به نام خدا

امیرحسین باریکلو 9730003 و آرمان حاتمی 9730008

قسمت اول)

جواب سوال ←با پیاده سازی این قسمت در حالت عادی با مشکل race condition روبه میشویم برای حل آن باید از semaphore در پیاده سازی خود استفاده کنیم

نحوه پیاده سازی :

ابتدا کتابخانه ها را فراخوانی میکنیم

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <wait.h>
#include <unistd.h>
#include <inttypes.h>
#include <math.h>
```

```
#include <sys/time.h>
```

سپس توابع down , up , writer , reader و متغیر های گلوبال pid , number را تعریف میکنیم

```
typedef int Semaphore;

void down(Semaphore* s);
void up(Semaphore* s);
void writer(Semaphore* s1,Semaphore* s2);
void reader(Semaphore* s1,Semaphore* s2,Semaphore* s3,Semaphore* s4);

int pid;
```

```
int numb
```

سپس حد بالی counter و تعداد readerها و تعداد writerها را از کاربر میگیریم

```
 printf("enter counter limit:" );
  scanf("%d",&number);
  int reader_number;
  printf("enter reader number:");
  scanf("%d",&reader_number);
  int writer_number;
  printf("enter writer number:");
```

```
scanf("%d",&writer_number);
```

در مرحله بعد یک shared memory میسازیم که ارتباط بین فرایند ها و اشتراک اطلاعات از ان طریق ممکن میشود

پس از ساخت shared memory ان را به یک ارایه 4 تایی تیدیل کرده و هریک از ان ها در بردارنده مقدار های mutex,db,counter ,rc هسستند

```
int segment_id;
  int* shared_memory;
  int shared_segment_size = 4*sizeof(int);
  key_t key = ftok("shmfile",65);
  segment_id = shmget(key, shared_segment_size,0666|IPC_CREAT);
  int* hist = (int*)shmat(segment_id,NULL,0);

  //mutex = hist[0]
  //db = hist[1]
   //rc = hist [2]
  //counter = hist[3]
  for (int i = 0; i < 4; ++i)
  {
      if(i < 2)
          hist[i] = 1;
      else
          hist[i] = 0;
      }
```

در مرحله اخر به تعدادی کع کاربر دستور داده با استفاده از fork پردازه reader و writer میسازیم که پردازه های reader و تابع reader و پردازه های writerتابع writerرا فراخوانی کرده و پس از اتمام ان ها هر یک از پردازه ها تمام میشود

```
      for (int i = 0; i < reader_number; ++i)
      {
          pid = fork();
          if(pid == 0){
              reader(&hist[1],&hist[3],&hist[0],&hist[2]);
              //writer(&hist[1],&hist[3]);
              exit(0);
          }
          if(pid<0)
              printf("error\n");



      }
      for (int i = 0; i < writer_number; ++i)
      {
          pid = fork();
          if(pid==0){
              writer(&hist[1],&hist[3]);
              exit(0);
          }
                                                                          }
```

پس از ایجاد پردازه ها پردازه اصلی منتظر میماند تا همه تمام شوند

```
while(hist[3]<number);
```

توابع reader , writer

به این صورت است که در پایین اورده شده

```
void writer(Semaphore* db,Semaphore* counter)
{
    pid_t pid_new;
    pid_new = getpid();برای گرفتن مقدار پراسس ای دی ای که در ان هستیم
    while (*counter<number) {



        down(db); مقدار db را یکی کم میکند با تابع down که در ادامه به ان میپردازیم این کار برای این است که در ناحیه بحرانی پایین در هر زمان فقط یک پراسس موجود باشد (چند ریدر با هم مشکلی ندارد)

        *counter = *counter + 1;
         printf("pid :%d and ",pid_new);
        printf("counter: %d\n",*counter);

        up(db); مقدار db را یکی زیاد میکنیم تا پراسس ها بفهمند که میتوانند وارد ناحیه بحرانی شوند
        sleep(1);



    }
                                                                                    }

void reader(Semaphore* db,Semaphore* counter,Semaphore* mutex,Semaphore* rc)
{
    pid_t pid_new;
    pid_new = getpid();
    while (*counter<number) {

        down(mutex); چون متغیر آرسی بین پردازه ها مشترک است پس احتمال شرایط مسابقه برای ان وجود دارد به همین دلیل از سمافور برای هندل کردن آن استفاده میکنیم در اینجا مقدار سمافور را کم میکنیم و اگر شرایط را پردازه داشت وارد ناحیه بحرانی میشود
        *rc = *rc + 1;
        if (*rc == 1)
            down(db); سمافور را کم میکنیم تا رایتری همزمان وارد نشود
        up(mutex); میکنیم upسمافور متغیر مشترک آرسی را
        int temp = *counter;عملیت خواندن از کانتر
        printf("process number : %d ,value of counter : %d\n",pid_new,temp);
        down(mutex); برای تغییر میوتکس سمافور را کم میکنیم تا وارد ناحیه بحرانی شویم مانند بالا
        *rc = *rc - 1;
        if (*rc == 0)
            up(db); از ناحیه بحرانی خارج میشویم
        up(mutex);
        sleep(1);

