به نام خدا

گزارش آزمایش ششم آزمایشگاه سیستم های عامل

زهرا رحیمی

شماره دانشجوئی: ۹۸۳۱۰۲۶

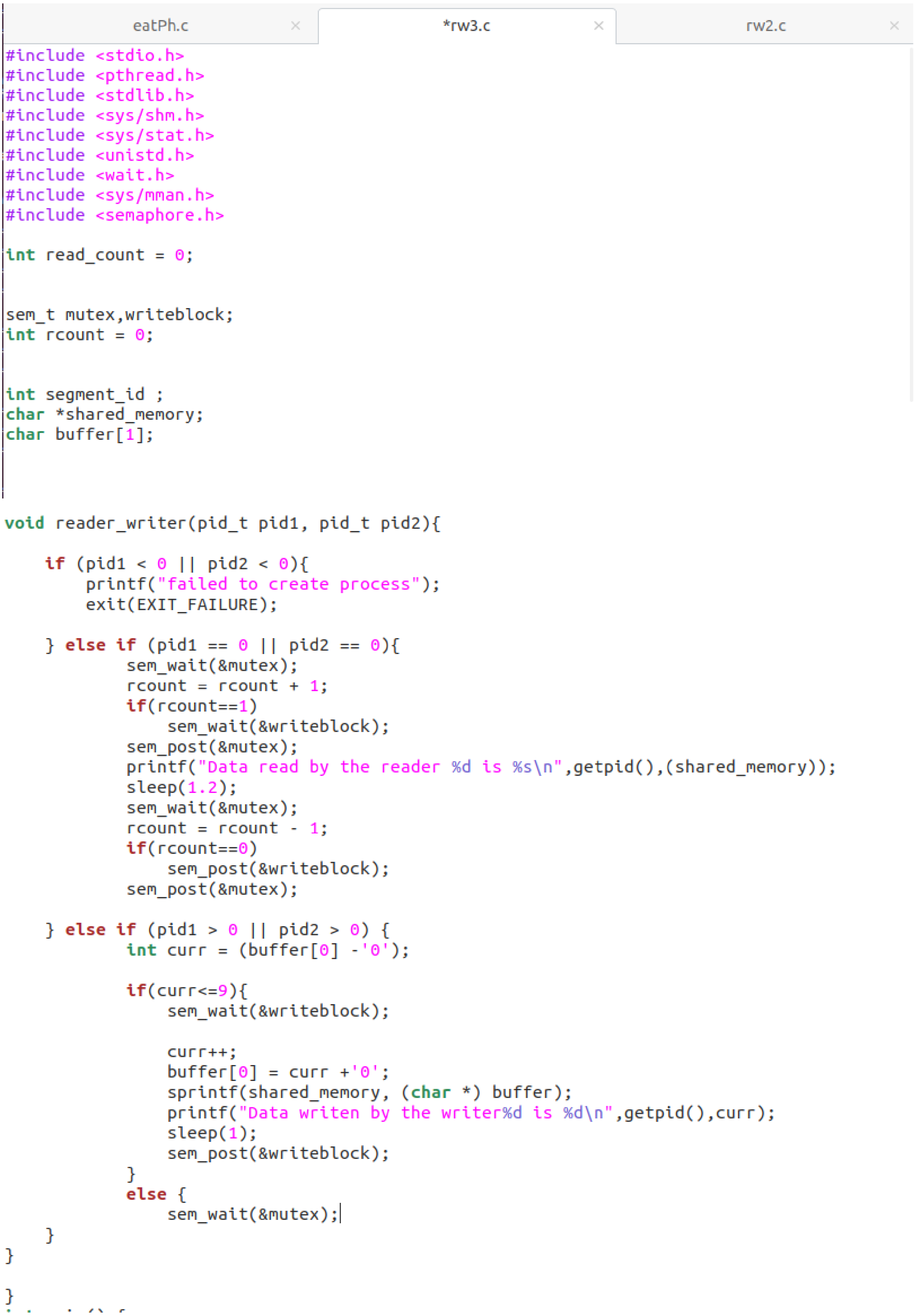
استاد آزمایشگاه: سرکار خانم حسینی

پاییز ۱۴۰۰

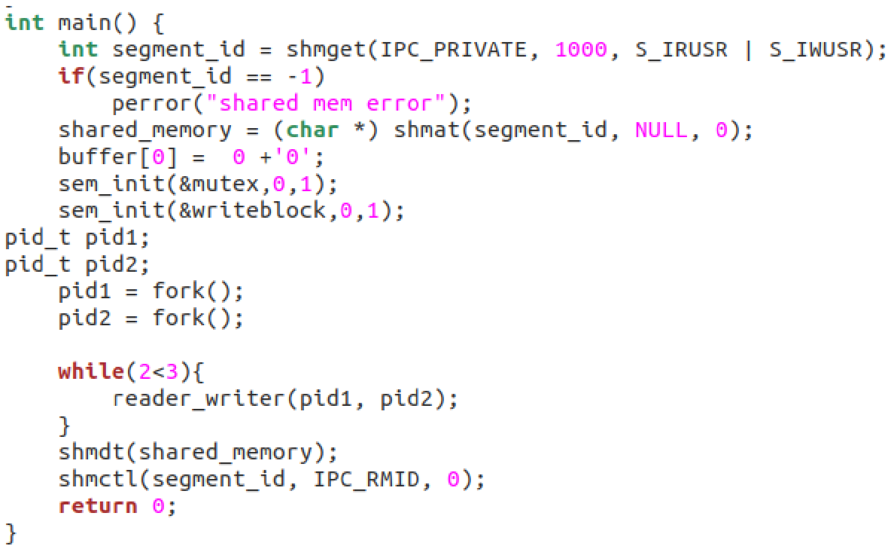
بخش اول: مسئله خوانندگان-نویسندگان:

همان طور که در دستور کار آمده در این بخش چند فرآیند خواننده قرار است اطلاعاتی که فرآیند نویسنده روی حافظه مشترک می گذارد را بخوانند. وجود حافظه مشترک و دو فرآیند همکار که بر هم اثر گذاری و از هم اثر پذیری دارند موجب به وجود آمدن شرایط race condition شده که این باعث می شود برای مثال اگر فرآیند نویسنده داده را یک واحد زیاد و سپس کم کند، یک فرآیند داده دست نخورده، یک فرآیند داده افزایش یافته و یک فرآیند داده کم شده را بخواند. این برای ما مطلوب نیست در نتیجه با استفاده از blocking اجبار می کنیم در قسمت هایی از برنامه context switch رخ ندهد.

این مسئله هم با fork و ساخت فرآیند و هم با thread و ساخت رشته پیاده سازی شده که در ادامه ابتدا کد مربوط به فرآیند و سپس رشته را مشاهده می کنیم.

