Assignment 5: **Reader Writer Problem solution with Mutex, Semaphore.**

Name: Chaitanya Suresh Uge
Roll Number: 333064
GR Number: 21910718
Division: C Batch: C3

Theory:

- If one of the people tries editing the file, no other person should be reading or writing at the same time, otherwise changes will not be visible to him/her.
- However if some person is reading the file, then others may read it at the same time.
- Precisely in OS we call this situation as the readers-writers problem

Problem parameters:

- One set of data is shared among a number of processes.
- Once a writer is ready, it performs its writing. Only one writer may write at a time.
- If a process is writing, no other process can read it.
- If at least one reader is reading, no other process can write. - Readers may not write and only read.

Functions for semaphore :

– wait() : decrements the semaphore value.
– signal() : increments the semaphore value.
Writer process:

- Writer requests the entry to the critical section.
- If allowed i.e. wait() gives a true value, it enters and performs the write. If not allowed, it keeps on waiting.

- It exits the critical section.

Reader process:

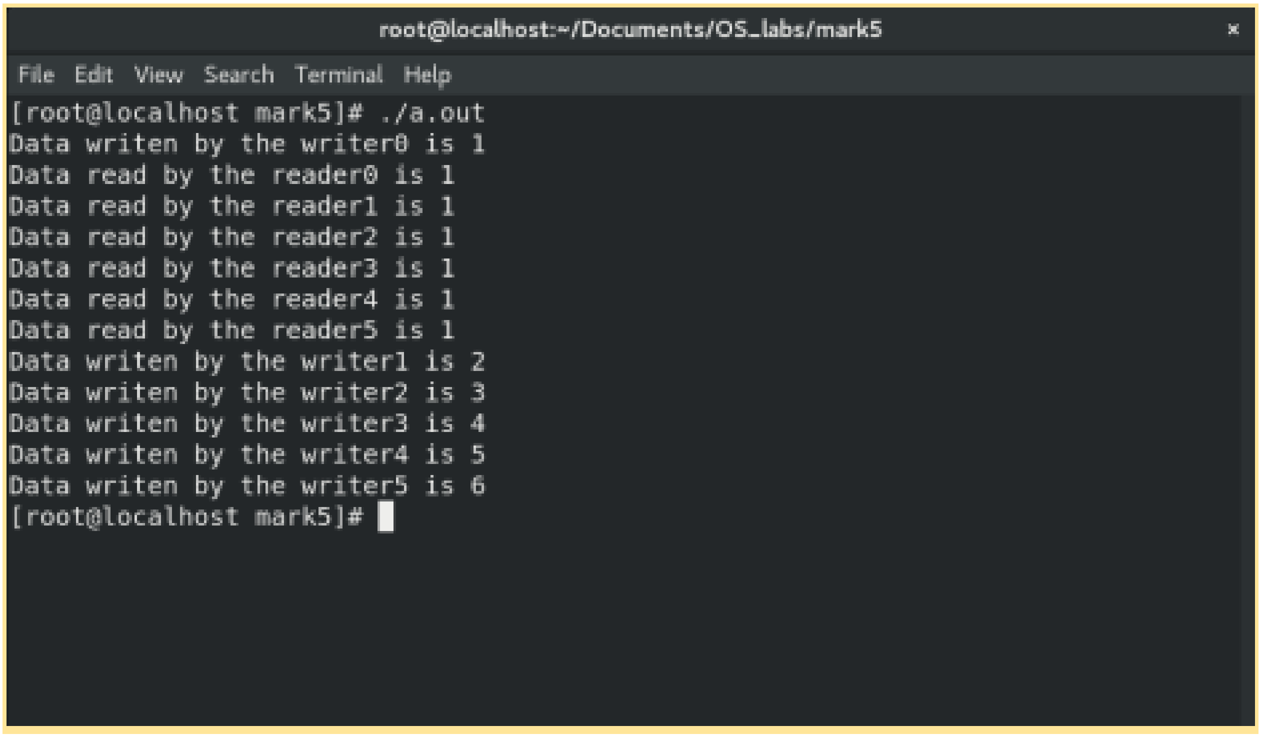
- Reader requests the entry to the critical section.
- If allowed:
    - it increments the count of the number of readers inside the critical section. If this reader is the first reader entering, it locks the wrt semaphore to restrict the entry of writers if any reader is inside. -
    - It then signals mutex as any other reader is allowed to enter while others are already reading.
    - After performing reading, it exits the critical section. When exiting, it checks if no more reader is inside, it signals the semaphore “wrt” as now, the writer can enter the critical section.
- If not allowed, it keeps on waiting.

Source Code:

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

sem_t mutex,writeblock;
int data = 0,rcount = 0;

void *reader(void *arg)
{
 int f;
 f = ((int)arg);
 sem_wait(&mutex);
 rcount = rcount + 1;
 if(rcount==1)
```

```
   sem_wait(&writeblock);
  sem_post(&mutex);
  printf("Data read by the reader%d is %d\n",f,data);
  sleep(1);
  sem_wait(&mutex);
  rcount = rcount - 1;
  if(rcount==0)
   sem_post(&writeblock);
  sem_post(&mutex);
}

void *writer(void *arg)
{
  int f;
  f = ((int) arg);
  sem_wait(&writeblock);
  data++;
  printf("Data writen by the writer%d is %d\n",f,data);
  sleep(1);
  sem_post(&writeblock);
}

int main()
{
  pthread_t rtid[20],wtid[20];
  sem_init(&mutex,0,1);
  sem_init(&writeblock,0,1);
  for(int i=0;i<=5;i++)
  {
    pthread_create(&wtid[i],NULL,writer,(void *)i);
                pthread_create(&rtid[i],NULL,reader,(void *)i);
  }

  for(int i=0;i<=5;i++)
  {
    pthread_join(wtid[i],NULL);
    pthread_join(rtid[i],NULL);
  }

          return 0;
}
```

OUTPUT:


root@localhost:~/Documents/OS_labs/mark5
File Edit View Search Terminal Help

```
[root@localhost mark5]# ./a.out
Data writen by the writer0 is 1
Data read by the reader0 is 1
Data read by the reader1 is 1
Data read by the reader2 is 1
Data read by the reader3 is 1
Data read by the reader4 is 1
Data read by the reader5 is 1
Data writen by the writer1 is 2
Data writen by the writer2 is 3
Data writen by the writer3 is 4
Data writen by the writer4 is 5
Data writen by the writer5 is 6
[root@localhost mark5]# 
```