READER -WRITER Program

```
#include<stdio.h>
#include<unistd.h>
#include<pthread.h>
#include<semaphore.h>
#include<stdlib.h>
pthread_mutex_t rwmutex;
pthread_mutex_t lock;
int rdcnt,nr,nw;
pthread_t thread;
void *reader(void *arg);
void *writer(void *arg);
void init();
void main()
{
        int i;
        init();
        printf("\nEnter no of readers ");
        scanf("%d",&nr);
        printf("\nEnter no of writers ");
        scanf("%d",&nw);
        for(i=0;i<nw;i++)
        {
                int *arg=malloc(sizeof(int *));
                *arg=i;
                pthread_create(&thread,NULL,writer,arg);
        }
        for(i=0;i<nr;i++)
        {
                int *arg=malloc(sizeof(int *));
                *arg=i;
                pthread_create(&thread,NULL,reader,arg);
        }

        for(i=0;i<nw;i++)
        {
                pthread_join(thread,NULL);
        }
        for(i=0;i<nr;i++)
        {
                pthread_join(thread,NULL);
        }

}
void init()
{
        pthread_mutex_init(&lock,NULL);
        pthread_mutex_init(&rwmutex,NULL);
        rdcnt=0;
}
void *reader(void *arg)
```

```
{
        int i=*(int *)arg;
        int cnt=0;
        printf("\nreader %d is trying to read",i+1);
        pthread_mutex_lock(&lock);
        rdcnt++;
        if(rdcnt==1)
                pthread_mutex_lock(&rwmutex);
        printf("\nreader %d is reading ",i+1);
        pthread_mutex_unlock(&lock);
        sleep(3);
        pthread_mutex_lock(&lock);
        rdcnt--;
        if(rdcnt==0)
                pthread_mutex_unlock(&rwmutex);
        pthread_mutex_unlock(&lock);
        printf("\nreader %d is leaving",i+1);
}
void *writer(void *arg)
{
        int i=*(int *)arg;
        printf("\nwriter %d is trying to write",i+1);
        pthread_mutex_lock(&rwmutex);
        printf("\nwriter %d is writing ",i+1);
        sleep(3);
        pthread_mutex_unlock(&rwmutex);
        printf("\nwriter %d is leaving",i+1);

}
```

OUTPUT:

```
pl-lab@pllab-OptiPlex-3000:~$ gcc readwrite.c
pl-lab@pllab-OptiPlex-3000:~$ ./a.out

Enter no of readers 4

Enter no of writers 4

writer 1 is trying to write
writer 1 is writing
writer 2 is trying to write
writer 3 is trying to write
writer 4 is trying to write
reader 2 is trying to read
reader 1 is trying to read
reader 4 is trying to read
reader 3 is trying to read
writer 1 is leaving
writer 2 is writing
writer 2 is leaving
writer 3 is writing
writer 3 is leaving
```

writer 4 is writing
writer 4 is leaving
reader 2 is reading
reader 1 is reading
reader 4 is reading
reader 3 is reading
reader 2 is leaving
reader 1 is leaving
reader 4 is leaving