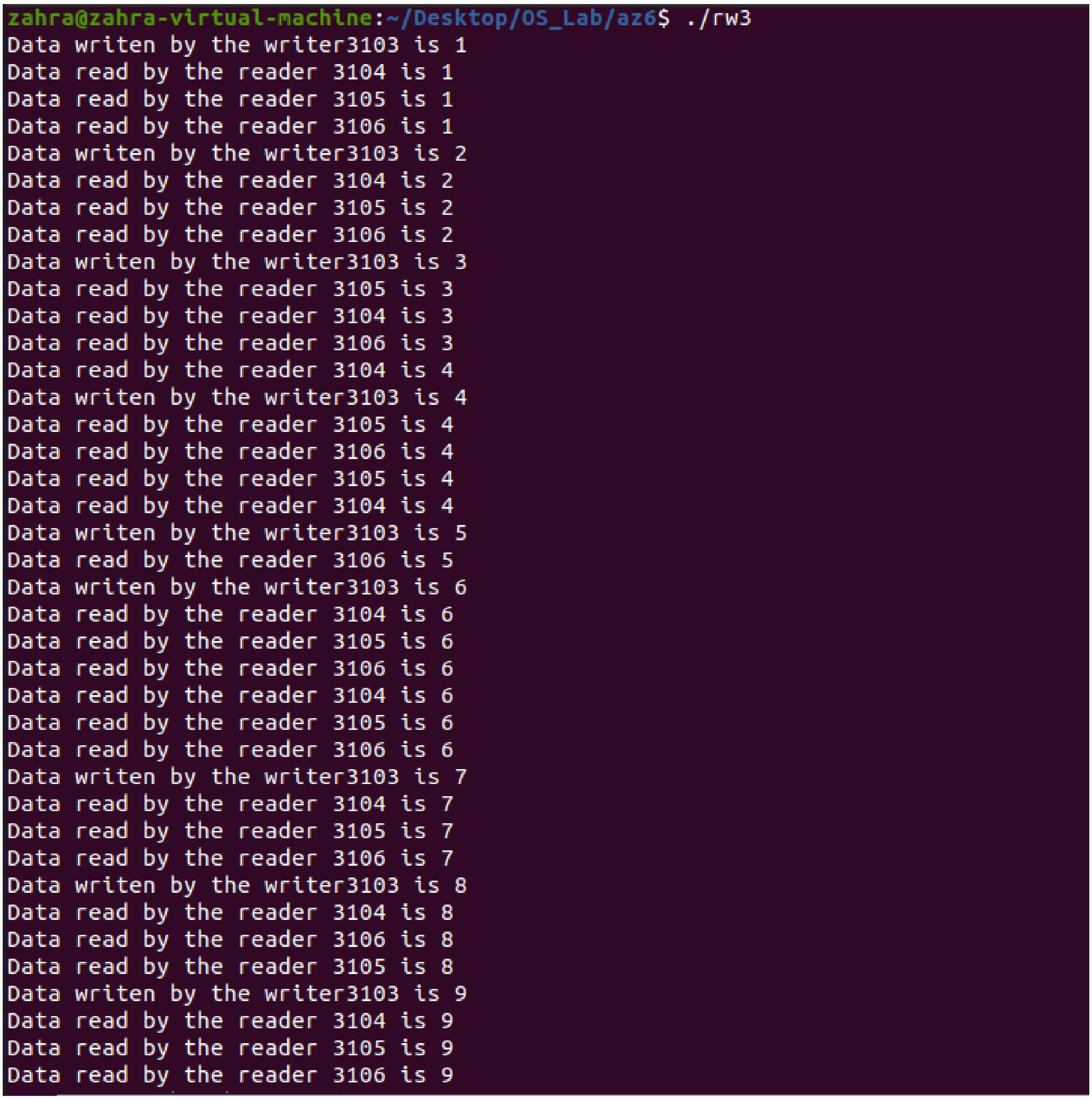


```
#include <stdio.h>
#include <pthread.h>
#include <stdlib.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <unistd.h>
#include <wait.h>
#include <sys/mman.h>
#include <semaphore.h>

int read_count = 0;


sem_t mutex,writeblock;
int rcount = 0;


int segment_id ;
char *shared_memory;
char buffer[1];



void reader_writer(pid_t pid1, pid_t pid2){

    if (pid1 < 0 || pid2 < 0){
        printf("failed to create process");
        exit(EXIT_FAILURE);

    } else if (pid1 == 0 || pid2 == 0){
            sem_wait(&mutex);
            rcount = rcount + 1;
            if(rcount==1)
                sem_wait(&writeblock);
            sem_post(&mutex);
            printf("Data read by the reader %d is %s\n",getpid(),(shared_memory));
            sleep(1.2);
            sem_wait(&mutex);
            rcount = rcount - 1;
            if(rcount==0)
                sem_post(&writeblock);
            sem_post(&mutex);

    } else if (pid1 > 0 || pid2 > 0) {
            int curr = (buffer[0] -'0');

            if(curr<=9){
                sem_wait(&writeblock);

                curr++;
                buffer[0] = curr +'0';
                sprintf(shared_memory, (char *) buffer);
                printf("Data writen by the writer%d is %d\n",getpid(),curr);
                sleep(1);
                sem_post(&writeblock);
            }
            else {
                sem_wait(&mutex);
    }
}

}
```



