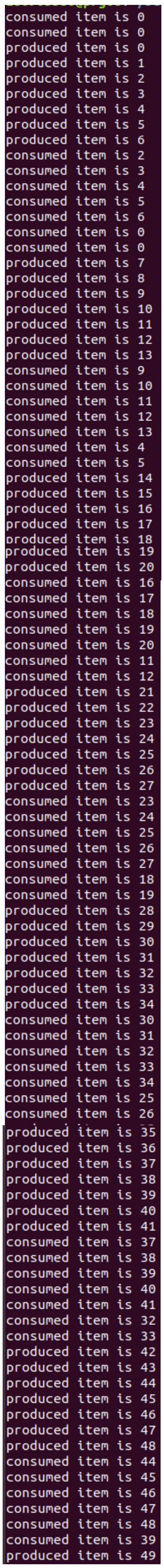
Q1. Extra-10 Producer Consumer

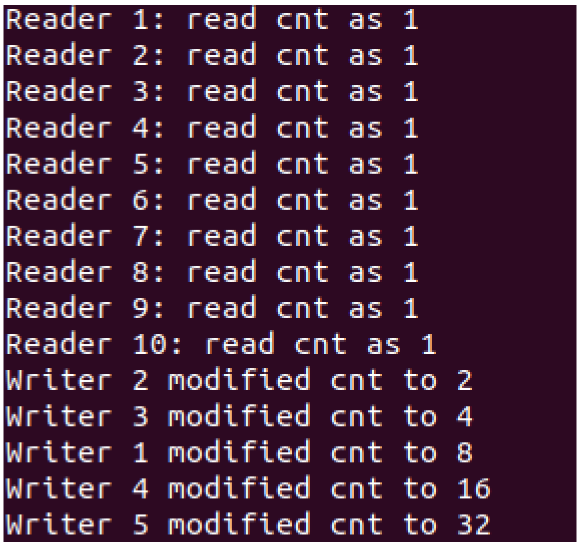
```c
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>

int buf[10],f,r;
sem_t mutex,full,empty;
void *produce(void *arg)
{
        int i;
        for (i = 0; i < 50; i++)
        {
                sem_wait(&empty);
                sem_wait(&mutex);
                printf("produced item is %d\n", i);
                buf[(++r)%10] = i;
                sleep(1);
                sem_post(&mutex);
                sem_post(&full);
        }
}
void *consume(void *arg)
{
        int item,i;
        for (i = 0; i < 50; i++)
        {
                sem_wait(&full);
                sem_wait(&mutex);
                item = buf[(++f)%10];
                printf("consumed item is %d\n", item);
                sleep(1);
                sem_post(&mutex);
                sem_post(&empty);
        }
}

int main()
{
        pthread_t tid1,tid2;
        sem_init(&mutex,0,1);
        sem_init(&full,0,2);
        sem_init(&empty,0,5);
        pthread_create(&tid1,NULL,produce,NULL);
        pthread_create(&tid2,NULL,consume,NULL);
        pthread_join(tid1,NULL);
        pthread_join(tid2,NULL);
        return 0;
}
```



```
consumed item is 0
consumed item is 0
produced item is 0
produced item is 1
produced item is 2
produced item is 3
produced item is 4
produced item is 5
produced item is 6
consumed item is 2
consumed item is 3
consumed item is 4
consumed item is 5
consumed item is 6
consumed item is 0
consumed item is 0
produced item is 7
produced item is 8
produced item is 9
produced item is 10
produced item is 11
produced item is 12
produced item is 13
consumed item is 9
consumed item is 10
consumed item is 11
consumed item is 12
consumed item is 13
consumed item is 4
consumed item is 5
produced item is 14
produced item is 15
produced item is 16
produced item is 17
produced item is 18
produced item is 19
produced item is 20
consumed item is 16
consumed item is 17
consumed item is 18
consumed item is 19
consumed item is 20
consumed item is 11
consumed item is 12
produced item is 21
produced item is 22
produced item is 23
produced item is 24
produced item is 25
produced item is 26
produced item is 27
consumed item is 23
consumed item is 24
consumed item is 25
consumed item is 26
consumed item is 27
consumed item is 18
consumed item is 19
produced item is 28
produced item is 29
produced item is 30
produced item is 31
produced item is 32
produced item is 33
produced item is 34
consumed item is 30
consumed item is 31
consumed item is 32
consumed item is 33
consumed item is 34
consumed item is 25
consumed item is 26
produced item is 35
produced item is 36
produced item is 37
produced item is 38
produced item is 39
produced item is 40
produced item is 41
consumed item is 37
consumed item is 38
consumed item is 39
consumed item is 40
consumed item is 41
consumed item is 32
consumed item is 33
produced item is 42
produced item is 43
produced item is 44
produced item is 45
produced item is 46
produced item is 47
produced item is 48
consumed item is 44
consumed item is 45
consumed item is 46
consumed item is 47
consumed item is 48
consumed item is 39
produced item is 49
```

