n\n",new_item);
    arr[i] = new_item;

    signal(&s);
    signal(&full);
  }
  pthread_exit(0);
}
void *consumer(void* param)
{
  for(int i=0;i<BUFF_SIZ;i++)
  {
    wait(&full);
    wait(&s);
```

```
        printf("counter = %d\n",s);
        printf("Consumer inside critical section\n");
        printf("%d Job finished\n\n", arr[i]);


        signal(&s);
        signal(&empt);
    }
    pthread_exit(0);
}
int main()
{
    pthread_t tid_p,tid_c;
    pthread_attr_t attr1,attr2;
    pthread_attr_init(&attr1);
    pthread_attr_init(&attr2);


    pthread_create(&tid_p,&attr1,producer,NULL);
    pthread_create(&tid_c,&attr2,consumer,NULL);


    pthread_join(tid_p,NULL);
    pthread_join(tid_c,NULL);
    return 0;
```

}



```
tanya@tanya-VirtualBox:~/PES1UG19EC326/OS_LAB_WEEK_6$ gcc -o rc1 rc1.c -lpthrea
d
tanya@tanya-VirtualBox:~/PES1UG19EC326/OS_LAB_WEEK_6$ ./rc1
counter = 0
Producer inside critical section
100 Started Job

counter = 0
Producer inside critical section
101 Started Job

counter = 0
Producer inside critical section
102 Started Job

counter = -1
Consumer inside critical section
100 Job finished

counter = 0
Consumer inside critical section
101 Job finished

counter = 0
Consumer inside critical section
102 Job finished

counter = 0
Producer inside critical section
```



```
counter = 1
Consumer inside critical section
108 Job finished

counter = 1
Consumer inside critical section
109 Job finished

counter = 1
Consumer inside critical section
110 Job finished

counter = 1
Consumer inside critical section
111 Job finished

counter = 1
Consumer inside critical section
112 Job finished

counter = 1
Consumer inside critical section
113 Job finished

counter = 1
Consumer inside critical section
114 Job finished

tanya@tanya-VirtualBox:~/PES1UG19EC326/OS_LAB_WEEK_6$ 
```