                                                                                    }
```

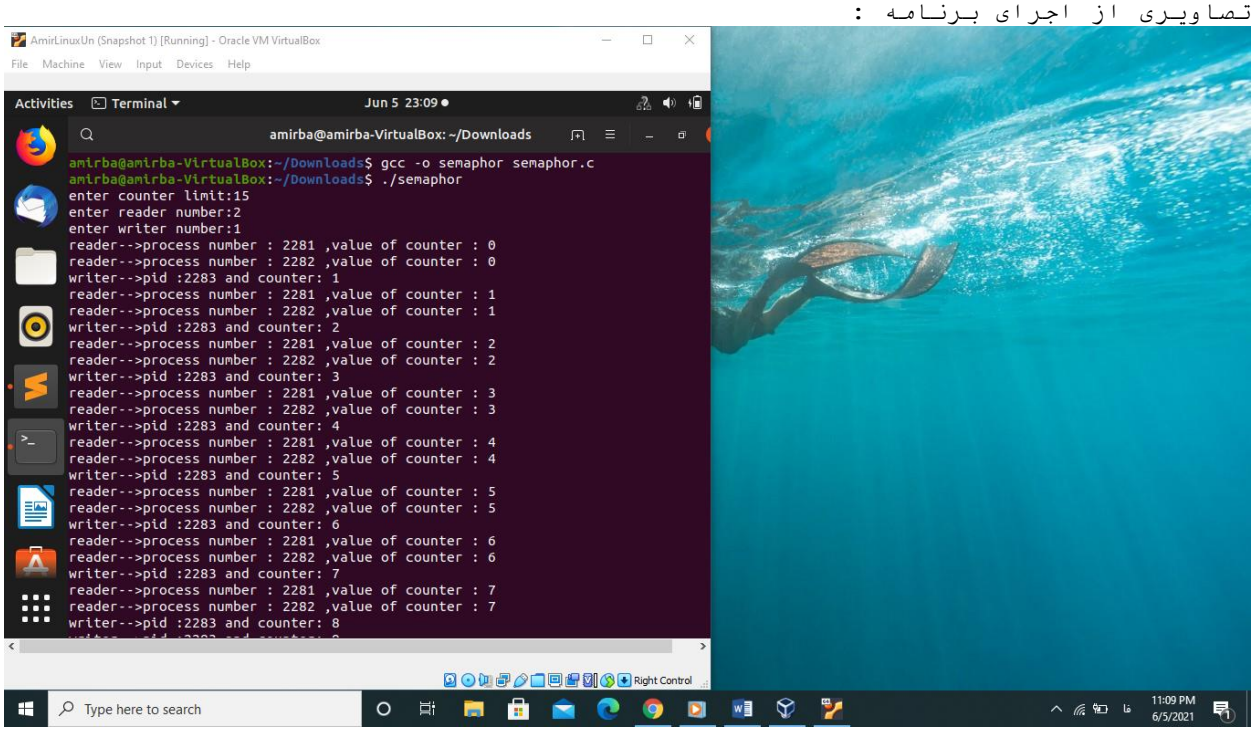
```
void down(Semaphore* s){
    while(*s <= 0);

    *s = *s - 1;}
```

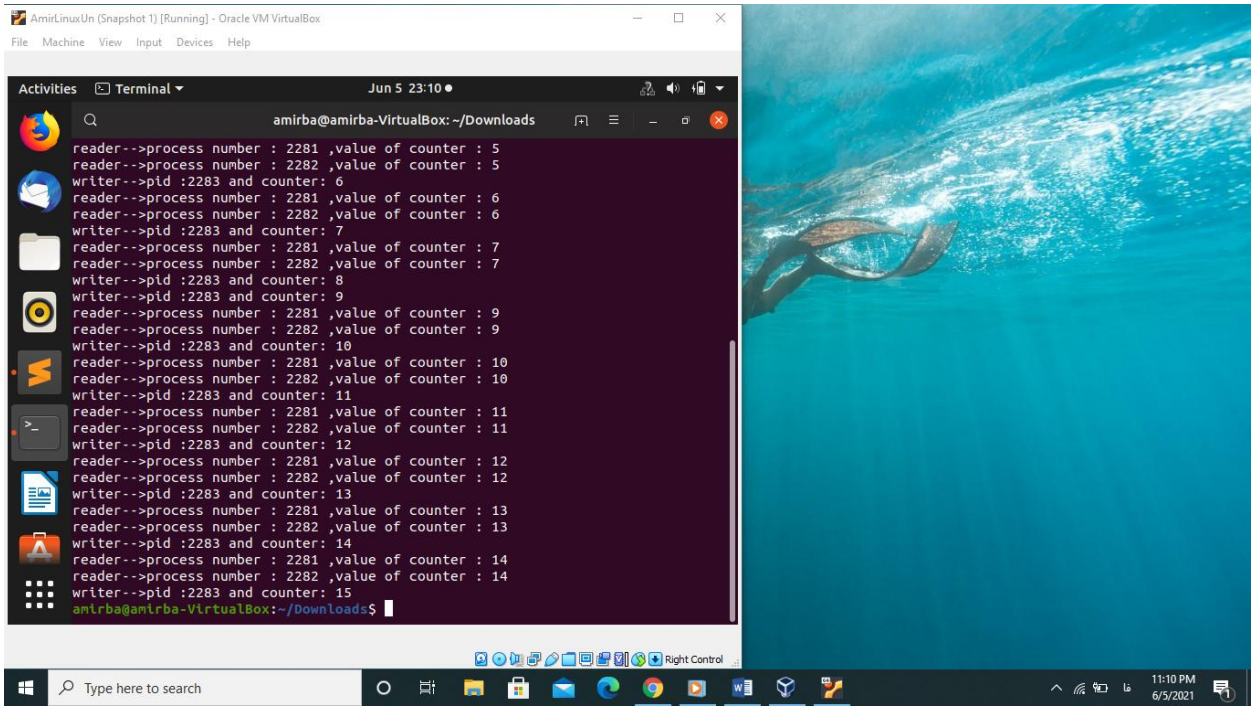
تابع down که وظیف کم کردن مقدار سمافور را دارد به این صورت کار میکند که اگر مقدار سمافور کمتر مساوی صفر باشد در حلقه گیر کرده و نمیگذارد که پردازه از ان خارج شود تا کاره پردازه های دیگر در ناحیه بحرانی تمام شود سپس یک واحد از ان دوباره کم میکند تا پردازه های دیگر نتوانند وارد شوند

