Ex. No.: 8
Date: 27.3.25

PRODUCER CONSUMER USING SEMAPHORES

Aim: To write a program to implement solution to producer consumer problem using semaphores.

Algorithm:

1. Initialize semaphore empty, full and mutex.
2. Create two threads- producer thread and consumer thread.
3. Wait for target thread termination.
4. Call sem_wait on empty semaphore followed by mutex semaphore before entry into critical section.
5. Produce/Consume the item in critical section.
6. Call sem_post on mutex semaphore followed by full semaphore
7. before exiting critical section.
8. Allow the other thread to enter its critical section.
9. Terminate after looping ten times in producer and consumer Threads each.

Program Code:

```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>
#define BUFFER_SIZE 3
int buffer [Buffer_SIZE];
int n=0; out=0;
sem_t empty, full;
P_thread_mutex_t mutex;
void produce()
{
```

```
if (sem_trywait (&empty) != 0)
    {
      printf ("Buffer is Full! \n");
      return;
    }
pthread_mutex_lock (&mutex);
buffer [in] = int i;
printf ("producer produces the item %d \n",
                                buffer [n]);
in= (int i) %. BUFFER_SIZE;
pthread_mutex_unlock (&mutex);
sem_post(&full);
}
void consumer()
{
  if (sem_trywait (&full) != 0)
    {
      printf ("Buffer is empty \n");
      return
    }
  pthread_mutex_lock (&mutex);
  printf ("consumer consumes item \n");
  out= (out+1) %. BUFFER_SIZE;
  pthread_mutex_unlock (&mutex);
  sem_post (&empty);
}
int main()
{
  int choices;
  sem_init (&empty ,0, BUFFER_SIZE);
  sem_init (&full ,0, 0);
```

```
pthread_mutex_init (&mutex, NULL);
while (1)
{
    printf ("\n 1. Produces \n2. consumer \n3. Exit\n");
    printf ("Enter the choice:")
    scanf ("%d", &choice);
    switch (choice)
    {
        case 1:
            produce ();
            break;
        case 2:
            consumer ();
            break;
        case 3:
            sem_destroy (&empty);
            sem_destroy (&full);
            pthread_mutex_destroy (&mutex);
            return 0;
        default:
            printf ("Invalid choice! Try again\n");
    }
}
}
```

Sample Output:

1. Producer
2.Consumer
3.Exit
Enter your choice:1
Producer produces the item 1
Enter your choice:2
Consumer consumes item
1 Enter your choice:2
Buffer is empty!!
Enter your choice:1
Producer produces the item 1
Enter your choice:1
Producer produces the item 2
Enter your choice:1
Producer produces the item 3
Enter your choice:1
Buffer is full!!
Enter your choice:3

Result:

Thus the code for Producer - Consumer using semaphore is executed successfully.