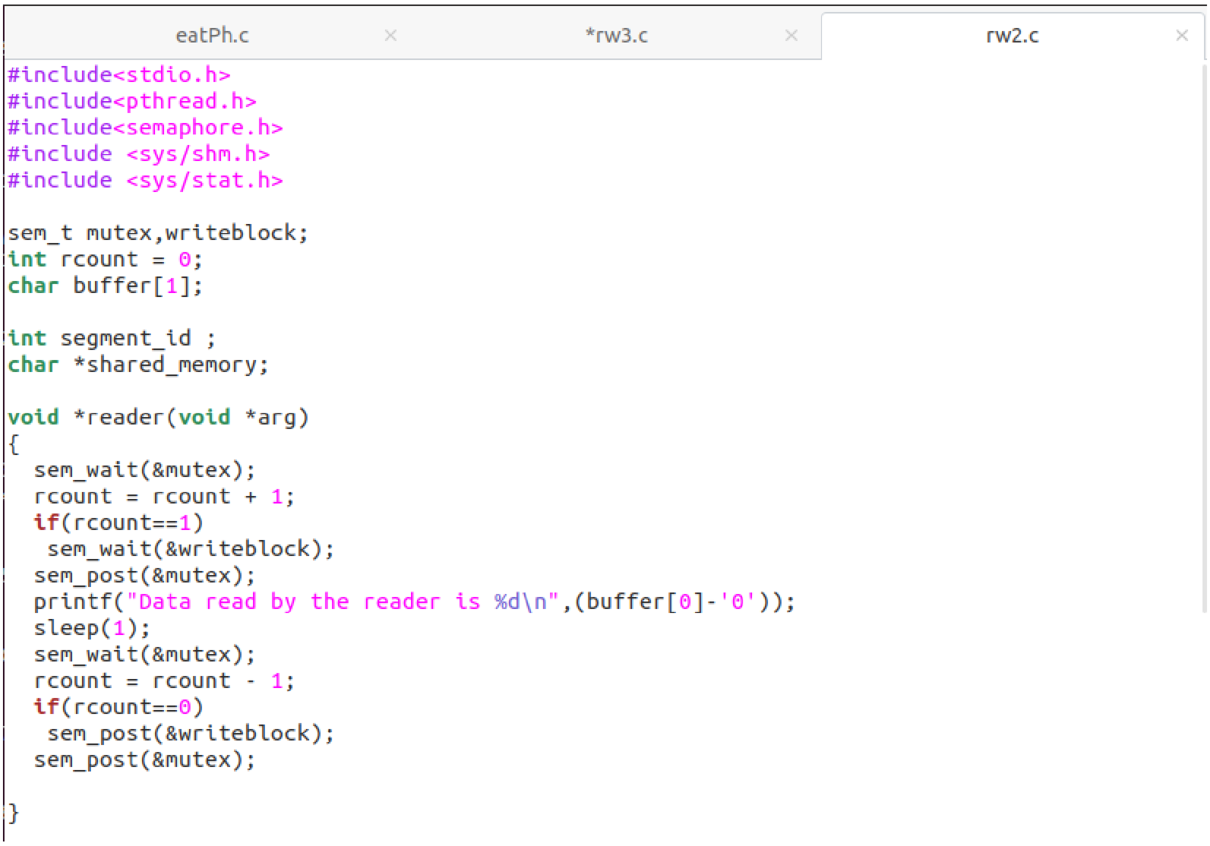
```
int main() {
    int segment_id = shmget(IPC_PRIVATE, 1000, S_IRUSR | S_IWUSR);
    if(segment_id == -1)
        perror("shared mem error");
    shared_memory = (char *) shmat(segment_id, NULL, 0);
    buffer[0] =  0 +'0';
    sem_init(&mutex,0,1);
    sem_init(&writeblock,0,1);
pid_t pid1;
pid_t pid2;
    pid1 = fork();
    pid2 = fork();

    while(2<3){
        reader_writer(pid1, pid2);
    }
    shmdt(shared_memory);
    shmctl(segment_id, IPC_RMID, 0);
    return 0;
}
```

