UCS 1411 - Operating Systems Lab

Exercise 6 - Implementation of Producer/Consumer Problem using Semaphores

Mahesh Bharadwaj K - 185001089

1 To write a C program to create parent/child processes to implement the producer/consumer problem using semaphores in pthread library.

Program

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <semaphore.h>
#include <pthread.h> // for semaphore operations sem_init,sem_wait,sem_post
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/sem.h>
#include <sys/wait.h>
#include <sys/errno.h>
#include <sys/types.h>
extern int errno;
#define SIZE 10   /* size of the shared buffer*/
#define VARSIZE 1 /* size of shared variable=1byte*/
#define INPUTSIZE 20
#define SHMPERM 0666 /* shared memory permissions */
int segid;           /* id for shared memory bufer */
int empty_id;
int full_id;
int mutex_id;
char *buff;
char *input_string;
sem_t *empty;
sem_t *full;
sem_t *mutex;
int p = 0, c = 0;
//
// Producer function
//
void produce()
{
    int i = 0;
    while (1)
    {
        if (i >= strlen(input_string))
        {
            printf("\n Producer %d exited \n", getpid());
            wait(NULL);
            exit(1);
        }
        printf("\nProducer %d trying to aquire Semaphore Empty \n", getpid());
```

```
        sem_wait(empty);
        printf("\nProducer %d successfully aquired Semaphore Empty \n", getpid());
        printf("\nProducer %d trying to aquire Semaphore Mutex \n", getpid());
        sem_wait(mutex);
        printf("\nProducer %d successfully aquired Semaphore Mutex \n", getpid());
        buff[p] = input_string[i];
        printf("\nProducer %d Produced Item [ %c ] \n", getpid(), input_string[i])
    ;
        i++;
        p++;
        printf("\nItems in Buffer %d \n", p);
        sem_post(mutex);
        printf("\nProducer %d released Semaphore Mutex \n", getpid());
        sem_post(full);
        printf("\nProducer %d released Semaphore Full \n", getpid());
        sleep(2 / random());
    } //while
} //producer fn
//
// Consumer function
//
void consume()
{
    int i = 0;
    while (1)
    {
        if (i >= strlen(input_string))
        {
            printf("\n Consumer %d exited \n", getpid());
            exit(1);
        }
        printf("\nConsumer %d trying to aquire Semaphore Full \n", getpid());
        sem_wait(full);
        printf("\nConsumer %d successfully aquired Semaphore Full \n", getpid());
        printf("\nConsumer %d trying to aquire Semaphore Mutex \n", getpid());
        sem_wait(mutex);
        printf("\nConsumer %d successfully aquired Semaphore Mutex\n", getpid());
        printf("\nConsumer %d Consumed Item [ %c ] \n", getpid(), buff[c]);
        buff[c] = ' ';
        c++;
        printf("\nItems in Buffer %d \n", strlen(input_string) - c);
        i++;
        sem_post(mutex);
        printf("\nConsumer %d released Semaphore Mutex \n", getpid());
        sem_post(empty);
        printf("\nConsumer %d released Semaphore Empty \n", getpid());
        sleep(1);
    } //while
} //consumer fn


int main()
{
    int i = 0;
    pid_t temp_pid;
    segid = shmget(IPC_PRIVATE, SIZE, IPC_CREAT | IPC_EXCL | SHMPERM);
    empty_id = shmget(IPC_PRIVATE, sizeof(sem_t), IPC_CREAT | IPC_EXCL | SHMPERM);
    full_id = shmget(IPC_PRIVATE, sizeof(sem_t), IPC_CREAT | IPC_EXCL | SHMPERM);
    mutex_id = shmget(IPC_PRIVATE, sizeof(sem_t), IPC_CREAT | IPC_EXCL | SHMPERM);
    buff = shmat(segid, (char *)0, 0);
    empty = shmat(empty_id, (char *)0, 0);
```

