Name - Kaushal Oza
Roll no - 40
SRN – 201900754
DIV – A

♦ ESE LAB OPERATING SYSTEMS ♦

- Problem Statement-
  Implement the Producer Consumer Problem using threads and mutex

- Algorithm-

1) Create two threads one for producer and one for consumer.
2) Initialise 2 threads one for producer and one for consumer then we'll initialize mutex
3) After that we will associate our producer and consumer to the 2 threads we created in step 1.
4) In the producer function we will first create a random item using a random integer.
5) We will call the mutex lock function and pass the mutex variable that we initialised in step 2 (Which will prevent the consumer from accessing the buffer if it is empty).
6) Now, we will store the item into the buffer and unlock the mutex by passing it into the mutex unlock() function which will check if the mutex = 0. If yes, then it will with 1 .(This will allow

the consumer to access the buffer).

7) In the consumer function we will check the mutex by passing it into the mutex log function (It will check if the mutex is equal to 0. If yes, then it will pause or block the consumer. If not, the consumer will be able to access the buffer.
8) Now, after the consumer is into the buffer it will retrieve the item and it will unlock the mutex, by passing it into the mutex unlock function (It will check if the mutex is set to 0. If yes, then it will increment by 1).

- CODE-

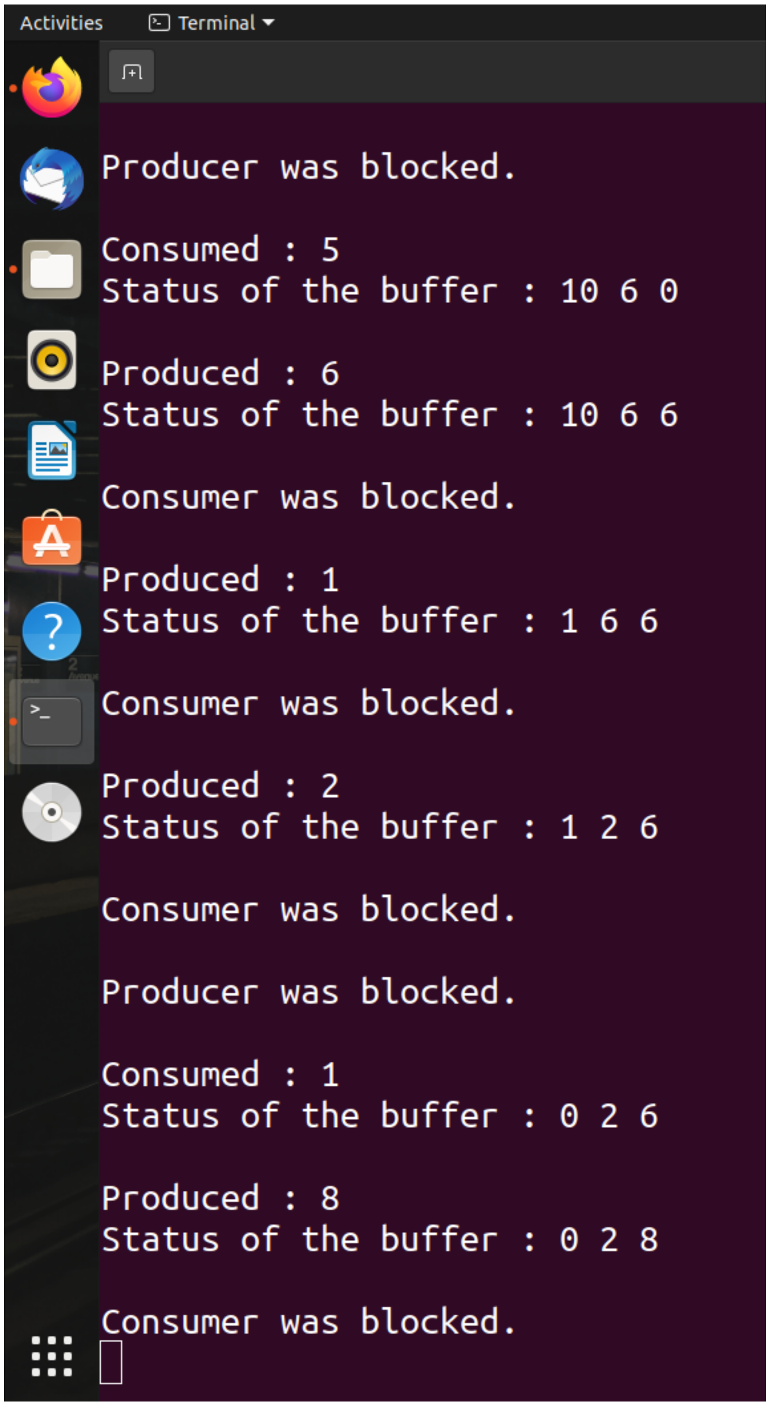
```
#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <pthread.h>

#include <time.h>
```

```
#include <math.h>

#define Iterations 100

#define Size 3 //BUFFER SIZE = 3

int in = 0;

int out = 0;

int buffer[Size] = {0};

pthread_mutex_t mutex;

void show()

{
        for (int i = 0; i < Size; i++)

        printf("%d ", buffer[i]);

        printf("\n");

}



void *prod(void *_args)      //PRODUCER
FUNCTION

{

        int item;

        for (int i = 0; i < Iterations; i++)

        {

        printf("\nConsumer was blocked.\n");
        sleep(rand() % 3);

        item = 1 + rand() % 10;

        pthread_mutex_lock(&mutex); //Same as
Wait() operation

        buffer[in] = item;

        printf("\nProduced : %d", buffer[in]);

        printf("\nStatus of the buffer : ");

        show();

        in = (in + 1) % Size;

        pthread_mutex_unlock(&mutex);  //Same as
signal() operation

        }

}
```