Q2. First readers-writers problem

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

sem_t wrt;
pthread_mutex_t mutex;
int cnt = 1;
int numreader = 0;
void *writer(void *wno)
{
    sem_wait(&wrt);
    cnt = cnt*2;
    printf("Writer %d modified cnt to %d\n",(*((int *)wno)),cnt);
    sem_post(&wrt);
}
void *reader(void *rno)
{
    pthread_mutex_lock(&mutex);
    numreader++;
    if(numreader == 1)
        sem_wait(&wrt);
    pthread_mutex_unlock(&mutex);
    printf("Reader %d: read cnt as %d\n",*((int *)rno),cnt);
    pthread_mutex_lock(&mutex);
    numreader--;
    if(numreader == 0)
        sem_post(&wrt);
    pthread_mutex_unlock(&mutex);
}
int main()
{
    pthread_t read[10],write[5];
    pthread_mutex_init(&mutex, NULL);
    sem_init(&wrt,0,1);
    int a[10] = {1,2,3,4,5,6,7,8,9,10};
    for(int i = 0; i < 10; i++)
        pthread_create(&read[i], NULL, (void *)reader, (void *)&a[i]);
    for(int i = 0; i < 5; i++)
        pthread_create(&write[i], NULL, (void *)writer, (void *)&a[i]);
    for(int i = 0; i < 10; i++)
        pthread_join(read[i], NULL);
    for(int i = 0; i < 5; i++)
        pthread_join(write[i], NULL);
    pthread_mutex_destroy(&mutex);
    sem_destroy(&wrt);
    return 0;
}
```



```
Reader 1: read cnt as 1
Reader 2: read cnt as 1
Reader 3: read cnt as 1
Reader 4: read cnt as 1
Reader 5: read cnt as 1
Reader 6: read cnt as 1
Reader 7: read cnt as 1
Reader 8: read cnt as 1
Reader 9: read cnt as 1
Reader 10: read cnt as 1
Writer 2 modified cnt to 2
Writer 3 modified cnt to 4
Writer 1 modified cnt to 8
Writer 4 modified cnt to 16
Writer 5 modified cnt to 32
```

Q3. Deadlock

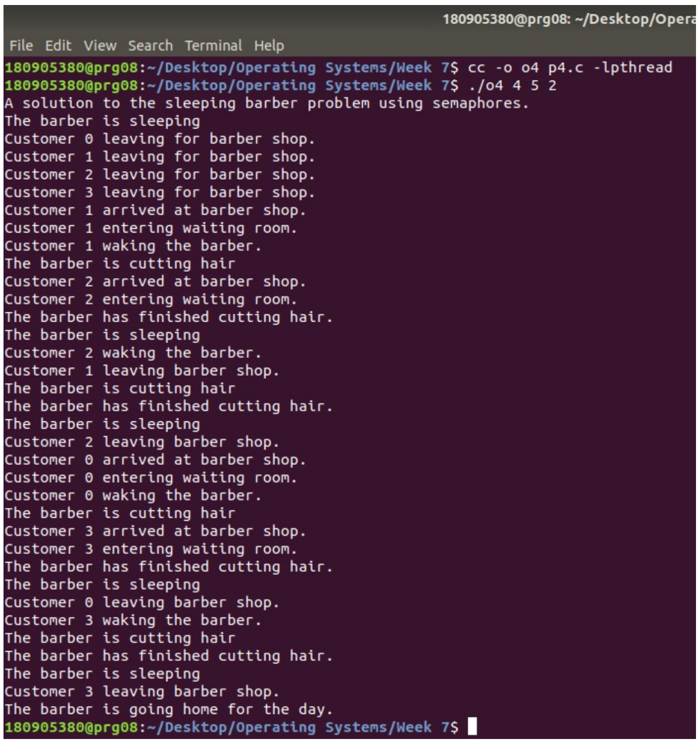
```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>
int num;
sem_t mutex1,mutex2;
void *A(void *arg)
{
   sem_wait(&mutex1);
   sem_wait(&mutex2);
          num = 2;
          printf("function a executing\nnum = %d\n",num);
   sem_post(&mutex2);
   sem_post(&mutex1);
}
void *B(void *arg)
{
   sem_wait(&mutex2);
   sem_wait(&mutex1);
          num = 1;
          printf("function b executing\nnum = %d\n",num);
   sem_post(&mutex1);
   sem_post(&mutex2);
}
int main()
{
          pthread_t tid1,tid2;
          sem_init(&mutex1,0,1);
          sem_init(&mutex2,0,1);
          pthread_create(&tid1,NULL,A,NULL);
          pthread_create(&tid2,NULL,B,NULL);
          pthread_join(tid1,NULL);
          pthread_join(tid2,NULL);
          return 0;
}
```

Terminal has no output due to deadlock

Q4. Sleepy Barber Problem

