VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY
Permanently Affiliated to JNTU Kakinada, Approved by AICTE
Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified
Nambur, Pedakakani (M), Guntur (Dt) - 522508
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
B.Tech Program is Accredited by NBA

CERTIFICATE

Name of the Lab : OPERATING SYSTEMS
Name of the Student : Thota Nagababu
Student Regd. No. : 18BQ1A05K3
CLASS : III B.TECH. I SEM CSE – D
GIT HUB LINK:
https://github.com/nagababuthota984/5K3-OS-LAB

INDEX

S. No.	Name of the Experiment	LAST DATE	PAGE NO
6	Write a C program that illustrates two processes communicating using shared memory	JAN 5	3-7

Experiment 6

AIM: Write a C program that illustrates two processes communicating using shared memory

DESCRIPTION:

Producer consumer problem is also known as bounded buffer problem. In this problem we have two processes, producer and consumer, who share a fixed size buffer. Producer work is to produce data or items and put in buffer. Consumer work is to remove data from buffer and consume it. We have to make sure that producer do not produce data when buffer is full and consumer do not remove data when buffer is empty.

The producer should go to sleep when buffer is full. Next time when consumer removes data it notifies the producer and producer starts producing data again. The consumer should go to sleep when buffer is empty. Next time when producer add data it notifies the consumer and consumer starts consuming data. This solution can be achieved using semaphores.

SYNTAX:

i) int signal(**int**) :

signal is an user-defined method. It will take an integer as parameter and returns an integer.

ii) **int** wait(**int**):

wait is also an user-defined method. It will take an integer as parameter and returns an integer.

Iii) void producer():

Producer method will try to produce data. It will only produce if the mutex is equal to one and the buffer is not full. While producing the data, mutex will be made 0, so that consumer will not try to consume data before producing has not been completed. After producing the data it will make mutex to 1.

iv) void consumer():

Consumer method will try to consume the data. It will only consume the data if the mutex is equal to one and the buffer is not empty. While consuming data, mutex will be made 0, so that producer will not try to produce data while consuming is going on.After consuming data it will make mutex to 1 again.

CODE:

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

int mutex=1,full=0,empty=3,x=0;

int main()
{
    int n;
    void producer();
    void consumer();
    int wait(int);
    int signal(int);
    printf("\n1.Producer\n2.Consumer\n3.Exit");
```

```
	while(1)
	{
		printf("\nEnter your choice:");
		scanf("%d",&n);
		switch(n)
		{
			case 1: if((mutex==1)&&(empty!
=0))
					producer();
				else
					printf("Buffer is full!!");
				break;
			case 2: if((mutex==1)&&(full!=0))
					consumer();
				else
					printf("Buffer is
empty!!");
				break;
			case 3:
				exit(0);
				break;
		}
	}

	return 0;
}

int wait(int s)
{
```

```
        return (--s);
}

int signal(int s)
{
        return(++s);
}

void producer()
{
        mutex=wait(mutex);
        full=signal(full);
        empty=wait(empty);
        x++;
        printf("\nProducer produces the item %d",x);
        mutex=signal(mutex);
}

void consumer()
{
        mutex=wait(mutex);
        full=wait(full);
        empty=signal(empty);
        printf("\nConsumer consumes item %d",x);
        x--;
        mutex=signal(mutex);
}
```

OUTPUT SCREENSHOTS:

```
nag-1211@hp:~/Documents/3-1/OS Lab/Linux Exp 6$ gcc exp6.c
nag-1211@hp:~/Documents/3-1/OS Lab/Linux Exp 6$ ./a.out

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:3
```