Experiment 5 **Surya Narayanan**

Producer-Consumer Problem

s5 csb 48

---

1. Algorithm

Step 1 : START

Step 2: include necessary headers.

Step 3: Initialize and declare necessary variables and functions

Step 4: Display menu to choose between producer and consumer

Step 5: Repeat the following till user choice is to continues

Step 5.1 : Get user choice

Step 5.2 : if choice = producer

Step 5.2.1 - check if mutex is 1 and buffer is empty if yes go to function producer().

Step 5.2.2 - Else display "Buffer is full.

Step 5.2.3 - Go Step 4.

Step 5.3: if choice = consumer

Step 5.3.1 - check if mutex is 1. and buffer is full: if yes, go to function consumer()

Step 5.3.2 - Else display "empty buffer".

Step 5.3.3 :- Go to step 4.

Step 5.4: if choice = Exit, go to step 6.

Step 6. STOP.

Producer ()

Step 1: START

Step 2. Set value of mutex as value obtained from function wait.

Step 3. Set value of full as obtained from function signal ()

Step 4. Set empty as value from function wait (empty)

Step 5. increment x by 1

Step 6. Display the message and set mutex value to that obtained from signal (mutex).

Step 7. STOP

Consumer ()

Step 1: START

Step 2: Set mutex value to that obtained from

Step 3: Set value of full to that obtained from wait (full)

Step 4: Display the message and decrement by 1

Step 5: Set new mutex value as signal (mutex)

Step 6: STOP.

Wait()

Step 1: start

Step 2: Decrement semaphore if $S \geq 0$

Step 3: Return the decremented value

Step 4: STOP.

Signal()

Step 1: Start

Step 2: Increment semaphore value

Step 3: Return the incremented value

Step 4: STOP.

Program

```
#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("PRODUCER-CONSUMER PROBLEM\n1)PRODUCER\n2)CONSUMER\n3)EXIT");

    while (1)
    {
        printf("\nCHOICE:");
        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("EMPTY BUFFER");
            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 %d", x);
    mutex = signal(mutex);
}
void consumer()
{
    mutex = wait(mutex);
    full = wait(full);
    empty = signal(empty);
    printf("\nCONSUMER CONSUMES %d", x);
    x--;
    mutex = signal(mutex);
}
```

Output


Ubuntu
/mnt/e/school/ss/5 06:38 PM > gcc exp5.c
/mnt/e/school/ss/5 06:38 PM > ./a.out
PRODUCER-CONSUMER PROBLEM
1)PRODUCER
2)CONSUMER
3)EXIT
CHOICE:1
PRODUCER PRODUCES 1
CHOICE:2
CONSUMER CONSUMES 1
CHOICE:1
PRODUCER PRODUCES 1
CHOICE:1
PRODUCER PRODUCES 2
CHOICE:1
PRODUCER PRODUCES 3
CHOICE:2
CONSUMER CONSUMES 3
CHOICE:2
CONSUMER CONSUMES 2
CHOICE:2
CONSUMER CONSUMES 1
CHOICE:2
EMPTY BUFFER
CHOICE:3
/mnt/e/school/ss/5 06:39 PM >
6:39 PM