Process Synchronization

**Objective:**

Write a C program to simulate producer-consumer problem using semaphores.

**TASK:**

**DESCRIPTION**

Producer-consumer problem, is a common paradigm for cooperating processes. A producer process produces information that is consumed by a consumer process. One solution to the producer-consumer problem uses shared memory. To allow producer and consumer processes to run concurrently, there must be available a buffer of items that can be filled by the producer and emptied by the consumer. This buffer will reside in a region of memory that is shared by the producer and consumer processes. A producer can produce one item while the consumer is consuming another item. The producer and consumer must be synchronized, so that the consumer does not try to consume an item that has not yet been produced.

**12.3PROGRAM**

#include<stdio.h>

void main()

{

int buffer[10], bufsize, in, out, produce, consume, choice=0; in = 0;

out = 0; bufsize = 10;

while(choice !=3)

{

printf(“\n1. Produce \t 2. Consume \t3. Exit”);

printf(“\nEnter your choice: ”);

scanf(“%d”, &choice);

switch(choice) {

case 1: if((in+1)%bufsize==out)

printf(“\nBuffer is Full”);

else

{

printf(“\nEnter the value: “);

scanf(“%d”, &produce);

buffer[in] = produce;

in = (in+1)%bufsize;

}

Break; case 2:

if(in == out)

printf(“\nBuffer is Empty”);

else

{

consume = buffer[out];

printf(“\nThe consumed value is %d”, consume);

out = (out+1)%bufsize;

|  |  |  |
| --- | --- | --- |
|  |  | } |
|  |  | break; |
| } } } |  |  |
| ***OUTPUT*** |  |  |
| 1. Produce | 2. Consume | 3. Exit |
| Enter your choice: 2 | |  |
| Buffer is Empty |  |  |
| 1. Produce | 2. Consume | 3. Exit |
| Enter your choice: 1 | |  |
| Enter the value: 100 | |  |
| 1. Produce | 2. Consume | 3. Exit |
| Enter your choice: 2 | |  |
| The consumed value is 100 | |  |
| 1. Produce | 2. Consume | 3. Exit |

Enter your choice: 3