در زیر خروجی کد را می بینیم:

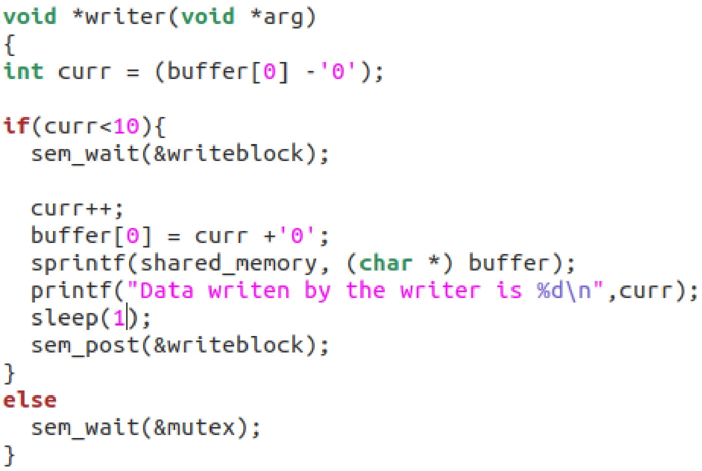


```
zahra@zahra-virtual-machine:~/Desktop/OS_Lab/az6$ ./rw3
Data writen by the writer3103 is 1
Data read by the reader 3104 is 1
Data read by the reader 3105 is 1
Data read by the reader 3106 is 1
Data writen by the writer3103 is 2
Data read by the reader 3104 is 2
Data read by the reader 3105 is 2
Data read by the reader 3106 is 2
Data writen by the writer3103 is 3
Data read by the reader 3105 is 3
Data read by the reader 3104 is 3
Data read by the reader 3106 is 3
Data read by the reader 3104 is 4
Data writen by the writer3103 is 4
Data read by the reader 3105 is 4
Data read by the reader 3106 is 4
Data read by the reader 3105 is 4
Data read by the reader 3104 is 4
Data writen by the writer3103 is 5
Data read by the reader 3106 is 5
Data writen by the writer3103 is 6
Data read by the reader 3104 is 6
Data read by the reader 3105 is 6
Data read by the reader 3106 is 6
Data read by the reader 3104 is 6
Data read by the reader 3105 is 6
Data read by the reader 3106 is 6
Data writen by the writer3103 is 7
Data read by the reader 3104 is 7
Data read by the reader 3105 is 7
Data read by the reader 3106 is 7
Data writen by the writer3103 is 8
Data read by the reader 3104 is 8
Data read by the reader 3106 is 8
Data read by the reader 3105 is 8
Data writen by the writer3103 is 9
Data read by the reader 3104 is 9
Data read by the reader 3105 is 9
Data read by the reader 3106 is 9
```