```
void up(Semaphore* s){
    *s = *s + 1;
}
```

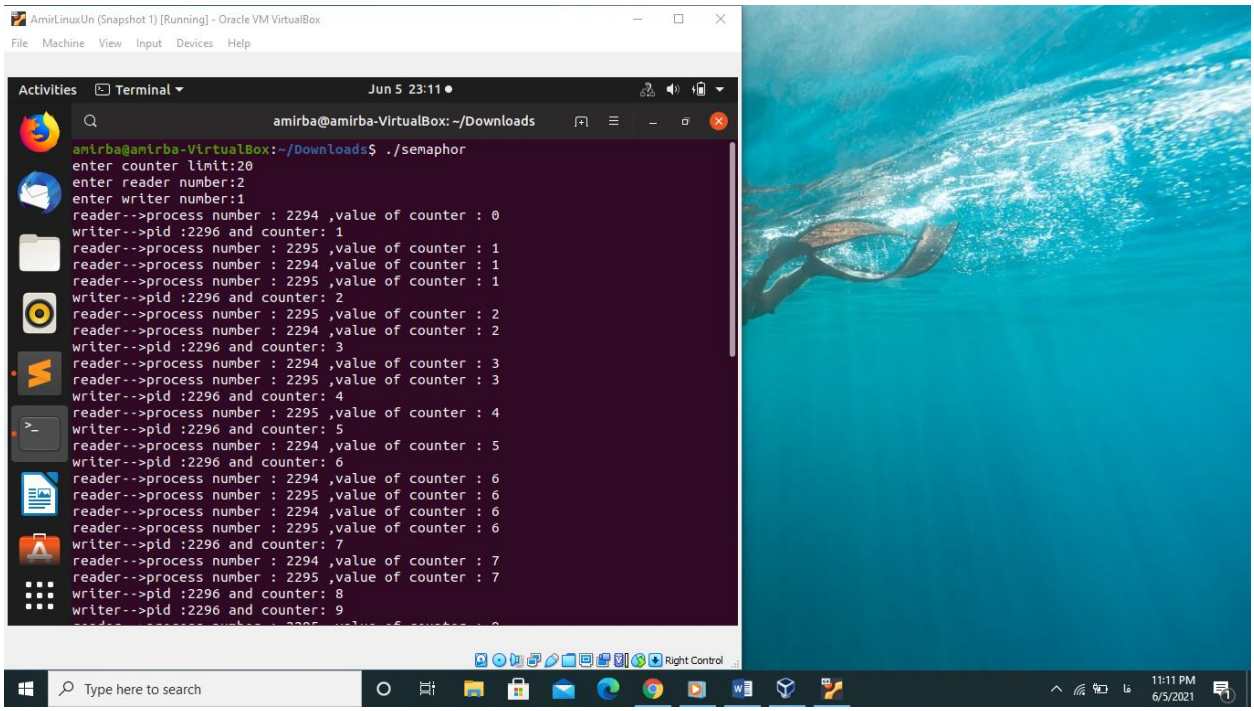
این تابع نیز مقدار سمافور را یکی زیاد کرده تا پردازه های دیگر متوجه شوند که کار این پردازه در ناحیه بحرانی تمام شده است

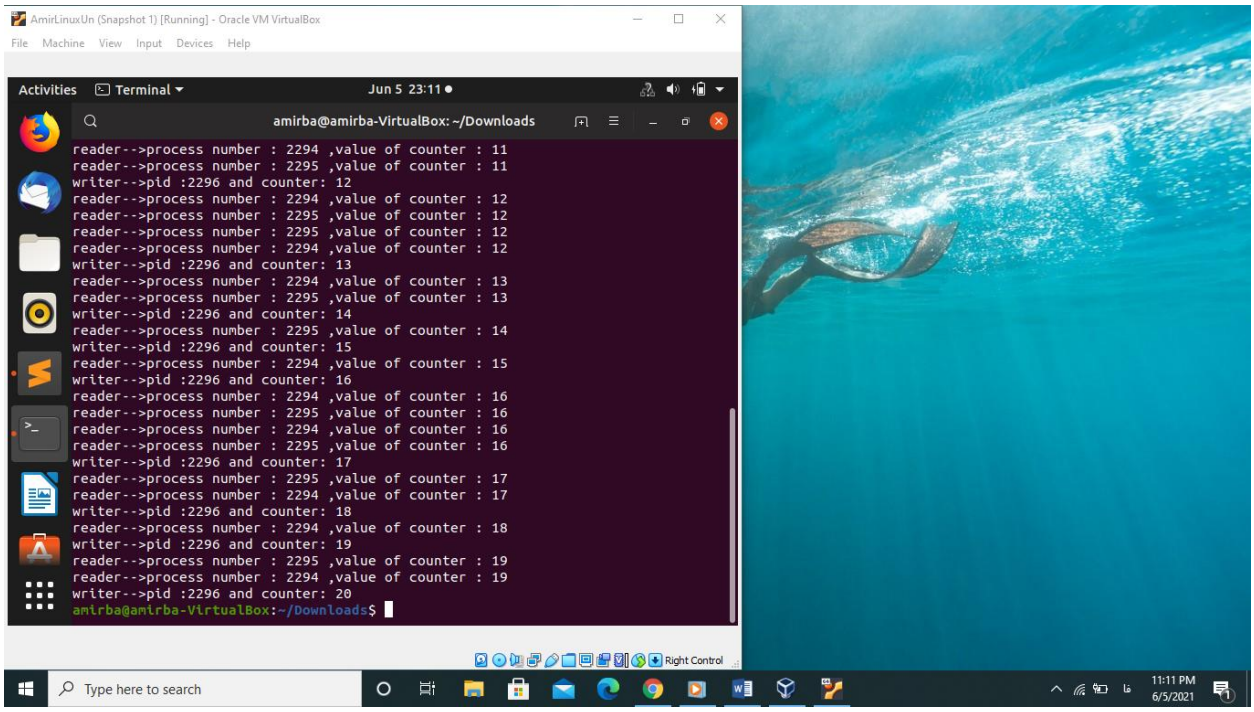
تصاویری از اجرای برنامه :




AmirLinuxUn (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Jun 5 23:10
amirba@amirba-VirtualBox: ~/Downloads
reader-->process number : 2281 ,value of counter : 5
reader-->process number : 2282 ,value of counter : 5
writer-->pid :2283 and counter: 6
reader-->process number : 2281 ,value of counter : 6
reader-->process number : 2282 ,value of counter : 6
writer-->pid :2283 and counter: 7
reader-->process number : 2281 ,value of counter : 7
reader-->process number : 2282 ,value of counter : 7
writer-->pid :2283 and counter: 8
writer-->pid :2283 and counter: 9
reader-->process number : 2281 ,value of counter : 9
reader-->process number : 2282 ,value of counter : 9
writer-->pid :2283 and counter: 10
reader-->process number : 2281 ,value of counter : 10
reader-->process number : 2282 ,value of counter : 10
writer-->pid :2283 and counter: 11
