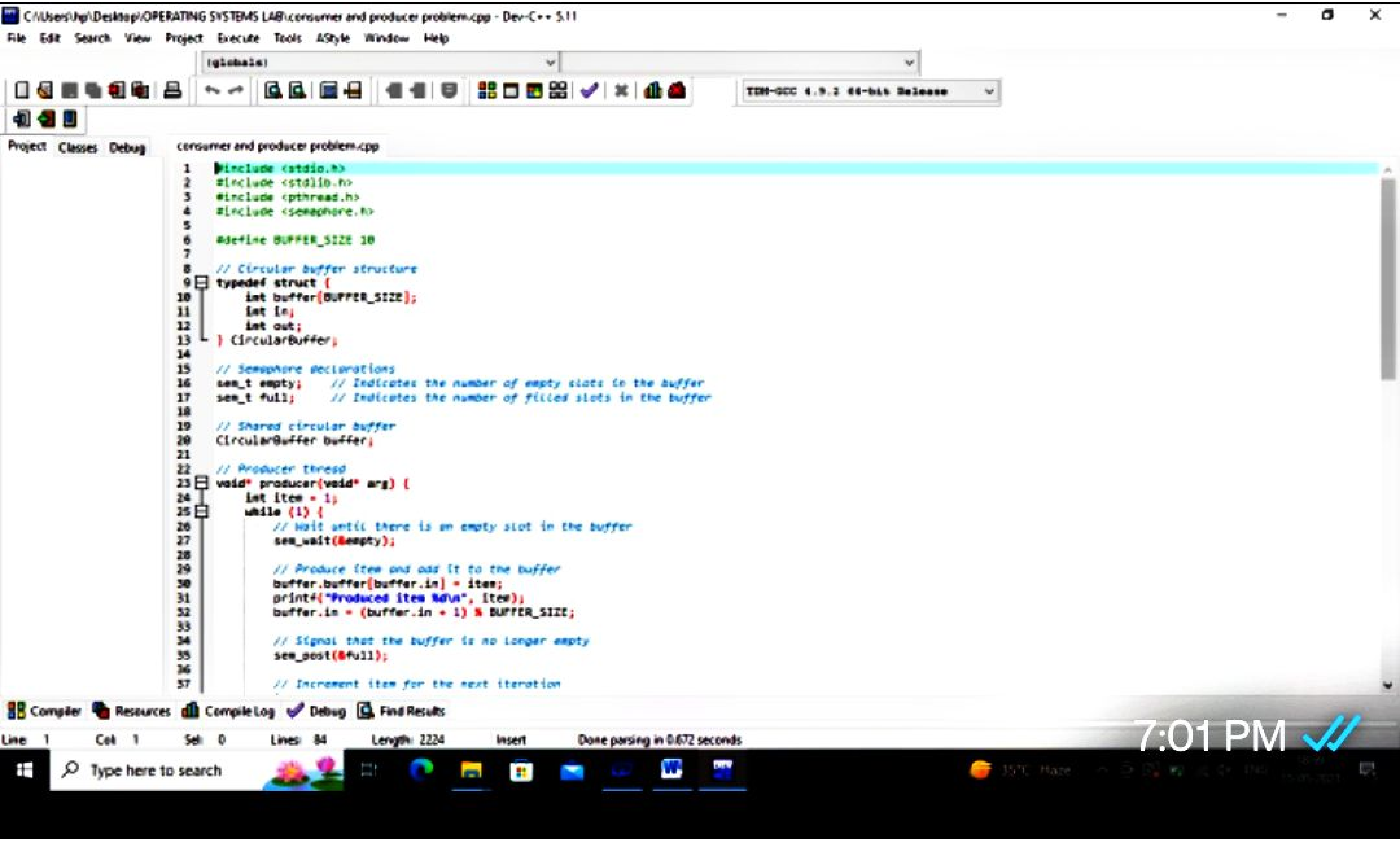
18. Write a program for solving the producer consumer problem with the following scenario: The producer should produce data only when the buffer is not full. Data can only be consumed by the consumer if and only if the memory buffer is not empty.

