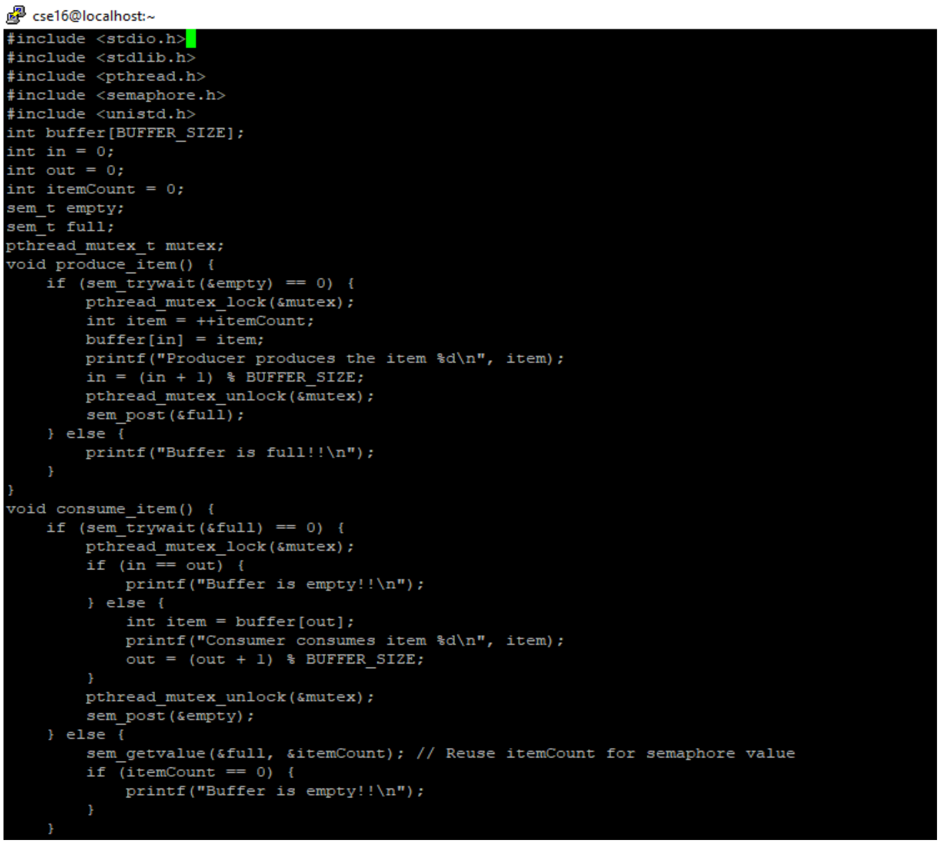
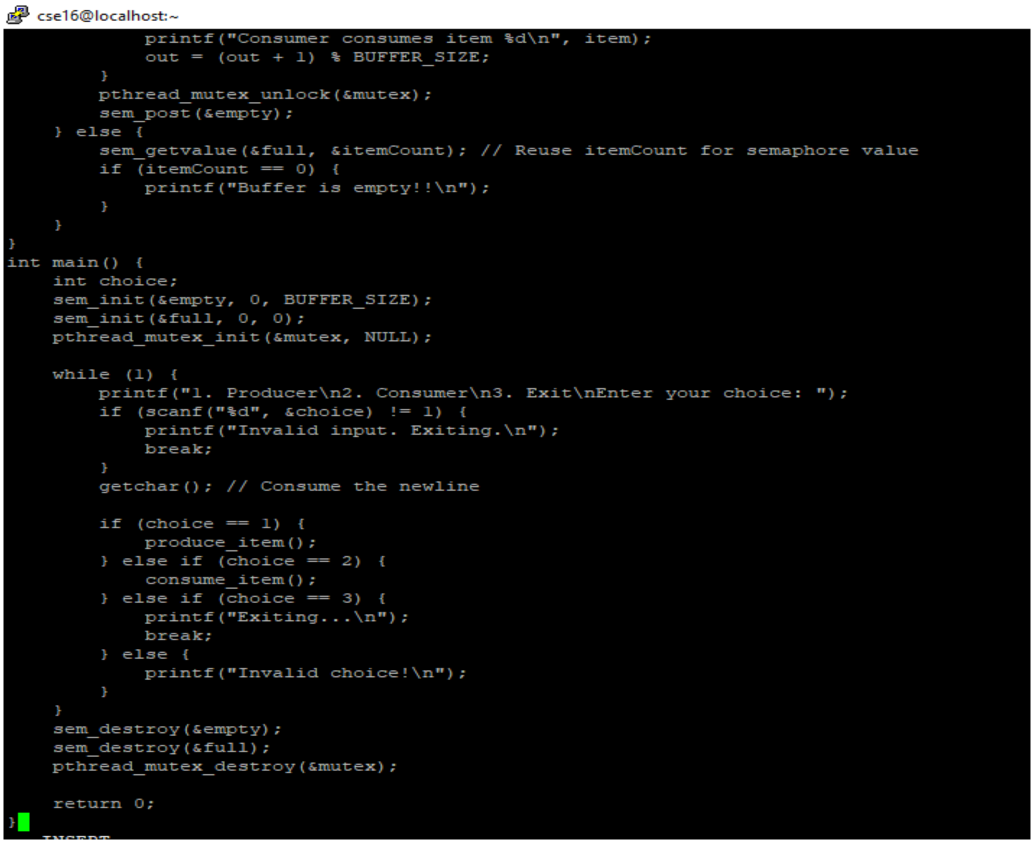
8.

