U18CO018

Shubham Shekhaliya

Assignment – 9 (OS)

1. Implement Producer Consumer Problem solution using Semaphore.

Description

The producer and consumer share a fixed-size buffer used as a queue. The producer’s job is to generate data and put this in the buffer. The consumer’s job is to consume the data from this buffer, one at a time.

The producer should go to sleep when buffer is full. Next time when consumer removes data it notifies the producer and producer starts producing data again. The consumer should go to sleep when buffer is empty. Next time when producer add data it notifies the consumer and

consumer starts consuming data. This solution can be achieved using semaphores.

**Code :-**

#include <pthread.h>

#include <semaphore.h>

#include <stdlib.h>

#include <stdio.h>

#define MaxItems 5   // Maximum items a producer can produce or a consumer can consume

#define BufferSize 5 // Size of the buffer

sem\_t empty;

sem\_t full;

int in = 0;

int out = 0;

int buffer[BufferSize];

pthread\_mutex\_t mutex;

void \*producer(void \*pno) {

    int item;

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

        item = rand(); // Produce an random item

        sem\_wait(&empty);

        pthread\_mutex\_lock(&mutex);

        buffer[in] = item;

        printf("Producer %d: Insert Item %d at %d\n", \*((int \*)pno), buffer[in], in);

        in = (in + 1) % BufferSize;

        pthread\_mutex\_unlock(&mutex);

        sem\_post(&full);

    }

}

void \*consumer(void \*cno) {

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

        sem\_wait(&full);

        pthread\_mutex\_lock(&mutex);

        int item = buffer[out];

        printf("Consumer %d: Remove Item %d from %d\n", \*((int \*)cno), item, out);

        out = (out + 1) % BufferSize;

        pthread\_mutex\_unlock(&mutex);

        sem\_post(&empty);

    }

}

int main() {

    pthread\_t pro[5], con[5];

    pthread\_mutex\_init(&mutex, NULL);

    sem\_init(&empty, 0, BufferSize);

    sem\_init(&full, 0, 0);

    int a[5] = {1, 2, 3, 4, 5}; //Just used for numbering the producer and consumer

    for (int i = 0; i < 5; i++)

        pthread\_create(&pro[i], NULL, producer, (void \*)&a[i]);

    for (int i = 0; i < 5; i++)

        pthread\_create(&con[i], NULL, consumer, (void \*)&a[i]);

    for (int i = 0; i < 5; i++)

        pthread\_join(pro[i], NULL);

    for (int i = 0; i < 5; i++)

        pthread\_join(con[i], NULL);

    pthread\_mutex\_destroy(&mutex);

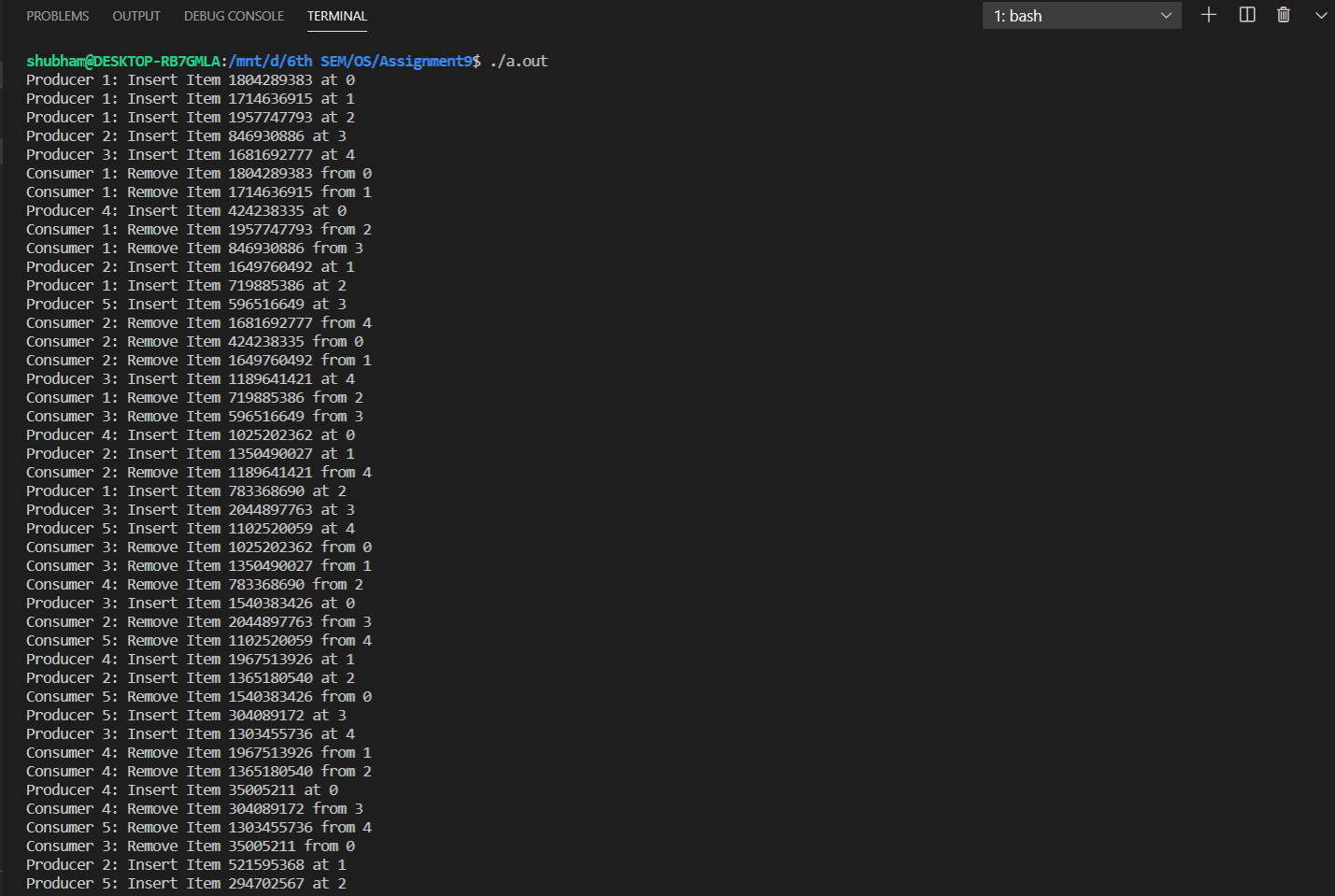
    sem\_destroy(&empty);

    sem\_destroy(&full);

    return 0;

}

**Output:-**

****