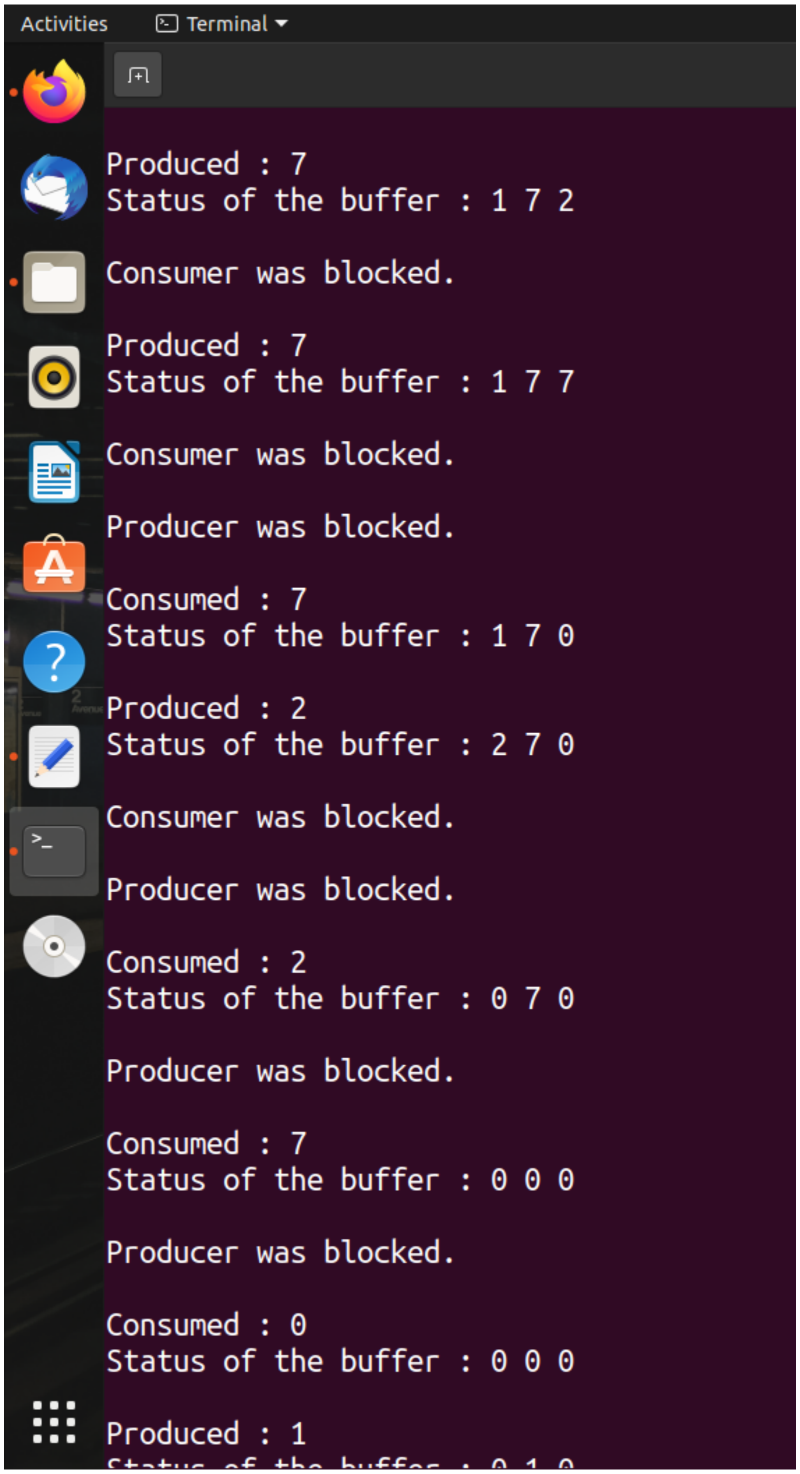
```
void *cons(void *_args)      //CONSUMER
FUNCTION

{

        for (int i = 0; i < Iterations; i++)

        {

        sleep(rand() % 3);

        printf("\nProducer was blocked.\n");

        int item = buffer[out];

        pthread_mutex_lock(&mutex); //Same as
Wait() operation

        buffer[out] = 0;

        printf("\nConsumed : %d", item);
        printf("\nStatus of the buffer : ");

        show();

        out = (out + 1) % Size;

        pthread_mutex_unlock(&mutex);  //Same as
signal() operation

        }

}




int main()

{



        pthread_t pro, con, pro1, con1;
        /* INITIALISED THE MUTEX */

        pthread_mutex_init(&mutex, NULL);



        /* FIRST SET OF PRODUCER AND
CONSUMER */

        pthread_create(&pro, NULL, prod, NULL);

        pthread_create(&con, NULL, cons, NULL);



        sleep(20);  //DELAY TIME



        /* SECOND SET OF PRODUCER AND
CONSUMER */
```

```
    }

        pthread_create(&pro1, NULL, prod, NULL);

        pthread_create(&con1, NULL, cons, NULL);



        sleep(20);  //DELAY TIME