```
    full = shmat(full_id, (char *)0, 0);
    mutex = shmat(mutex_id, (char *)0, 0);
    // Initializing Semaphores Empty , Full & Mutex
    sem_init(empty, 1, SIZE);
    sem_init(full, 1, 0);
    sem_init(mutex, 1, 1);
    printf("\n Main Process Started \n");
    printf("\n Enter the input string (20 characters MAX) : ");
    input_string = (char *)malloc(20);
    scanf("%s", input_string);
    printf("Entered string : %s", input_string);
    temp_pid = fork();
    if (temp_pid > 0) //parent
    {
        produce();
    }
    else //child
    {
        consume();
    }
    shmdt(buff);
    shmdt(empty);
    shmdt(full);
    shmdt(mutex);
    shmctl(segid, IPC_RMID, NULL);
    semctl(empty_id, 0, IPC_RMID, NULL);
    semctl(full_id, 0, IPC_RMID, NULL);
    semctl(mutex_id, 0, IPC_RMID, NULL);
    sem_destroy(empty);
    sem_destroy(full);
    sem_destroy(mutex);
    printf("\n Main process exited \n\n");
    return (0);
} //main
```

Output

Main Process Started

Enter the input string (20 characters MAX) : hello Entered string : hello Producer 16667 trying to aquire Semaphore Empty

Producer 16667 successfully aquired Semaphore Empty

Producer 16667 trying to aquire Semaphore Mutex

Producer 16667 successfully aquired Semaphore Mutex

Producer 16667 Produced Item [h]

Items in Buffer 1

Producer 16667 released Semaphore Mutex

Producer 16667 released Semaphore Full Entered string : hello Consumer 16679 trying to aquire Semaphore Full

Consumer 16679 successfully aquired Semaphore Full

Producer 16667 trying to aquire Semaphore Empty Consumer 16679 trying to aquire Semaphore Mutex

Producer 16667 successfully aquired Semaphore Empty

Consumer 16679 successfully aquired Semaphore Mutex Producer 16667 trying to aquire Semaphore Mutex

Consumer 16679 Consumed Item [h]

Items in Buffer 4

Consumer 16679 released Semaphore Mutex

Consumer 16679 released Semaphore Empty

Producer 16667 successfully aquired Semaphore Mutex

Producer 16667 Produced Item [e]
Items in Buffer 2
Producer 16667 released Semaphore Mutex
Producer 16667 released Semaphore Full
Producer 16667 trying to aquire Semaphore Empty
Producer 16667 successfully aquired Semaphore Empty
Producer 16667 trying to aquire Semaphore Mutex
Producer 16667 successfully aquired Semaphore Mutex
Producer 16667 Produced Item [l]
Items in Buffer 3
Producer 16667 released Semaphore Mutex
Producer 16667 released Semaphore Full
Producer 16667 trying to aquire Semaphore Empty
Producer 16667 successfully aquired Semaphore Empty
Producer 16667 trying to aquire Semaphore Mutex
Producer 16667 successfully aquired Semaphore Mutex
Producer 16667 Produced Item [l]
Items in Buffer 4
Producer 16667 released Semaphore Mutex
Producer 16667 released Semaphore Full
Producer 16667 trying to aquire Semaphore Empty
Producer 16667 successfully aquired Semaphore Empty
Producer 16667 trying to aquire Semaphore Mutex
Producer 16667 successfully aquired Semaphore Mutex
Producer 16667 Produced Item [o]
Items in Buffer 5
Producer 16667 released Semaphore Mutex
Producer 16667 released Semaphore Full
Producer 16667 exited
Consumer 16679 trying to aquire Semaphore Full
Consumer 16679 successfully aquired Semaphore Full
Consumer 16679 trying to aquire Semaphore Mutex
Consumer 16679 successfully aquired Semaphore Mutex
Consumer 16679 Consumed Item [e]
Items in Buffer 3
Consumer 16679 released Semaphore Mutex
Consumer 16679 released Semaphore Empty
Consumer 16679 trying to aquire Semaphore Full
Consumer 16679 successfully aquired Semaphore Full
Consumer 16679 trying to aquire Semaphore Mutex
Consumer 16679 successfully aquired Semaphore Mutex
Consumer 16679 Consumed Item [l]
Items in Buffer 2
Consumer 16679 released Semaphore Mutex
Consumer 16679 released Semaphore Empty
Consumer 16679 trying to aquire Semaphore Full
Consumer 16679 successfully aquired Semaphore Full
Consumer 16679 trying to aquire Semaphore Mutex
Consumer 16679 successfully aquired Semaphore Mutex
Consumer 16679 Consumed Item [l]
Items in Buffer 1
Consumer 16679 released Semaphore Mutex
Consumer 16679 released Semaphore Empty
Consumer 16679 trying to aquire Semaphore Full
Consumer 16679 successfully aquired Semaphore Full
Consumer 16679 trying to aquire Semaphore Mutex

Consumer 16679 successfully aquired Semaphore Mutex
Consumer 16679 Consumed Item [o]
Items in Buffer 0
Consumer 16679 released Semaphore Mutex
Consumer 16679 released Semaphore Empty
Consumer 16679 exited

---

2 Modify the program as separate client / server process programs to generate 'N' random numbers in producer and write them into shared memory. Consumer process should read them from shared memory and display them in terminal

Producer / Server Program

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <semaphore.h>
#include <pthread.h> // for semaphore operations sem_init,sem_wait,sem_post
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/sem.h>
#include <sys/wait.h>
#include <sys/errno.h>
#include <sys/types.h>
#include <math.h>

extern int errno;
#define true 1
#define SIZE 10  /* size of the shared buffer*/
#define VARSIZE 1 /* size of shared variable=1byte*/
#define INPUTSIZE 20
#define SHMPERM 0666 /* shared memory permissions */
int segid;           /* id for shared memory bufer */
int empty_id;
int full_id;
int mutex_id;
int *buff;
sem_t *empty;
sem_t *full;
sem_t *mutex;
int main()
{
    pid_t temp_pid;
    segid = shmget(315, SIZE * sizeof(int), IPC_CREAT | IPC_EXCL | SHMPERM);
    empty_id = shmget(316, sizeof(sem_t), IPC_CREAT | IPC_EXCL | SHMPERM);
    full_id = shmget(317, sizeof(sem_t), IPC_CREAT | IPC_EXCL | SHMPERM);
    mutex_id = shmget(318, sizeof(sem_t), IPC_CREAT | IPC_EXCL | SHMPERM);
    buff = shmat(segid, (char *)0, 0);
    empty = shmat(empty_id, (char *)0, 0);
    full = shmat(full_id, (char *)0, 0);
    mutex = shmat(mutex_id, (char *)0, 0);
    // Initializing Semaphores Empty , Full & Mutex
    sem_init(empty, 1, SIZE);
    sem_init(full, 1, 0);
    sem_init(mutex, 1, 1);
```

```
    printf("\nProducer / Server Process Started \n");

    sleep(2);

    unsigned int i = 0,p = 0;
    while (true)
    {
        if (i >= 10)
        {
            printf("\n Producer %d exited \n", getpid());
            wait(NULL);
            exit(1);
        }

        printf("\nProducer %d trying to aquire Semaphore Empty \n", getpid());
        sem_wait(empty);
        printf("\nProducer %d successfully aquired Semaphore Empty \n", getpid());

        printf("\nProducer %d trying to aquire Semaphore Mutex \n", getpid());
        sem_wait(mutex);
        printf("\nProducer %d successfully aquired Semaphore Mutex \n", getpid());

        buff[p] = (random() % 10 + 1);
        printf("\nProducer %d Produced Item [ %d ] \n", getpid(), buff[p]);
        i++;
        p++;
        printf("\nItems in Buffer %d \n", p);
        sem_post(mutex);
        printf("\nProducer %d released Semaphore Mutex \n", getpid());
        sem_post(full);
        printf("\nProducer %d released Semaphore Full \n", getpid());
        sleep(1);
    } //while



    shmdt(buff);
    shmdt(empty);
    shmdt(full);
    shmdt(mutex);

    printf("\nProducer / Server Process Exited \n\n");
    return (0);
} //main
```

