Ex. No.: 8
Date: 29.03.2025

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 <stdlib.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>
#define BUFFER_SIZE 3
int buffer [BufferSIZE];
int count=0;
sem_t empty;
sem_t full;
pthread_mutex_t mutex.
```

```
void * producer (void *arg )
{
  Static int item = 1;
  Pthread_mutex_loc (& mutex):
   if (count = = BUFFER_SIZE )
    { print ("Buffer is full!" \n");
      pthread_mutex_unlock (& mutex);
      return NULL;
    }
  Pthread_mutex_unlock (& mutex);
  sem_wait (&empty);
  Pthread_mutex_lock/&mutex);
  buffer [count] = item;
  Printf("Producer Produces the item %d\n", item);
  item ++;
  count++;
  pthread_mutex_unlock (& mutex);
  sem_post (&full);
  return NULL;
  }
void *consumer (void*arg ){
  if (count == 0){
    printf (" Buffer is empty! \n");
    return NULL;
  }
  sem_wait (& full);
  Pthread_mutex_lock (&mutex);
  if(count>0){
    int item = buffer [count - 1];
    printf ("consumer consumes item %d\n",
              item);

    count --;
  }
  pthread_mutex_unlock (&mutex):
  sem_past (& empty);
  return null;
}
```

```
{
pthread_t prodthread, consthread;
int choice;
sem_init (&empty, 0, BUFFER_SIZE);
sem_init (&full, 0, 0);
pthread_mutex_init (&mutex, NULL);
while (1)
{
  printf ("\n 1. Producer\n 2. consumer \n 3. Exit
          \n Enter your choice : ");
  scanf("%d", &choice);
  switch (choice)
  {
   case 1:
     pthread_create (&prodthread, NULL, produce
                     NULL);
     pthread_join (prodthread, NULL);
     break;
   case 2:
     pthread_create (&consthread, NULL,
                     consumer, NULL);
     pthread_join (consthread, NULL);
     break;
   case 3:
     printf ("Exiting ....\n");
     sem_destroy (&empty);
     sem_destroy (&full);
     pthread_mutex_destroy (&mutex);
     exit (0);
   default:
     printf (" Invalid choice !");
  }
}
return 0;
}
```

output:

1. producer
2. consumer
3. Exit

Enter your choice : 1
producer produces item 1
enter your choice : 2
consumer consumes item 1
enter your choice : 2
Buffer is empty!!
Enter your choice : 1
Producer produces item 2
enter your choice : 1
Producer produces item 3
Enter your choice : 1
producer produces item 4
Enter your choice : 1
Buffer is full!
Enter your choice : 3
Exiting

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:

Hence the program to implement solution to producer consumer problem using semaphores has been executed sucessfully.