Test Case:

Buffer Size: 3

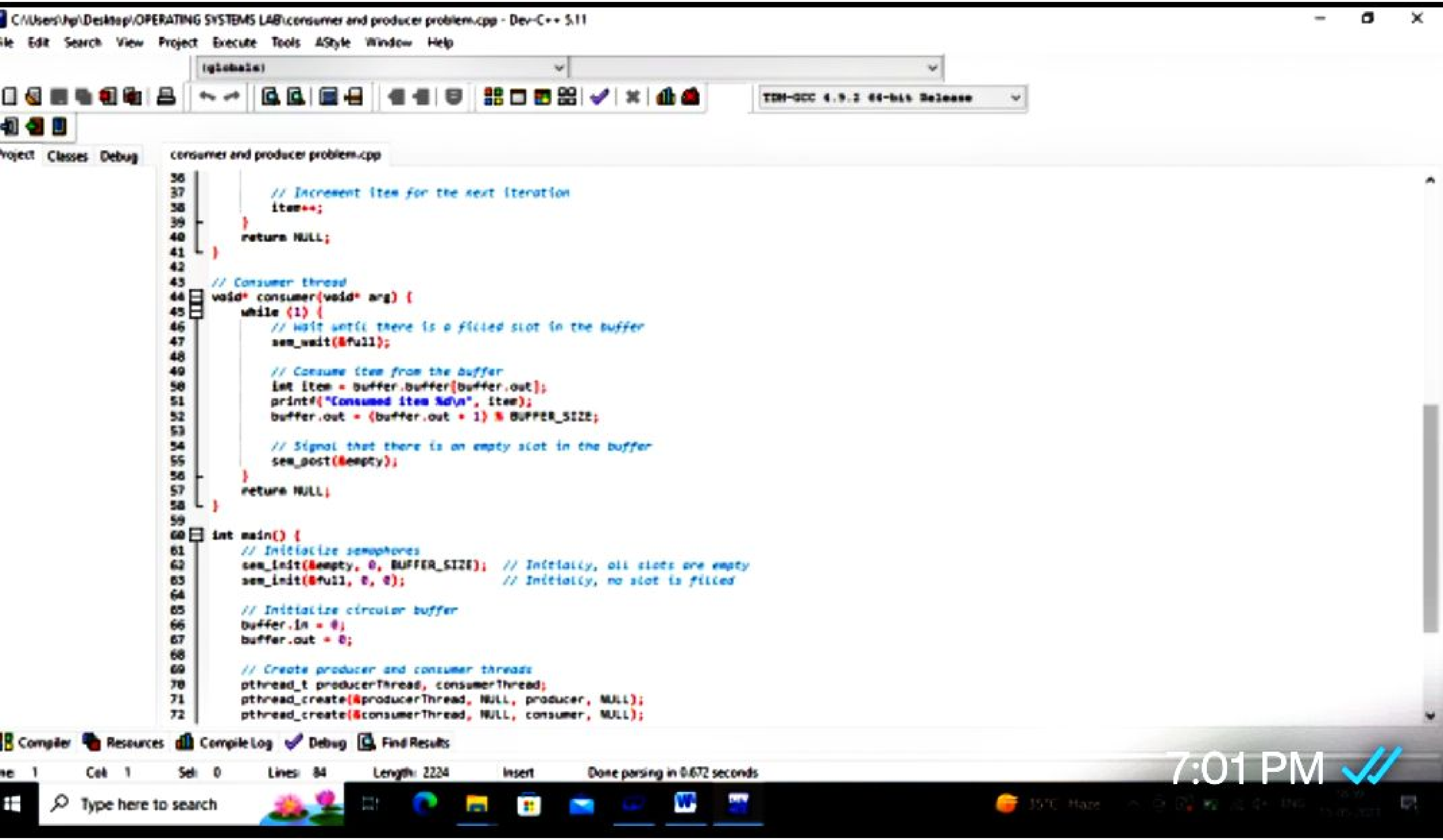
Consume an item in the beginning and show that the buffer is EMPTY

Produce 4 items and show that the buffer is FULL


C:\Users\hp\Desktop\OPERATING SYSTEMS LAB\consumer and producer problem.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
TDM-GCC 4.9.2 64-bit Release
Project Classes Debug
consumer and producer problem.cpp
Compiler Resources Compile Log Debug Find Results
7:01 PM
Type here to search