Consumer / Client Program

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <semaphore.h>
#include <pthread.h> // for semaphore operations sem_init,sem_wait,sem_post
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/sem.h>
#include <sys/wait.h>
#include <sys/errno.h>
#include <sys/types.h>
extern int errno;
```

```c
#define true 1
#define SIZE 10   /* size of the shared buffer*/
#define VARSIZE 1 /* size of shared variable=1byte*/
#define INPUTSIZE 20
#define SHMPERM 0666 /* shared memory permissions */
int segid;           /* id for shared memory bufer */
int empty_id;
int full_id;
int mutex_id;
int *buff;
sem_t *empty;
sem_t *full;
sem_t *mutex;

int main()
{
    pid_t temp_pid;
    segid = shmget(315, SIZE * sizeof(int), IPC_EXCL | SHMPERM);
    empty_id = shmget(316, sizeof(sem_t),  IPC_EXCL | SHMPERM);
    full_id = shmget(317, sizeof(sem_t), IPC_EXCL | SHMPERM);
    mutex_id = shmget(318, sizeof(sem_t),  IPC_EXCL | SHMPERM);
    buff = shmat(segid, (char *)0, 0);
    empty = shmat(empty_id, (char *)0, 0);
    full = shmat(full_id, (char *)0, 0);
    mutex = shmat(mutex_id, (char *)0, 0);