reader-->process number : 2281 ,value of counter : 11
reader-->process number : 2282 ,value of counter : 11
writer-->pid :2283 and counter: 12
reader-->process number : 2281 ,value of counter : 12
reader-->process number : 2282 ,value of counter : 12
writer-->pid :2283 and counter: 13
reader-->process number : 2281 ,value of counter : 13
reader-->process number : 2282 ,value of counter : 13
writer-->pid :2283 and counter: 14
reader-->process number : 2281 ,value of counter : 14
reader-->process number : 2282 ,value of counter : 14
writer-->pid :2283 and counter: 15
amirba@amirba-VirtualBox:~/Downloads$
Right Control
Type here to search
11:10 PM
6/5/2021

}


AmirLinuxUn (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Jun 5 23:11
amirba@amirba-VirtualBox: ~/Downloads
amirba@amirba-VirtualBox:~/Downloads$ ./semaphor
enter counter limit:20
enter reader number:2
enter writer number:1
reader-->process number : 2294 ,value of counter : 0
writer-->pid :2296 and counter: 1
reader-->process number : 2295 ,value of counter : 1
reader-->process number : 2294 ,value of counter : 1
reader-->process number : 2295 ,value of counter : 1
writer-->pid :2296 and counter: 2
reader-->process number : 2295 ,value of counter : 2
reader-->process number : 2294 ,value of counter : 2
writer-->pid :2296 and counter: 3
reader-->process number : 2294 ,value of counter : 3
reader-->process number : 2295 ,value of counter : 3
writer-->pid :2296 and counter: 4
reader-->process number : 2295 ,value of counter : 4
writer-->pid :2296 and counter: 5
reader-->process number : 2294 ,value of counter : 5
writer-->pid :2296 and counter: 6
reader-->process number : 2294 ,value of counter : 6
reader-->process number : 2295 ,value of counter : 6
reader-->process number : 2294 ,value of counter : 6
reader-->process number : 2295 ,value of counter : 6
writer-->pid :2296 and counter: 7
reader-->process number : 2294 ,value of counter : 7
reader-->process number : 2295 ,value of counter : 7
writer-->pid :2296 and counter: 8
writer-->pid :2296 and counter: 9
Right Control
Type here to search
11:11 PM
6/5/2021



قسمت دوم)

فلاسفه :

در این قسمت باید به ازای هر فیلسوف یک ترد ساختیم به صورتی که هر فیلسوف بتواند هم فکر کند و هم غذا بخورد

ولی مشکلی که وجود داشت نبود تعداد چنگال های کافی بود

