COLLEGE OF COMPUTING AND INFORMATION SCIENCES

Task # 11

Class Id	110084	**Course Title**	OS
Student Id	**64091**	**Student Name**	**Hassaan Raheem**
Total Marks	05	**Obtained Marks**	

QUESTION:

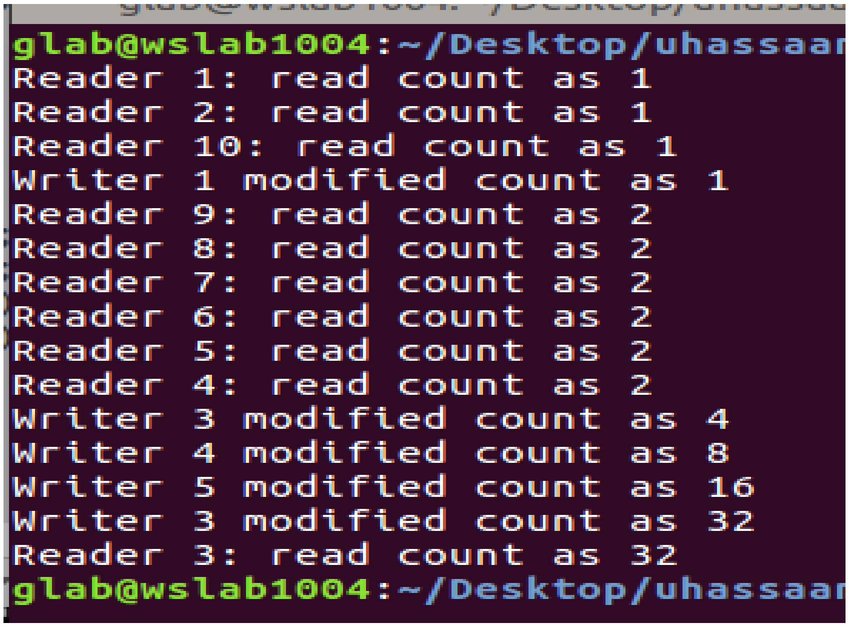
1. Any number of readers may simultaneously be reading from a file. Only one writer at a time may write to a file, and no reader can be reading while a writer is writing. Using semaphores, write a solution to the readers and writers problem that gives priority to readers.

CODE :

```
#include <pthread.h>

#include <semaphore.h>

#include <stdio.h>

#include <unistd.h>

//global variables

sem_t wrt;

pthread_mutex_t mutex;

int cnt = 1;

int numreader = 0;

//writer function

void *writer(void *wno)

{

      sem_wait(&wrt);

      cnt = cnt*2;
```

```
        printf("Writer %d modified cnt to %d\n",(*((int *)wno)),cnt);
        sem_post(&wrt);
}
//reader function
void *reader(void *rno)
{
        // Reader becomes lock before modifying numreader
        pthread_mutex_lock(&mutex);
        numreader++;
        if(numreader == 1) {
        sem_wait(&wrt); // If id becomes first reader, then it will block the writer
        }
        pthread_mutex_unlock(&mutex);
        // Reading Section
        printf("Reader %d: read cnt as %d\n",*((int *)rno),cnt);

        // Reader becomes lock before modifying numreader
        pthread_mutex_lock(&mutex);
        numreader--;
        if(numreader == 0) {
        sem_post(&wrt); // If this is the last reader, it will wake up the writer.
        }
        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}; //used numbers for representing producer/consumers.

      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;

}
```

OUTPUT :



```
glab@wslab1004:~/Desktop/uhassaa
Reader 1: read count as 1
Reader 2: read count as 1
Reader 10: read count as 1
Writer 1 modified count as 1
Reader 9: read count as 2
Reader 8: read count as 2
Reader 7: read count as 2
Reader 6: read count as 2
Reader 5: read count as 2
Reader 4: read count as 2
Writer 3 modified count as 4
Writer 4 modified count as 8
Writer 5 modified count as 16
Writer 3 modified count as 32
Reader 3: read count as 32
glab@wslab1004:~/Desktop/uhassaa
```