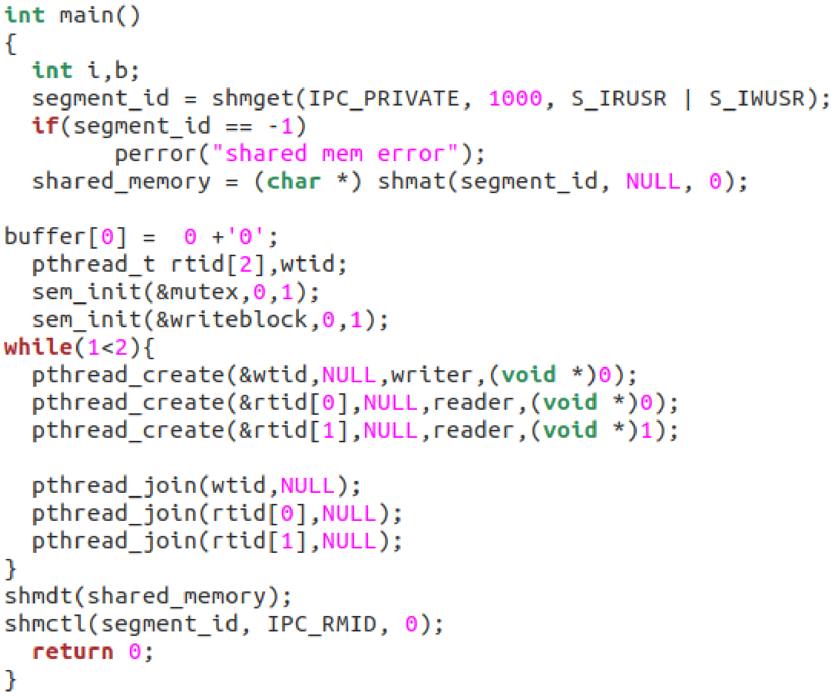
کد مربوط به پیاده سازی مسئله با thread:



```c
#include<stdio.h>
#include<pthread.h>
#include<semaphore.h>
#include <sys/shm.h>
#include <sys/stat.h>

sem_t mutex,writeblock;
int rcount = 0;
char buffer[1];

int segment_id ;
char *shared_memory;

void *reader(void *arg)
{
  sem_wait(&mutex);
  rcount = rcount + 1;
  if(rcount==1)
   sem_wait(&writeblock);
  sem_post(&mutex);
  printf("Data read by the reader is %d\n",(buffer[0]-'0'));
  sleep(1);
  sem_wait(&mutex);
  rcount = rcount - 1;
  if(rcount==0)
   sem_post(&writeblock);
  sem_post(&mutex);

}
```