برای حل این مشکل از سمافور استفاده کردیم مانند قسمت قبل به این صورت که برای هر چنگال یک mutex قرار دادیم که در ایتدا برای همه چنگال ها برابر با یک است و زمانی که ان چنگال توسط یک فیلسوف استفاده میشود ان چنگال را در حالت down or wait قرار میدهیم

و زمانی که کار فیلسوف مورد نظر با ان چنگال تمام شد ان را در حالت up or signal قرار میدهیم به این ترتیب هیچگاه یک چنگال توسط دو فیلسوف همزمان استفاده نمیشود

هر فرایند از پنج مرحله تشکیل شده است :

مرحله think: در این مرحله فیلسوف در حال فکر کردن یا انتظار است

مرحله pick_up : در این مرحله با توجه به شماره فیلسوف چنگال های چپ و راست ان مورد استفاده قرار میگیرد و این چنگال ها به حالت down میروند

مرحله eat: در این مرحله فیلسوف مشغول غذا خوردن است

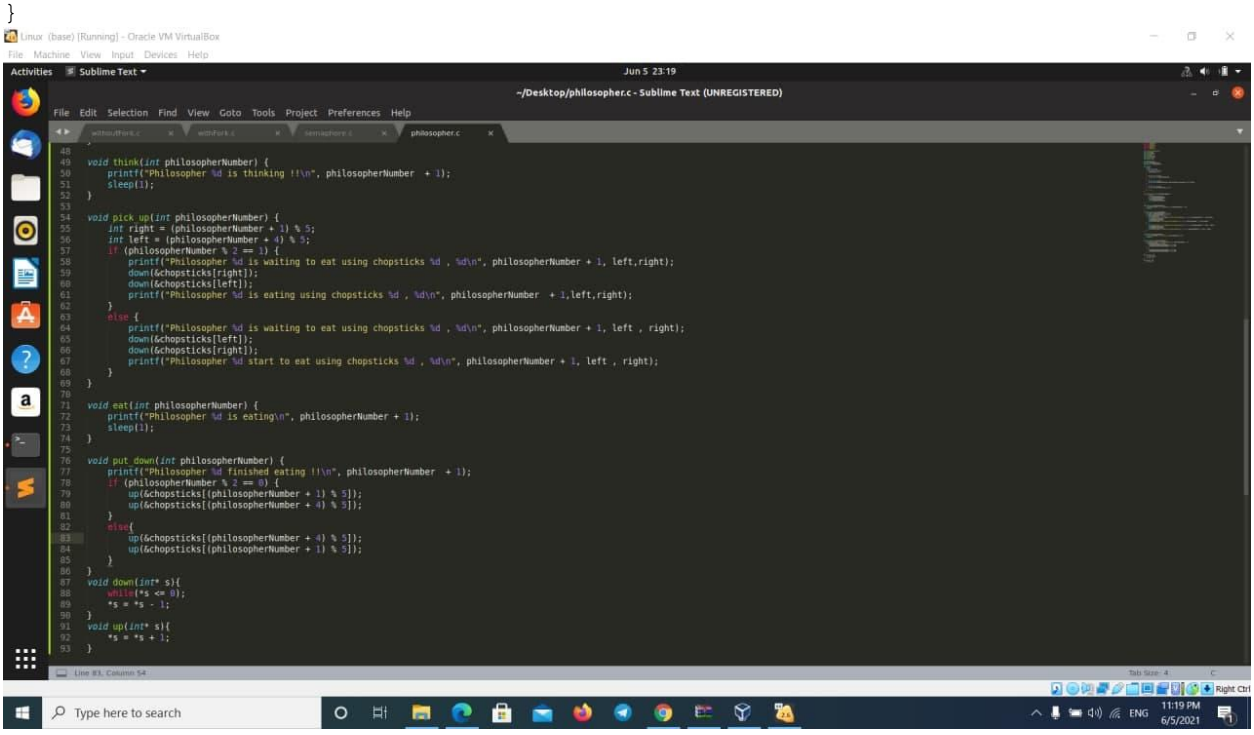
مرحلهput _down:در این مرحله با توجه به شماره فیلسوف بعد از اتمام غذا خوردن چنگال های چپ و راست فیلسوف ازاد میشوند و به حالت upمیروند

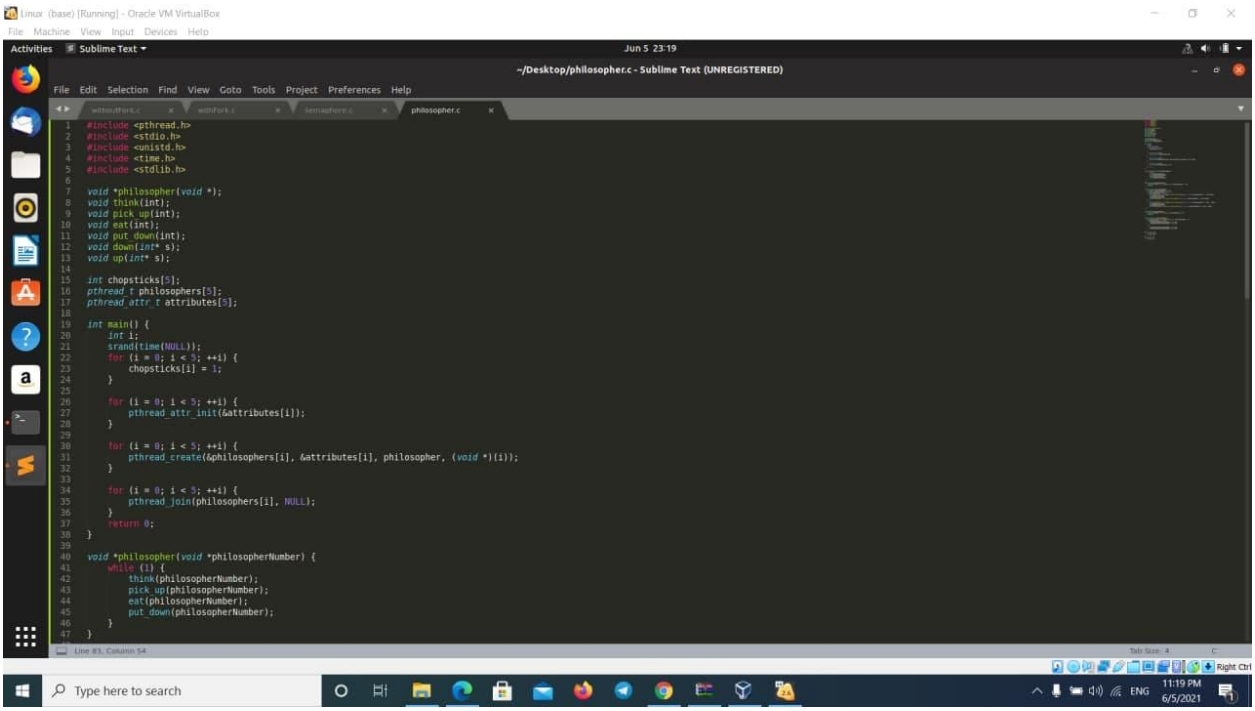
تمامی این مراحل برای هر فرایند مشخص داخل یک whileبی نهایت قرار دارد و دایما تکرار میشوند

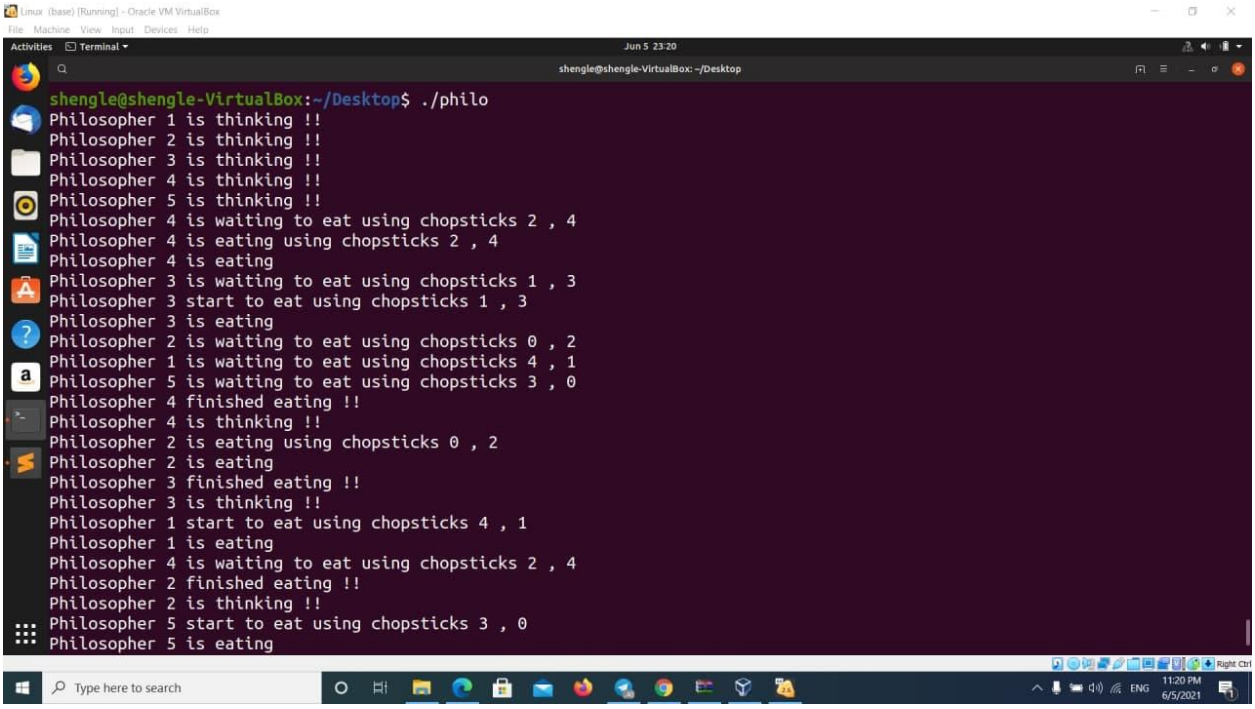
```
    #include <pthread.h>
#include <stdio.h>
#include <unistd.h>
#include <time.h>
#include <stdlib.h>
 هدر های مورد نظر را فراخوانی میکنیم