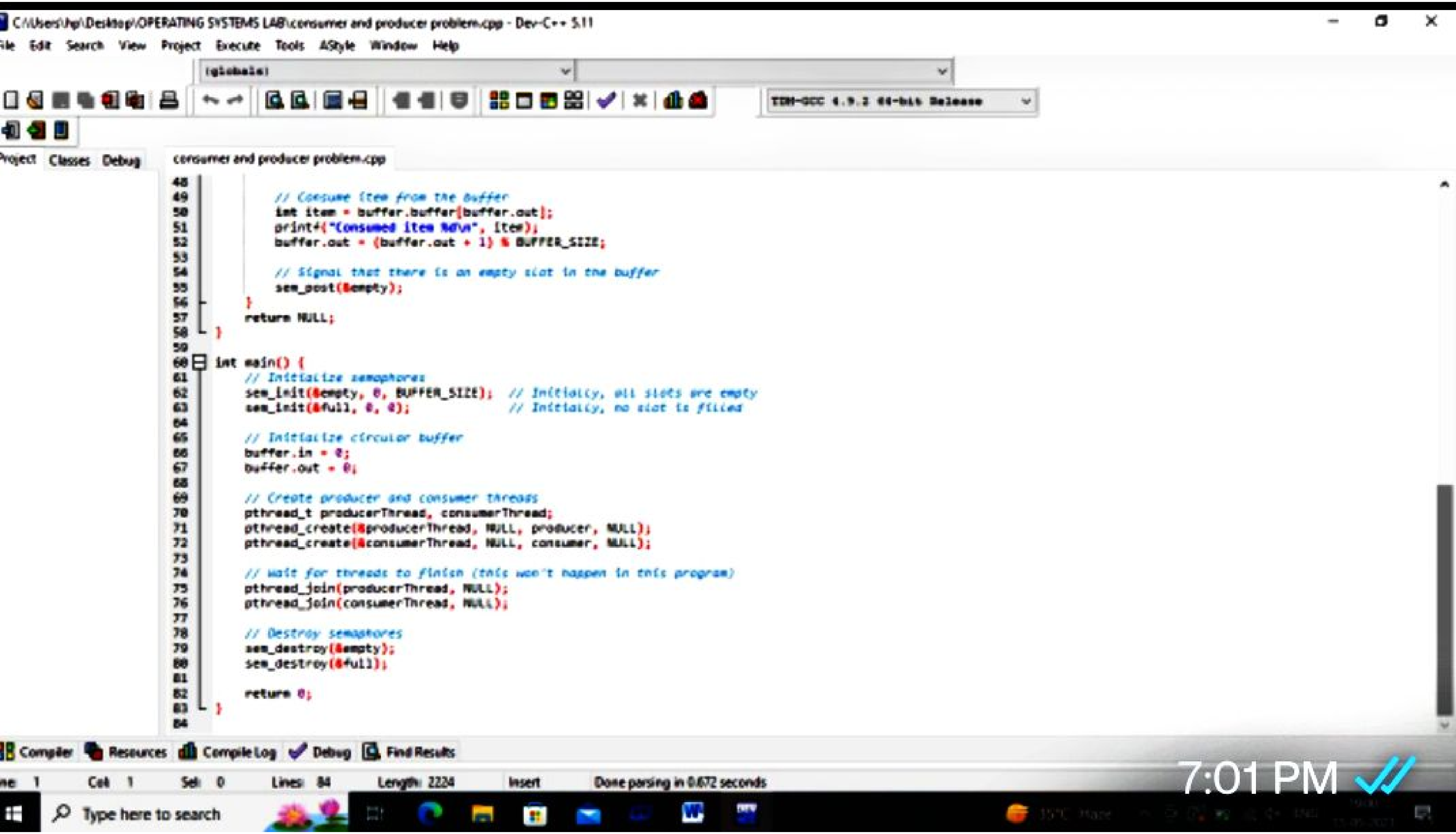
```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <semaphore.h>

#define BUFFER_SIZE 10

// Circular buffer structure
typedef struct {
    int buffer[BUFFER_SIZE];
    int in;
    int out;
} CircularBuffer;

// Semaphore declarations
sem_t empty;    // Indicates the number of empty slots in the buffer
sem_t full;     // Indicates the number of filled slots in the buffer

// Shared circular buffer
CircularBuffer buffer;

// Producer thread
void* producer(void* arg) {
    int item = 1;
    while (1) {
        // Wait until there is an empty slot in the buffer
        sem_wait(&empty);

        // Produce item and add it to the buffer
        buffer.buffer[buffer.in] = item;
        printf("Produced item %d\n", item);
        buffer.in = (buffer.in + 1) % BUFFER_SIZE;

        // Signal that the buffer is no longer empty
        sem_post(&full);

        // Increment item for the next iteration
```