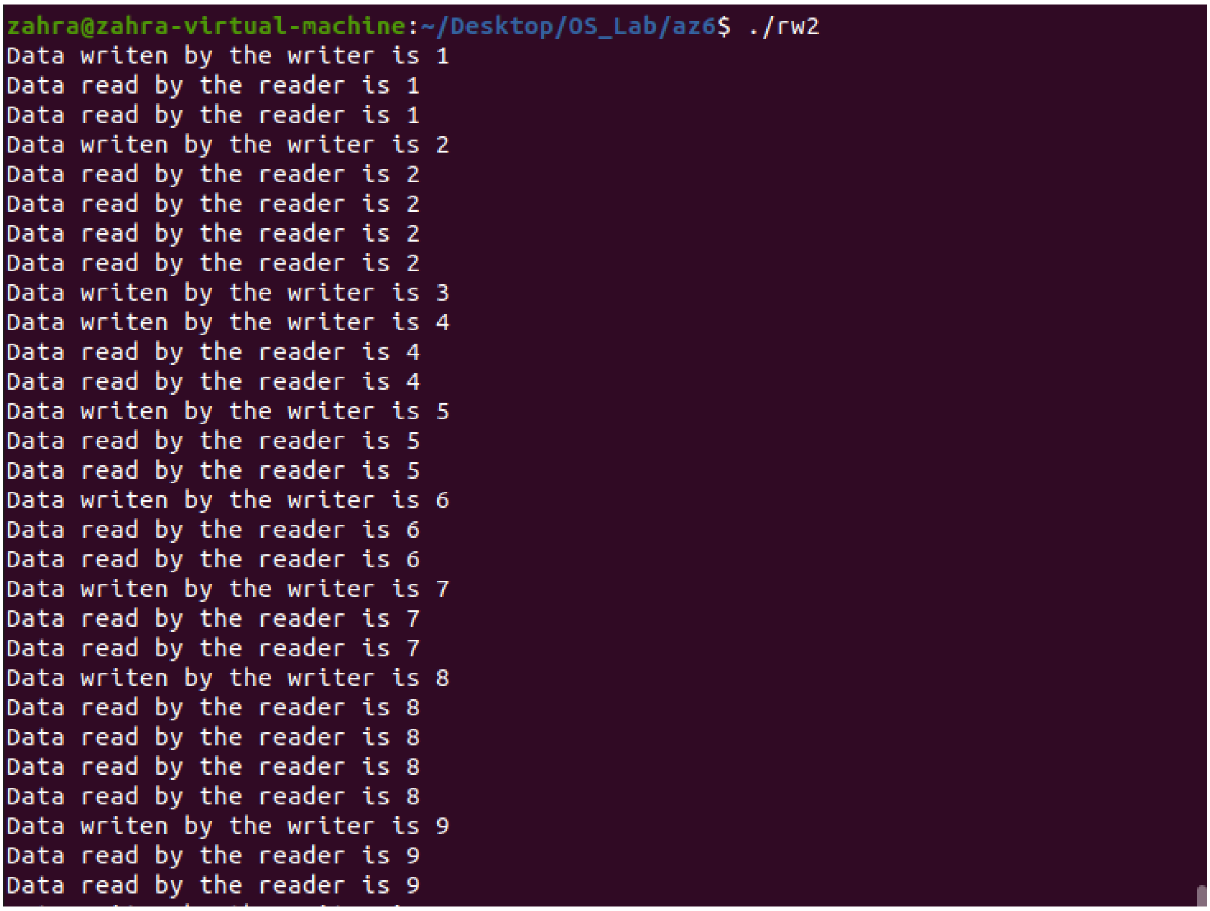


```c
void *writer(void *arg)
{
int curr = (buffer[0] -'0');

if(curr<10){
  sem_wait(&writeblock);

  curr++;
  buffer[0] = curr +'0';
  sprintf(shared_memory, (char *) buffer);
  printf("Data writen by the writer is %d\n",curr);
  sleep(1);
  sem_post(&writeblock);
}
else
  sem_wait(&mutex);
}
```



```
int main()
{
  int i,b;
  segment_id = shmget(IPC_PRIVATE, 1000, S_IRUSR | S_IWUSR);
  if(segment_id == -1)
        perror("shared mem error");
  shared_memory = (char *) shmat(segment_id, NULL, 0);

buffer[0] =  0 +'0';
  pthread_t rtid[2],wtid;
  sem_init(&mutex,0,1);
  sem_init(&writeblock,0,1);
while(1<2){
  pthread_create(&wtid,NULL,writer,(void *)0);
  pthread_create(&rtid[0],NULL,reader,(void *)0);
  pthread_create(&rtid[1],NULL,reader,(void *)1);

  pthread_join(wtid,NULL);
  pthread_join(rtid[0],NULL);
  pthread_join(rtid[1],NULL);
}
shmdt(shared_memory);
shmctl(segment_id, IPC_RMID, 0);
  return 0;
}
```

خروجی کد:

