**EXP NO:** 8 **PRODUCER CONSUMER USING SEMAPHORES**

**DATE:19/2/25**

**PROGRAM:**

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <semaphore.h>

#define BUFFER\_SIZE 5 // Define buffer size

int buffer[BUFFER\_SIZE]; // Shared buffer

int in = 0, out = 0; // Indices for producer and consumer

sem\_t empty, full, mutex; // Semaphores for synchronization

// Function for Producer

void\* producer(void\* param) {

int item;

for (int i = 0; i < 10; i++) {

item = i + 1; // Item produced

sem\_wait(&empty); // Wait for an empty slot

sem\_wait(&mutex); // Enter critical section

// Produce an item and add it to the buffer

buffer[in] = item;

printf("Producer produces the item %d\n", item);

in = (in + 1) % BUFFER\_SIZE; // Update the index for the next producer

sem\_post(&mutex); // Exit critical section

sem\_post(&full); // Signal that a new item is added to the buffer

sleep(1); // Simulate time taken to produce

}

pthread\_exit(0); // Exit the producer thread

}

// Function for Consumer

void\* consumer(void\* param) {

int item;

for (int i = 0; i < 10; i++) {

sem\_wait(&full); // Wait for an item to consume

sem\_wait(&mutex); // Enter critical section

// Consume an item from the buffer

item = buffer[out];

printf("Consumer consumes item %d\n", item);

out = (out + 1) % BUFFER\_SIZE; // Update the index for the next consumer

sem\_post(&mutex); // Exit critical section

sem\_post(&empty); // Signal that a space is available in the buffer

sleep(1); // Simulate time taken to consume

}

pthread\_exit(0); // Exit the consumer thread

}

int main() {

pthread\_t producer\_thread, consumer\_thread;

// Initialize semaphores

sem\_init(&empty, 0, BUFFER\_SIZE); // Initially, all buffer slots are empty

sem\_init(&full, 0, 0); // Initially, no items are in the buffer

sem\_init(&mutex, 0, 1); // Mutex to ensure mutual exclusion

// Create producer and consumer threads

pthread\_create(&producer\_thread, NULL, producer, NULL);

pthread\_create(&consumer\_thread, NULL, consumer, NULL);

// Wait for threads to finish

pthread\_join(producer\_thread, NULL);

pthread\_join(consumer\_thread, NULL);

// Destroy semaphores

sem\_destroy(&empty);

sem\_destroy(&full);

sem\_destroy(&mutex);

printf("Producer and Consumer have finished\n");

return 0;

}

**OUTPUT:**

Producer produces the item 1

Consumer consumes item 1

Producer produces the item 2

Consumer consumes item 2

Producer produces the item 3

Consumer consumes item 3

Producer produces the item 4

Consumer consumes item 4

Producer produces the item 5

Consumer consumes item 5

Producer produces the item 6

Consumer consumes item 6

Producer produces the item 7

Consumer consumes item 7

Producer produces the item 8

Consumer consumes item 8

Producer produces the item 9

Consumer consumes item 9

Producer produces the item 10

Consumer consumes item 10

Producer and Consumer have finished