```
#define _REENTRANT

#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <pthread.h>
#include <semaphore.h>
#define MAX_CUSTOMERS 25


void *customer(void *num);
void *barber(void *);
void randwait(int secs);

sem_t waitingRoom;
sem_t barberChair;
sem_t barberPillow;
sem_t seatBelt;

int allDone = 0;

int main(int argc, char *argv[])
{
   pthread_t btid;
   pthread_t tid[MAX_CUSTOMERS];
   long RandSeed;
   int i, numCustomers, numChairs;
   int Number[MAX_CUSTOMERS];
   if (argc != 4)
   {
          printf("Use: SleepBarber <Num Customers> <Num Chairs> <rand seed>\n");
          exit(-1);
   }
   numCustomers = atoi(argv[1]);
   numChairs = atoi(argv[2]);
   RandSeed = atol(argv[3]);
   if (numCustomers > MAX_CUSTOMERS)
   {
          printf("The maximum number of Customers is %d.\n", MAX_CUSTOMERS);
          exit(-1);
   }
   printf("A solution to the sleeping barber problem using semaphores.\n");
   srand48(RandSeed);
   for (i=0; i<MAX_CUSTOMERS; i++)
            Number[i] = i;
   sem_init(&waitingRoom, 0, numChairs);
   sem_init(&barberChair, 0, 1);
   sem_init(&barberPillow, 0, 0);
   sem_init(&seatBelt, 0, 0);
   pthread_create(&btid, NULL, barber, NULL);
   for (i=0; i<numCustomers; i++)
            pthread_create(&tid[i], NULL, customer, (void *)&Number[i]);
   for (i=0; i<numCustomers; i++)
            pthread_join(tid[i],NULL);
   allDone = 1;
   sem_post(&barberPillow);
   pthread_join(btid,NULL);
}

void *customer(void *number)
```

```
{
   int num = *(int *)number;
   printf("Customer %d leaving for barber shop.\n", num);
   randwait(5);
   printf("Customer %d arrived at barber shop.\n", num);
   sem_wait(&waitingRoom);
   printf("Customer %d entering waiting room.\n", num);
   sem_wait(&barberChair);
   sem_post(&waitingRoom);
   printf("Customer %d waking the barber.\n", num);
   sem_post(&barberPillow);
   sem_wait(&seatBelt);
   sem_post(&barberChair);
   printf("Customer %d leaving barber shop.\n", num);
}

void *barber(void *junk)
{
   while (!allDone)
   {
           printf("The barber is sleeping\n");
           sem_wait(&barberPillow);
           if (!allDone)
   {
              printf("The barber is cutting hair\n");
              randwait(3);
              printf("The barber has finished cutting hair.\n");
              sem_post(&seatBelt);
           }
           else
              printf("The barber is going home for the day.\n");
   }
}

void randwait(int secs)
{
   int len;
   len = (int) ((drand48() * secs) + 1);
   sleep(len);
}
```



180905380@prg08: ~/Desktop/Opera
File Edit View Search Terminal Help

```
180905380@prg08:~/Desktop/Operating Systems/Week 7$ cc -o o4 p4.c -lpthread
180905380@prg08:~/Desktop/Operating Systems/Week 7$ ./o4 4 5 2
A solution to the sleeping barber problem using semaphores.
The barber is sleeping
Customer 0 leaving for barber shop.
Customer 1 leaving for barber shop.
Customer 2 leaving for barber shop.
Customer 3 leaving for barber shop.
Customer 1 arrived at barber shop.
Customer 1 entering waiting room.
Customer 1 waking the barber.
The barber is cutting hair
Customer 2 arrived at barber shop.
Customer 2 entering waiting room.
The barber has finished cutting hair.
The barber is sleeping
Customer 2 waking the barber.
Customer 1 leaving barber shop.
The barber is cutting hair
The barber has finished cutting hair.
The barber is sleeping
Customer 2 leaving barber shop.
Customer 0 arrived at barber shop.
Customer 0 entering waiting room.
Customer 0 waking the barber.
The barber is cutting hair
Customer 3 arrived at barber shop.
Customer 3 entering waiting room.
The barber has finished cutting hair.
The barber is sleeping
Customer 0 leaving barber shop.
Customer 3 waking the barber.
The barber is cutting hair
The barber has finished cutting hair.
The barber is sleeping
Customer 3 leaving barber shop.
The barber is going home for the day.
180905380@prg08:~/Desktop/Operating Systems/Week 7$
```