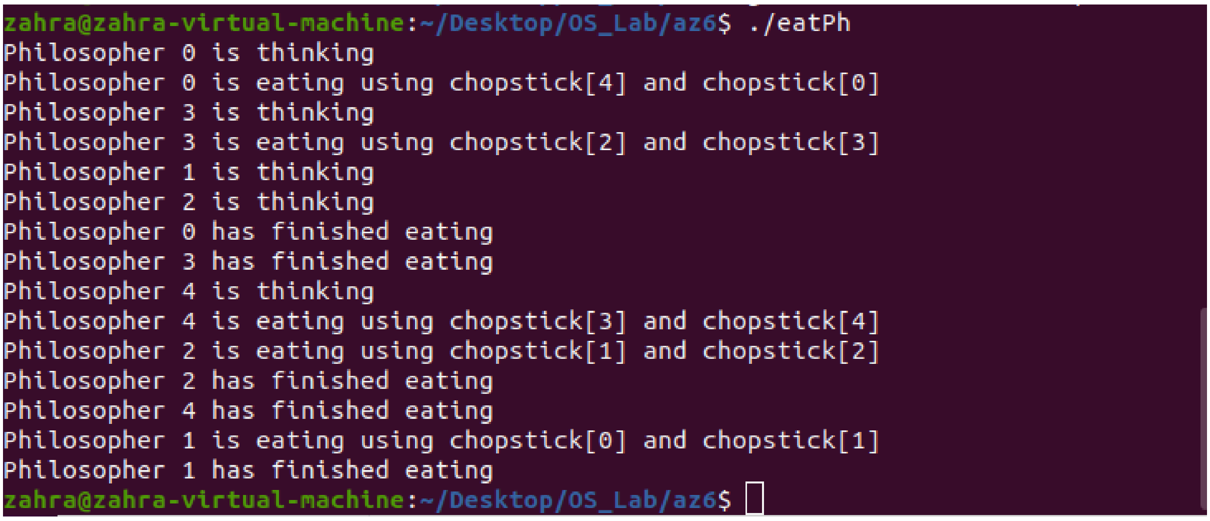


```
zahra@zahra-virtual-machine:~/Desktop/OS_Lab/az6$ ./rw2
Data writen by the writer is 1
Data read by the reader is 1
Data read by the reader is 1
Data writen by the writer is 2
Data read by the reader is 2
Data read by the reader is 2
Data read by the reader is 2
Data read by the reader is 2
Data writen by the writer is 3
Data writen by the writer is 4
Data read by the reader is 4
Data read by the reader is 4
Data writen by the writer is 5
Data read by the reader is 5
Data read by the reader is 5
Data writen by the writer is 6
Data read by the reader is 6
Data read by the reader is 6
Data writen by the writer is 7
Data read by the reader is 7
Data read by the reader is 7
Data writen by the writer is 8
Data read by the reader is 8
Data read by the reader is 8
Data read by the reader is 8
Data read by the reader is 8
Data writen by the writer is 9
Data read by the reader is 9
Data read by the reader is 9
```

بخش دوم: مسئله فیلسوف های غذاخور

در این بخش می بایست الگوریتمی پیاده سازی می شد که طی آن ۵ فیلسوف که دور یک میز با ۵ چوب چابستیک نشسته اند، بتوانند فکر کنند یا غذا بخورند.

الف) بله ممکن است، اگر هر پنج نفر چوب سمت یک طرف (برای مثال راست) را برداشته باشند و منتظر چوب سمت دیگر (چپ) باشند در این صورت هیچ پیشرفتی رخ نمی دهد و هر پنج نفر منتظر نفر کناری هستند تا چوب را زمین را بگذارد و آن فرد نیز منتظر نفر کناری تا ... برسد به نفر یک مانده به آخر که منتظر نفر اول است که این بن بست است.



```
zahra@zahra-virtual-machine:~/Desktop/OS_Lab/az6$ ./eatPh
Philosopher 0 is thinking
Philosopher 0 is eating using chopstick[4] and chopstick[0]
Philosopher 3 is thinking
Philosopher 3 is eating using chopstick[2] and chopstick[3]
Philosopher 1 is thinking
Philosopher 2 is thinking
Philosopher 0 has finished eating
Philosopher 3 has finished eating
Philosopher 4 is thinking
Philosopher 4 is eating using chopstick[3] and chopstick[4]
Philosopher 2 is eating using chopstick[1] and chopstick[2]
Philosopher 2 has finished eating
Philosopher 4 has finished eating
Philosopher 1 is eating using chopstick[0] and chopstick[1]
Philosopher 1 has finished eating
zahra@zahra-virtual-machine:~/Desktop/OS_Lab/az6$
```