Line: 1 Col: 1 Sel: 0 Lines: 84 Length: 2224 Insert Done parsing in 0.672 seconds


C:\Users\hp\Desktop\OPERATING SYSTEMS LAB\consumer and producer problem.cpp - Dev-C++ 5.11
ile Edit Search View Project Execute Tools AStyle Window Help
(globals)
TDM-GCC 4.9.2 64-bit Release
roject Classes Debug
consumer and producer problem.cpp
Compiler Resources Compile Log Debug Find Results
Type here to search
7:01 PM

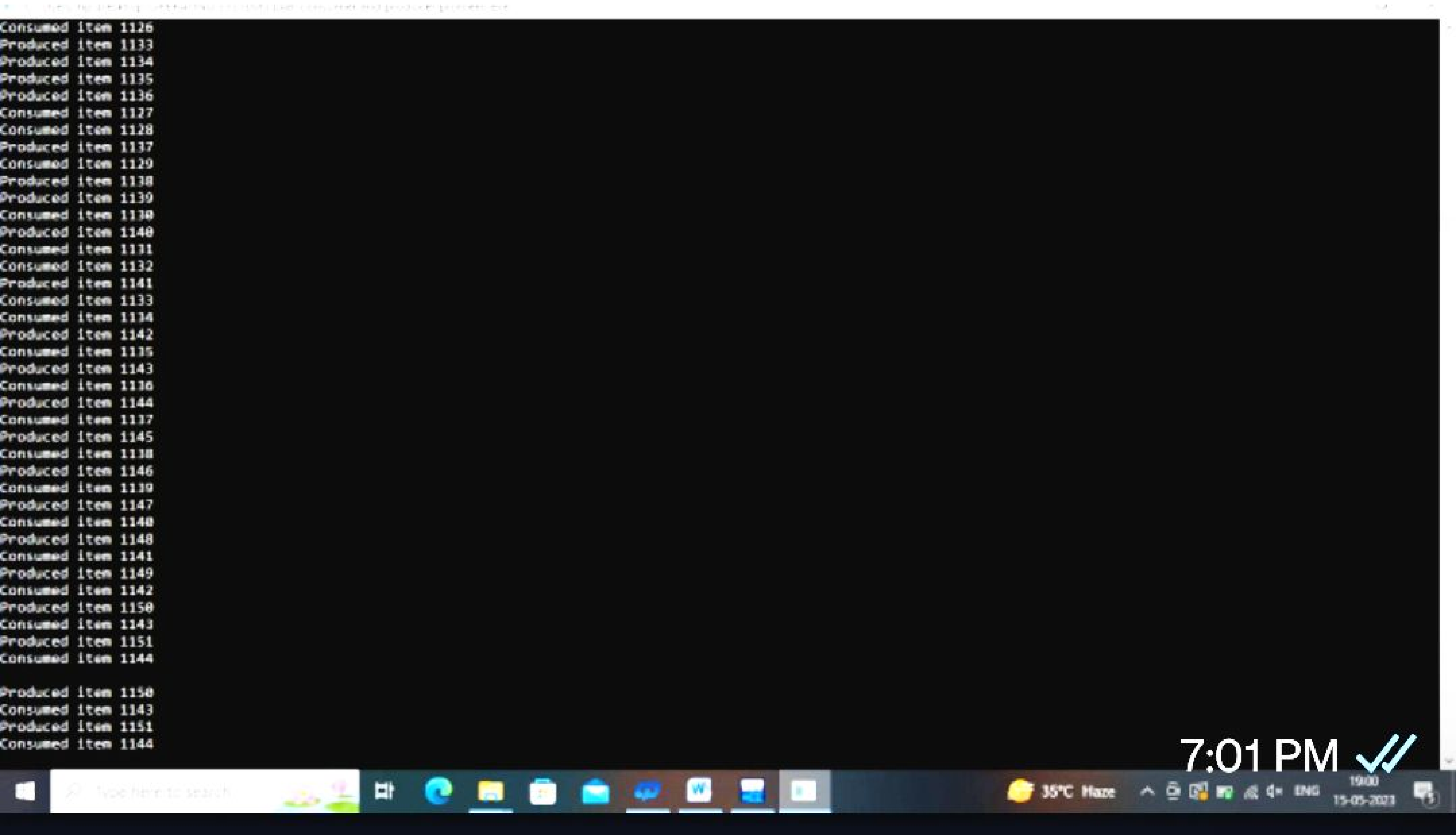
```
            // Increment item for the next iteration
            item++;
        }
        return NULL;
}

// Consumer thread
void* consumer(void* arg) {
        while (1) {
            // Wait until there is a filled slot in the buffer
            sem_wait(&full);

            // Consume item from the buffer
            int item = buffer.buffer[buffer.out];
            printf("Consumed item %d\n", item);
            buffer.out = (buffer.out + 1) % BUFFER_SIZE;

            // Signal that there is an empty slot in the buffer
            sem_post(&empty);
        }
        return NULL;
}

int main() {
        // Initialize semaphores
        sem_init(&empty, 0, BUFFER_SIZE);  // Initially, all slots are empty
        sem_init(&full, 0, 0);             // Initially, no slot is filled

        // Initialize circular buffer
        buffer.in = 0;
        buffer.out = 0;

        // Create producer and consumer threads
        pthread_t producerThread, consumerThread;
        pthread_create(&producerThread, NULL, producer, NULL);
        pthread_create(&consumerThread, NULL, consumer, NULL);
```

