Assignment 4 (A)

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

#define BUFFER_SIZE 5  // Size of the buffer

int buffer[BUFFER_SIZE];  // Shared buffer
int in = 0;               // Index for the producer to insert an
item
int out = 0;              // Index for the consumer to remove an
item

sem_t empty;              // Semaphore for counting empty slots
sem_t full;               // Semaphore for counting full slots
pthread_mutex_t mutex;    // Mutex to ensure mutual exclusion

// Function to display the contents of the buffer
void print_buffer() {
    printf("Buffer: [");
    for(int i = 0; i < BUFFER_SIZE; i++) {
        if (i == BUFFER_SIZE - 1)
            printf("%d", buffer[i]);
        else
            printf("%d, ", buffer[i]);
    }
    printf("]\n");
}

// Function executed by the producer thread
void *producer(void *param) {
```

```
    int item;
    for(int i = 0; i < 10; i++) {
        item = rand() % 100;  // Produce an item (random number
between 0 and 99)
        sem_wait(&empty);     // Decrement empty semaphore
        pthread_mutex_lock(&mutex);  // Acquire the lock

        // Critical section: Add item to buffer
        buffer[in] = item;
        printf("Producer produced: %d at index %d\n", item, in);
        in = (in + 1) % BUFFER_SIZE;

        // Display the buffer after producing an item
        print_buffer();

        pthread_mutex_unlock(&mutex);  // Release the lock
        sem_post(&full);    // Increment full semaphore
    }
    pthread_exit(0);
}

// Function executed by the consumer thread
void *consumer(void *param) {
    int item;
    for(int i = 0; i < 10; i++) {
        sem_wait(&full);     // Decrement full semaphore
        pthread_mutex_lock(&mutex);  // Acquire the lock

        // Critical section: Remove item from buffer
        item = buffer[out];
        printf("Consumer consumed: %d from index %d\n", item, out);
        buffer[out] = 0; // Optional: Reset consumed buffer index to
0 for visualization
```

```
        out = (out + 1) % BUFFER_SIZE;

        // Display the buffer after consuming an item
        print_buffer();

        pthread_mutex_unlock(&mutex);  // Release the lock
        sem_post(&empty);    // Increment empty semaphore
    }
    pthread_exit(0);
}

int main() {
    pthread_t prod_tid, cons_tid;

    // Initialize the semaphores and mutex
    sem_init(&empty, 0, BUFFER_SIZE);
    sem_init(&full, 0, 0);
    pthread_mutex_init(&mutex, NULL);

    // Initialize buffer with 0s
    for(int i = 0; i < BUFFER_SIZE; i++)
        buffer[i] = 0;

    // Create producer and consumer threads
    pthread_create(&prod_tid, NULL, producer, NULL);
    pthread_create(&cons_tid, NULL, consumer, NULL);

    // Wait for threads to complete
    pthread_join(prod_tid, NULL);
    pthread_join(cons_tid, NULL);

    // Destroy the semaphores and mutex
```

```
    sem_destroy(&empty);
    sem_destroy(&full);
    pthread_mutex_destroy(&mutex);

    return 0;
}
```

```
Output :
Producer produced: 83 at index 0
Buffer: [83, 0, 0, 0, 0]
Producer produced: 86 at index 1
Buffer: [83, 86, 0, 0, 0]
Producer produced: 77 at index 2
Buffer: [83, 86, 77, 0, 0]
Producer produced: 15 at index 3
Buffer: [83, 86, 77, 15, 0]
Producer produced: 93 at index 4
Buffer: [83, 86, 77, 15, 93]
Consumer consumed: 83 from index 0
Buffer: [0, 86, 77, 15, 93]
Consumer consumed: 86 from index 1
Buffer: [0, 0, 77, 15, 93]
Consumer consumed: 77 from index 2
Buffer: [0, 0, 0, 15, 93]
Consumer consumed: 15 from index 3
Buffer: [0, 0, 0, 0, 93]
Consumer consumed: 93 from index 4
Buffer: [0, 0, 0, 0, 0]
Producer produced: 35 at index 0
Buffer: [35, 0, 0, 0, 0]
Producer produced: 86 at index 1
Buffer: [35, 86, 0, 0, 0]
```

```
Producer produced: 92 at index 2
Buffer: [35, 86, 92, 0, 0]
Producer produced: 49 at index 3
Buffer: [35, 86, 92, 49, 0]
Producer produced: 21 at index 4
Buffer: [35, 86, 92, 49, 21]
Consumer consumed: 35 from index 0
Buffer: [0, 86, 92, 49, 21]
Consumer consumed: 86 from index 1
Buffer: [0, 0, 92, 49, 21]
Consumer consumed: 92 from index 2
Buffer: [0, 0, 0, 49, 21]
Consumer consumed: 49 from index 3
Buffer: [0, 0, 0, 0, 21]
Consumer consumed: 21 from index 4
Buffer: [0, 0, 0, 0, 0]
```