void *philosopher(void *);
void think(int);
void pick_up(int);
void eat(int);
void put_down(int);
void down(int* s);
void up(int* s);نوابغ مورد نیاز برای هر فیلسوف


int chopsticks[5];                برای چنگال ها mutex
pthread_t philosophers[5];              نرد برای فلاسفه
pthread_attr_t attributes[5];

int main() {
        int i;

        for (i = 0; i < 5; ++i) {
                chopsticks[i] = 1;
        }
                        قرار دادن همه میوتکس ها برابر با یک
        for (i = 0; i < 5; ++i) {
                pthread_attr_init(&attributes[i]);
        }

        for (i = 0; i < 5; ++i) {
                pthread_create(&philosophers[i], &attributes[i], philosopher,
(void *)(i));
        }

 وصل کردن هر ترد به تابع فیلسوفر

        for (i = 0; i < 5; ++i) {
                pthread_join(philosophers[i], NULL);
```

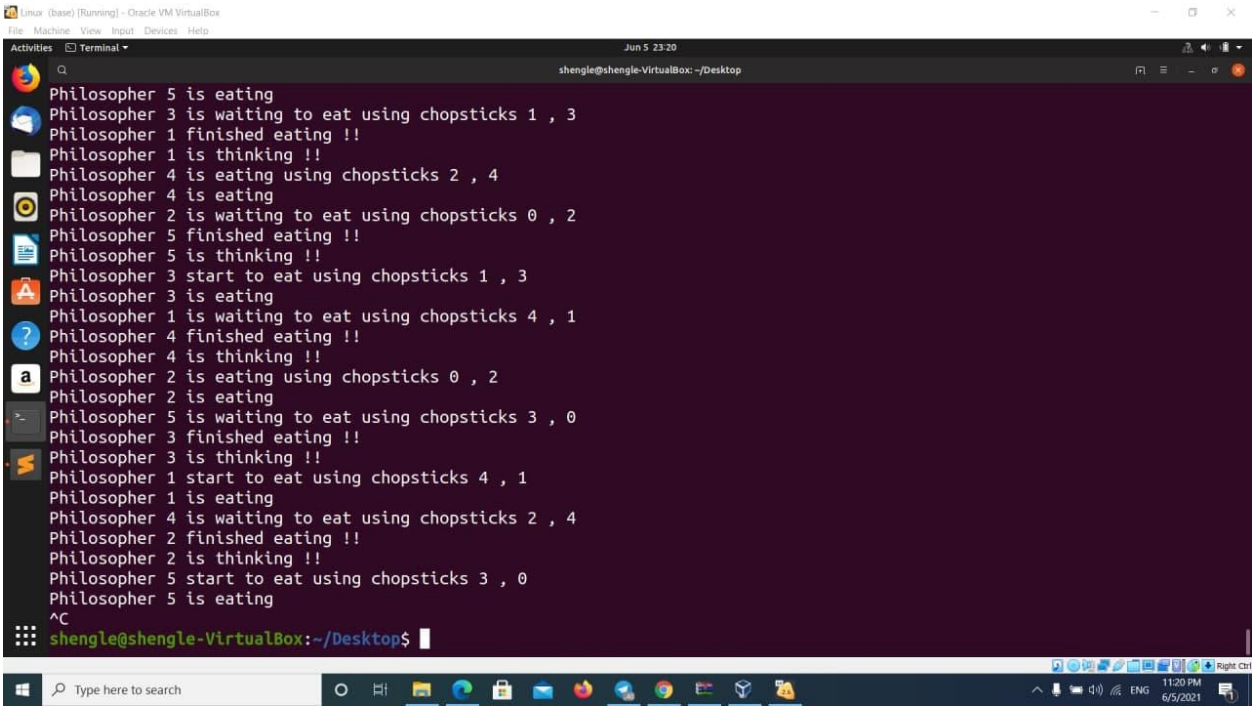
```
        }
        return 0;
}

void *philosopher(void *philosopherNumber) { قرار دادن همه مراحل یک فیلسوف
در حلقه بی نهایت
        while (1) {
                think(philosopherNumber);
                pick_up(philosopherNumber);
                eat(philosopherNumber);
                put_down(philosopherNumber);
        }
}

void think(int philosopherNumber) {
مدت زمانی که فیلسوف فکر میکند
        printf("Philosopher %d is thinking !!\n", philosopherNumber  + 1);
        sleep(1);
}

void pick_up(int philosopherNumber) {
 پیدا کردن چنگال چپ و راست هر فیلسوف
        int right = (philosopherNumber + 1) % 5;
        int left = (philosopherNumber + 4) % 5;
        if (philosopherNumber % 2 == 1) {
                printf("Philosopher %d is waiting to eat using chopsticks %d ,
%d\n", philosopherNumber + 1, left,right);
                down(&chopsticks[right]);
                down(&chopsticks[left]);
                printf("Philosopher %d is eating using chopsticks %d , %d\n",
philosopherNumber  + 1,left,right);
        }
        else {
                printf("Philosopher %d is waiting to eat using chopsticks %d ,
%d\n", philosopherNumber + 1, left , right);
                down(&chopsticks[left]);
                down(&chopsticks[right]);
                printf("Philosopher %d start to eat using chopsticks %d ,
%d\n", philosopherNumber + 1, left , right);
        }
}

void eat(int philosopherNumber) {
مدت زمانی که فیلسوف غذا میخورد
        printf("Philosopher %d is eating\n", philosopherNumber + 1);
        sleep(1);
}

void put_down(int philosopherNumber) {
با توجه به شماره فلسوف چنگال های برداشته شده ازاد میشوند
        printf("Philosopher %d finished eating !!\n", philosopherNumber  + 1);
        if (philosopherNumber % 2 == 0) {
                up(&chopsticks[(philosopherNumber + 1) % 5]);
                up(&chopsticks[(philosopherNumber + 4) % 5]);
        }
        else{
                up(&chopsticks[(philosopherNumber + 4) % 5]);
```