ne 1 Col 1 Sel 0 Lines 84 Length 2224 Insert Done parsing in 0.672 seconds


C:\Users\hp\Desktop\OPERATING SYSTEMS LAB\consumer and producer problem.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
TDM-GCC 4.9.2 64-bit Release
Project Classes Debug
consumer and producer problem.cpp
Compiler Resources Compile Log Debug Find Results
7:01 PM
Type here to search

```
            // Consume item from the buffer
            int item = buffer.buffer[buffer.out];
            printf("Consumed item %d\n", item);
            buffer.out = (buffer.out + 1) % BUFFER_SIZE;

            // Signal that there is an empty slot in the buffer
            sem_post(&empty);
    }
    return NULL;
}

int main() {
    // Initialize semaphores
    sem_init(&empty, 0, BUFFER_SIZE);  // Initially, all slots are empty
    sem_init(&full, 0, 0);             // Initially, no slot is filled

    // Initialize circular buffer
    buffer.in = 0;
    buffer.out = 0;

    // Create producer and consumer threads
    pthread_t producerThread, consumerThread;
    pthread_create(&producerThread, NULL, producer, NULL);
    pthread_create(&consumerThread, NULL, consumer, NULL);

    // Wait for threads to finish (this won't happen in this program)
    pthread_join(producerThread, NULL);
    pthread_join(consumerThread, NULL);

    // Destroy semaphores
    sem_destroy(&empty);
    sem_destroy(&full);

    return 0;
}
```

Line: 1 Col: 1 Sel: 0 Lines: 84 Length: 2224 Insert Done parsing in 0.672 seconds



```
Consumed item 1126
Produced item 1133
Produced item 1134
Produced item 1135
Produced item 1136
Consumed item 1127
Consumed item 1128
Produced item 1137
Consumed item 1129
Produced item 1138
Produced item 1139
Consumed item 1130
Produced item 1140
Consumed item 1131
Consumed item 1132
Produced item 1141
Consumed item 1133
Consumed item 1134
Produced item 1142
Consumed item 1135
Produced item 1143
Consumed item 1136
Produced item 1144
Consumed item 1137
Produced item 1145
Consumed item 1138
Produced item 1146
Consumed item 1139
Produced item 1147
Consumed item 1140
Produced item 1148
Consumed item 1141
Produced item 1149
Consumed item 1142
Produced item 1150
Consumed item 1143
Produced item 1151
Consumed item 1144

Produced item 1150
Consumed item 1143
Produced item 1151
Consumed item 1144
```