CODE-



cse16@localhost:~

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>
int buffer[BUFFER_SIZE];
int in = 0;
int out = 0;
int itemCount = 0;
sem_t empty;
sem_t full;
pthread_mutex_t mutex;
void produce_item() {
    if (sem_trywait(&empty) == 0) {
        pthread_mutex_lock(&mutex);
        int item = ++itemCount;
        buffer[in] = item;
        printf("Producer produces the item %d\n", item);
        in = (in + 1) % BUFFER_SIZE;
        pthread_mutex_unlock(&mutex);
        sem_post(&full);
    } else {
        printf("Buffer is full!!\n");
    }
}
void consume_item() {
    if (sem_trywait(&full) == 0) {
        pthread_mutex_lock(&mutex);
        if (in == out) {
            printf("Buffer is empty!!\n");
        } else {
            int item = buffer[out];
            printf("Consumer consumes item %d\n", item);
            out = (out + 1) % BUFFER_SIZE;
        }
        pthread_mutex_unlock(&mutex);
        sem_post(&empty);
    } else {
        sem_getvalue(&full, &itemCount); // Reuse itemCount for semaphore value
        if (itemCount == 0) {
            printf("Buffer is empty!!\n");
        }
    }
```



cse16@localhost:~

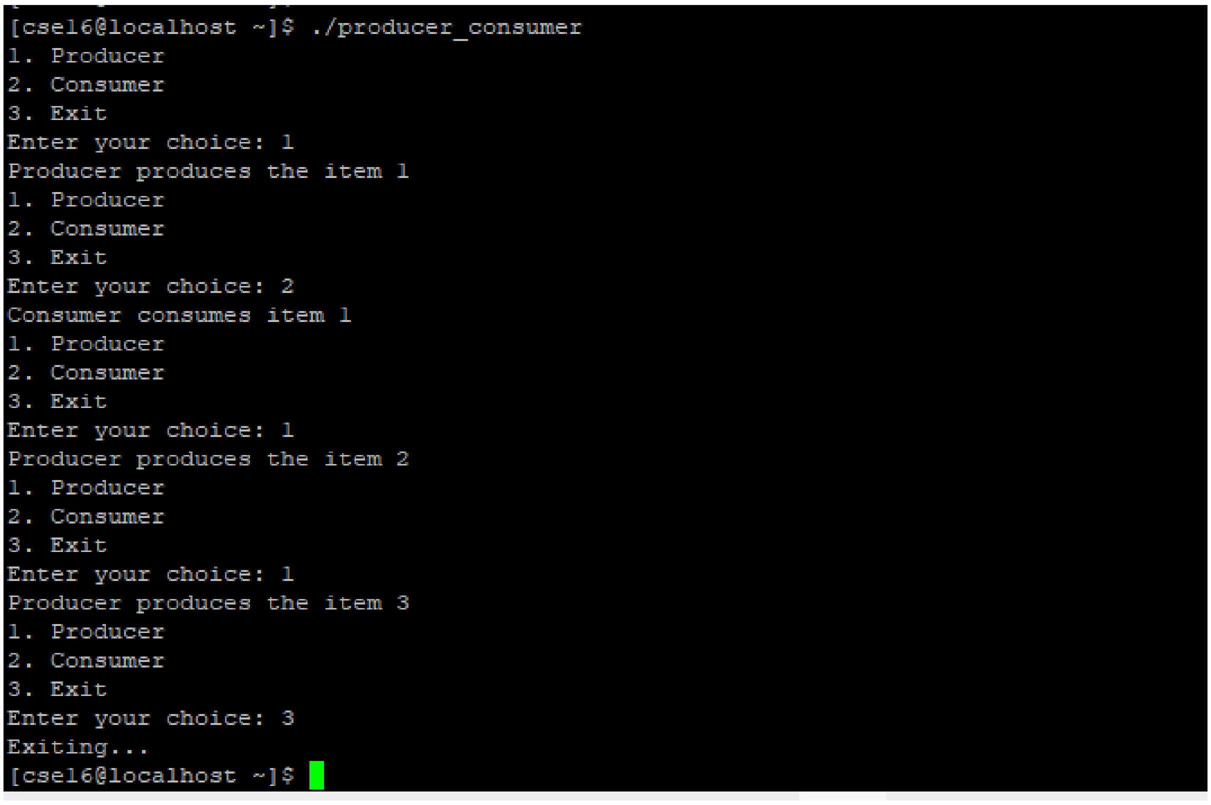
```
            printf("Consumer consumes item %d\n", item);
            out = (out + 1) % BUFFER_SIZE;
        }
        pthread_mutex_unlock(&mutex);
        sem_post(&empty);
    } else {