    printf("\nConsumer / Client Process Started \n");

    sleep(2);

    unsigned int i = 0,c = 0;
    while (true)
    {
        if (i >= 10)
        {
            printf("\nConsumer %d exited \n", getpid());
            exit(1);
        }

        printf("\nConsumer %d trying to aquire Semaphore Full \n", getpid());
        sem_wait(full);
        printf("\nConsumer %d successfully aquired Semaphore Full \n", getpid());
        printf("\nConsumer %d trying to aquire Semaphore Mutex \n", getpid());
        sem_wait(mutex);
        printf("\nConsumer %d successfully aquired Semaphore Mutex\n", getpid());
        printf("\nConsumer %d Consumed Item [ %d ] \n", getpid(), buff[c]);
        buff[c] = -1;
        c++;
        printf("\nItems in Buffer %d \n", SIZE - c);
        i++;
        sem_post(mutex);
        printf("\nConsumer %d released Semaphore Mutex \n", getpid());
        sem_post(empty);
        printf("\nConsumer %d released Semaphore Empty \n", getpid());
        sleep(1);
    } //while

    sleep(2);
    shmdt(buff);
    shmdt(empty);
    shmdt(full);
```

```
    shmdt(mutex);
    shmctl(segid, IPC_RMID, NULL);
    semctl(empty_id, 0, IPC_RMID, NULL);
    semctl(full_id, 0, IPC_RMID, NULL);
    semctl(mutex_id, 0, IPC_RMID, NULL);
    sem_destroy(empty);
    sem_destroy(full);
    sem_destroy(mutex);
    printf("\nConsumer / Client process exited \n\n");
    return (0);
} //main
```

Output

The Server & Client Programs were executed simultaneously

Producer Process Output

Producer / Server Process Started
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [4]
Items in Buffer 1
Producer 5048 released Semaphore Mutex
Producer 5048 released Semaphore Full
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [7]
Items in Buffer 2
Producer 5048 released Semaphore Mutex
Producer 5048 released Semaphore Full
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [8]
Items in Buffer 3
Producer 5048 released Semaphore Mutex
Producer 5048 released Semaphore Full
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [6]
Items in Buffer 4
Producer 5048 released Semaphore Mutex
Producer 5048 released Semaphore Full
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [4]
Items in Buffer 5
Producer 5048 released Semaphore Mutex

Producer 5048 released Semaphore Full
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [6]
Items in Buffer 6
Producer 5048 released Semaphore Mutex
Producer 5048 released Semaphore Full
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [7]
Items in Buffer 7
Producer 5048 released Semaphore Mutex
Producer 5048 released Semaphore Full
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [3]
Items in Buffer 8
Producer 5048 released Semaphore Mutex
Producer 5048 released Semaphore Full
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [10]
Items in Buffer 9
Producer 5048 released Semaphore Mutex
Producer 5048 released Semaphore Full
Producer 5048 trying to aquire Semaphore Empty
Producer 5048 successfully aquired Semaphore Empty
Producer 5048 trying to aquire Semaphore Mutex
Producer 5048 successfully aquired Semaphore Mutex
Producer 5048 Produced Item [2]
Items in Buffer 10
Producer 5048 released Semaphore Mutex
Producer 5048 released Semaphore Full
Producer 5048 exited

Consumer Process Output

Consumer / Client Process Started
Consumer 5213 trying to aquire Semaphore Full
Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [4]
Items in Buffer 9
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 trying to aquire Semaphore Full

Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [7]
Items in Buffer 8
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 trying to aquire Semaphore Full
Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [8]
Items in Buffer 7
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 trying to aquire Semaphore Full
Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [6]
Items in Buffer 6
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 trying to aquire Semaphore Full
Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [4]
Items in Buffer 5
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 trying to aquire Semaphore Full
Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [6]
Items in Buffer 4
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 trying to aquire Semaphore Full
Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [7]
Items in Buffer 3
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 trying to aquire Semaphore Full
Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [3]
Items in Buffer 2
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 trying to aquire Semaphore Full

Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [10]
Items in Buffer 1
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 trying to aquire Semaphore Full
Consumer 5213 successfully aquired Semaphore Full
Consumer 5213 trying to aquire Semaphore Mutex
Consumer 5213 successfully aquired Semaphore Mutex
Consumer 5213 Consumed Item [2]
Items in Buffer 0
Consumer 5213 released Semaphore Mutex
Consumer 5213 released Semaphore Empty
Consumer 5213 exited