        pthread_join(pro, NULL);

        pthread_join(con, NULL);



        /* DESTROYED THE MUTEX */

        pthread_mutex_destroy(&mutex);

        return 0;

}
```

- SCREENSHOTS/ OUTPUT-



```
Producer was blocked.

Consumed : 5
Status of the buffer : 10 6 0

Produced : 6
Status of the buffer : 10 6 6

Consumer was blocked.

Produced : 1
Status of the buffer : 1 6 6

Consumer was blocked.

Produced : 2
Status of the buffer : 1 2 6

Consumer was blocked.

Producer was blocked.

Consumed : 1
Status of the buffer : 0 2 6

Produced : 8
Status of the buffer : 0 2 8

Consumer was blocked.
```



```
Produced : 7
Status of the buffer : 1 7 2

Consumer was blocked.

Produced : 7
Status of the buffer : 1 7 7

Consumer was blocked.

Producer was blocked.

Consumed : 7
Status of the buffer : 1 7 0

Produced : 2
Status of the buffer : 2 7 0

Consumer was blocked.

Producer was blocked.

Consumed : 2
Status of the buffer : 0 7 0

Producer was blocked.

Consumed : 7
Status of the buffer : 0 0 0

Producer was blocked.

Consumed : 0
Status of the buffer : 0 0 0

Produced : 1
Status of the buffer : 0 1 0
```