```
                up(&chopsticks[(philosopherNumber + 1) % 5]);
        }
}
void down(int* s){
        while(*s <= 0);
        *s = *s - 1;
}
void up(int* s){
        *s = *s + 1;
}
```




Linux (base) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Sublime Text
Jun 5 23:19
~/Desktop/philosopher.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
withoutFork.c
withFork.c
semaphore.c
philosopher.c
#include <pthread.h>
#include <stdio.h>
#include <unistd.h>
#include <time.h>
#include <stdlib.h>
void *philosopher(void *);
void think(int);
void pick_up(int);
void eat(int);
void put_down(int);
void down(int* s);
void up(int* s);
int chopsticks[5];
pthread_t philosophers[5];
pthread_attr_t attributes[5];
int main() {
int i;
srand(time(NULL));
for (i = 0; i < 5; ++i) {
chopsticks[i] = 1;
}
for (i = 0; i < 5; ++i) {
pthread_attr_init(&attributes[i]);
}
for (i = 0; i < 5; ++i) {
pthread_create(&philosophers[i], &attributes[i], philosopher, (void *)(i));
}
for (i = 0; i < 5; ++i) {
pthread_join(philosophers[i], NULL);
}
return 0;
}
void *philosopher(void *philosopherNumber) {
while (1) {
think(philosopherNumber);
pick_up(philosopherNumber);
eat(philosopherNumber);
put_down(philosopherNumber);
}
}
Line 83, Column 54
Tab Size: 4
C
Right Ctrl
Type here to search
ENG
11:19 PM
6/5/2021


Linux (base) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Jun 5 23:20
shengle@shengle-VirtualBox: ~/Desktop
shengle@shengle-VirtualBox:~/Desktop$ ./philo
Philosopher 1 is thinking !!
Philosopher 2 is thinking !!
Philosopher 3 is thinking !!
Philosopher 4 is thinking !!
Philosopher 5 is thinking !!
Philosopher 4 is waiting to eat using chopsticks 2 , 4
Philosopher 4 is eating using chopsticks 2 , 4
Philosopher 4 is eating
Philosopher 3 is waiting to eat using chopsticks 1 , 3
Philosopher 3 start to eat using chopsticks 1 , 3
Philosopher 3 is eating
Philosopher 2 is waiting to eat using chopsticks 0 , 2
Philosopher 1 is waiting to eat using chopsticks 4 , 1
Philosopher 5 is waiting to eat using chopsticks 3 , 0
Philosopher 4 finished eating !!
Philosopher 4 is thinking !!
Philosopher 2 is eating using chopsticks 0 , 2
Philosopher 2 is eating
Philosopher 3 finished eating !!
Philosopher 3 is thinking !!
Philosopher 1 start to eat using chopsticks 4 , 1
Philosopher 1 is eating
Philosopher 4 is waiting to eat using chopsticks 2 , 4
Philosopher 2 finished eating !!
Philosopher 2 is thinking !!
Philosopher 5 start to eat using chopsticks 3 , 0
Philosopher 5 is eating
Right Ctrl
Type here to search
ENG
11:20 PM
6/5/2021


Linux (base) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Jun 5 23:20
shengle@shengle-VirtualBox: ~/Desktop
Philosopher 5 is eating
Philosopher 3 is waiting to eat using chopsticks 1 , 3
Philosopher 1 finished eating !!
Philosopher 1 is thinking !!
Philosopher 4 is eating using chopsticks 2 , 4
Philosopher 4 is eating
Philosopher 2 is waiting to eat using chopsticks 0 , 2
Philosopher 5 finished eating !!
Philosopher 5 is thinking !!
Philosopher 3 start to eat using chopsticks 1 , 3
Philosopher 3 is eating
Philosopher 1 is waiting to eat using chopsticks 4 , 1
Philosopher 4 finished eating !!
Philosopher 4 is thinking !!
Philosopher 2 is eating using chopsticks 0 , 2
Philosopher 2 is eating
Philosopher 5 is waiting to eat using chopsticks 3 , 0
Philosopher 3 finished eating !!
Philosopher 3 is thinking !!
Philosopher 1 start to eat using chopsticks 4 , 1
Philosopher 1 is eating
Philosopher 4 is waiting to eat using chopsticks 2 , 4
Philosopher 2 finished eating !!
Philosopher 2 is thinking !!
Philosopher 5 start to eat using chopsticks 3 , 0
Philosopher 5 is eating
^C
shengle@shengle-VirtualBox:~/Desktop$
Type here to search
ENG
11:20 PM
6/5/2021