        sem_getvalue(&full, &itemCount); // Reuse itemCount for semaphore value
        if (itemCount == 0) {
            printf("Buffer is empty!!\n");
        }
    }
}
int main() {
    int choice;
    sem_init(&empty, 0, BUFFER_SIZE);
    sem_init(&full, 0, 0);
    pthread_mutex_init(&mutex, NULL);

    while (1) {
        printf("1. Producer\n2. Consumer\n3. Exit\nEnter your choice: ");
        if (scanf("%d", &choice) != 1) {
            printf("Invalid input. Exiting.\n");
            break;
        }
        getchar(); // Consume the newline

        if (choice == 1) {
            produce_item();
        } else if (choice == 2) {
            consume_item();
        } else if (choice == 3) {
            printf("Exiting...\n");
            break;
        } else {
            printf("Invalid choice!\n");
        }
    }
    sem_destroy(&empty);
    sem_destroy(&full);
    pthread_mutex_destroy(&mutex);

    return 0;
}
```

OUTPUT-



```
[cse16@localhost ~]$ ./producer_consumer
1. Producer
2. Consumer
3. Exit
Enter your choice: 1
Producer produces the item 1
1. Producer
2. Consumer
3. Exit
Enter your choice: 2
Consumer consumes item 1
1. Producer
2. Consumer
3. Exit
Enter your choice: 1
Producer produces the item 2
1. Producer
2. Consumer
3. Exit
Enter your choice: 1
Producer produces the item 3
1. Producer
2. Consumer
3. Exit
Enter your choice: 3
Exiting...
[cse16@localhost ~]$
```