OS ASSIGNMENT 8

Aryan Sharma
20233083

OS Assignment 8

Aryan Sharma

2023308 3

Q1 →

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/wait.h>
#include <time.h>

void child(int* shared_var, int value){
    for(int i=0;i<5;i++){
        int temp= *shared_var;
        usleep(rand() % 100000);
        temp+=value;
        *shared_var = temp;
    }
    exit(0);
}

int main(){
    srand(time(NULL));
    int shmid = shmget(IPC_private, sizeof(int), IPC_CREAT|0666);
    int *shared_var = (int*)shmat(shmid, NULL, 0);
    *shared_var=1000;
    if(fork()==0) child(shared_var, 200);
    if(fork()==0) child(shared_var, 100);
```

Aryan Sharma
2023503

```
wait(NULL);
printf("Final value: %d\n", *shared_var);
shmdt(shared_var);
shmctl(shmid, IPC_RMID, NULL);
return 0;
}
```

Q2.)

a)

```
#include<pthread.h>
#include<semaphore.h>
#include<stdio.h>
#include<unistd.h>

#define BUF_SIZE 10
int buffer[BUF_SIZE][2], in=0, out=0;
sem_t empty, full, mutex;

void *producer(void *arg){
    int id= *(int*)arg;
    for(int i=0; i<20; i++){
        sem_wait(&empty);
        sem_wait(&mutex);
        buffer[in][0]= id;
        buffer[in][1] = 1;
        in = (in+1)% BUF_SIZE;
```

Aryan Sharma
2023083

```
        sem_post(&mutex);
        sem_post(&full);
    }
    return NULL;
}

void *consumer(void *arg){
    int id = *(int*)arg;
    for(int i=0; i<10; i++){
        sem_wait(&full);
        sem_wait(&mutex);
        int pid= buffer[out][0], val= buffer[out][1];
        out = (out+1) % BUF_SIZE;
        sem_post(&mutex);
        sem_post(&empty);
        printf("Consumer %d read: (%d,%d)\n", id, pid, val);
    }
    return NULL;
}

int main(){
    pthread_t prod[5], cons[10];
    sem_init(&empty, 0, BUF_SIZE);
    sem_init(&full, 0, 0);
    sem_init(&mutex, 0, 1);
```

Aryan Sharma
2023083

```
int ids[15];
for (int i=0; i<5; i++){
    ids[i]= i;
    pthread_create(&prod[i], NULL, producer, &ids[i]);
}
for(int i=0; i<10; i++){
    ids[i+5]=i;
    pthread_create(&cons[i], NULL, consumer, &ids[i+5]);
}
for(int i=0; i<5; i++) pthread_join(prod[i], NULL);
for(int i=0; i<10; i++) pthread_join(cons[i], NULL);
return 0;
}
```

Q3)

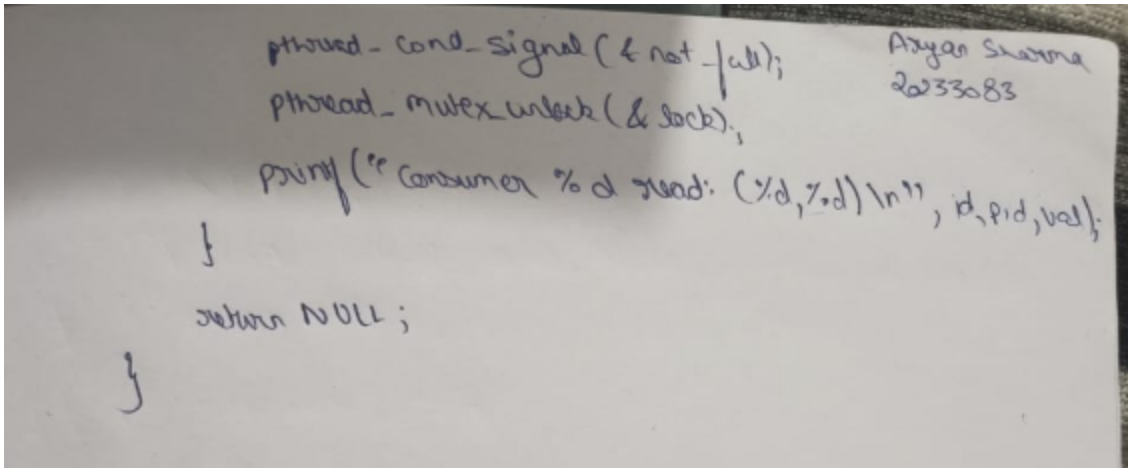
```
#include <pthread.h>
#include <stdio.h>
#define BUF_SIZE 10
int buffer[BUF_SIZE][2], in=0, out=0, count=0;
pthread_mutex_t lock= PTHREAD_MUTEX_INITIALIZER;
pthread_cond_t not_full= PTHREAD_COND_INITIALISER.
pthread_cond_t not_empty= PTHREAD_COND_INITIALIZER.
```

Aryan Sharma
202353265

```
void *producer(void *arg){
    int id = *(int *)arg;
    for(int i = 0; i<20; i++){
        pthread_mutex_lock(&lock);
        while(count == BUF_SIZE) pthread_cond_wait(&not_full,
                                                   &lock);

        buffer[in][0] = id;
        buffer[in][1] = i;
        in = (in+1) % BUF_SIZE; count++;
        pthread_cond_signal(&not_empty);
        pthread_mutex_unlock(&lock);
    }
    return NULL;
}

void *consumer(void *arg){
    int id = *(int *)arg;
    for(int i=0; i<20; i++){
        pthread_mutex_lock(&lock);
        while(count==0) pthread_cond_wait(&not_empty, &lock
        int pid = buffer[out][0], val = buffer[out][1];
        out = (out+1) % BUF_SIZE;
        count--;
```



```
        pthread_cond_signal(&not_full);
        pthread_mutex_unlock(&lock);
        printf("Consumer %d read: (%d, %d)\n", id, pid, val);
    }
    return NULL;
}
```

Aryan Saxena
2233083