پاسخ تکلیف چهارم سیستم عامل

سوال ۱

میزبان یک مهمانی، N >2 نفر مهمان را به خانه دعوت کرده است. میزبان نمی خواهد چندین دفعه در خانه را برای ورود میزبانان باز کند. N مهمان برای یکدیگر صبر می کنند و به یکباره وارد می شوند. میزبان و مهمانان با برنامه چند نخی پیاده سازی می شوند.

میزبان برای ورود همه مهمانان صبر می کند سپس openDoor() را فراخوانی می کند و یک متغییر شرطی را سیگنال میکند. مهمانان باید برای ورود Nنفر و باز شدن در صبر کنند و enterHouse() را فراخوانی کنند.

تنها با استفاده از متغیرهای تعریف شده در کد میزبان، کد مهمان را بنویسید.

```
//host
lock(m)
while (guest_count < N) wait(cv_host, m)
openDoor()
signal(cv_guest)
unlock(m)
```

کد مهمان:

```
lock(m)
guest_count++
if (guest_count == N)
  signal(cv_host)
wait(cv_guest, m)
signal(cv_guest)
unlock(m)
enterHouse()
```

سوال ۲

مسئله خوانندگان نویسندگان را با اولویت نویسندگان به خوانندگان در نظر بگیرید. می خواهیم lockهای مخصوصی با نامreader-writer locks داشته باشیم که بتوان در چنین شرایطی از آنها استفاده کرد.

سودوکدهای لازم را برای پیاده سازی توابع writeLock، readUnlock، readLock و writeUnlock بنویسید.

در پیاده سازی خود از mutex و conditional variable استفاده کنید(از سمافور استفاده نکنید).

پاسخ:

```
readLock:

lock(mutex)
while (writer_present || writers_waiting > 0)
  wait(reader_can_enter, mutex)
readcount++
unlock(mutex)
```

```
readUnlock:

lock(mutex)
readcount--
if (readcount == 0)
  signal(writer_can_enter)
unlock(mutex)
```

```
writeLock:

lock(mutex)
writer_waiting++
while(readcount > 0 || writer_present)
    wait(writer_can_enter, mutex)
writer_waiting--
writer_present = true
unlock(mutex)
```

```
writeUnlock:

lock(mutex)
writer_present = false
if (writer_waiting == 0)
  signal_broadcast(reader_can_enter)
else
  signal(writer_can_enter)
unlock(mutex)
```

سوال ۳

میخواهیم گذر ماشین ها از روی یک پل یک طرفه را مدیریت کنیم به صورتی که تا وقتی ماشین هایی در حال عبور از پل از یک سمت هستند به ماشین های دیگر که از سمت دیگر قصد ورود دارند اجازه ی ورود داده نشود .محدودیتی روی تعداد ماشین هایی که روی پل هستند وجود ندارد. اما میخواهیم مسئله ی گرسنگی را تا حدی حل کنیم.

بدین منظور اگر۵ماشین از سمت شمال از پل رد شدند و تعدادی(یک یا بیشتر) ماشین در سمت جنوب قصد ورود به پل را داشتند ، از ادامه عبور ماشین ها از سمت شمال جلوگیری کرده تا ماشین هایی از سمت جنوب از پل رد شوند. همین قانون برای سمت جنوب هم به صورت برعکس برقرار است.

با تابع north() ماشینی که در شمال پل است ، قصد ورود به پل را کرده و از پل رد میشود . همچنین با تابع south() ماشین های سمت جنوبر از پل رد میشوند. سمافور یا میوتکس های مورد نیاز برای حل مسئله را تعریف کرده و دو تابع نامبرده را با استفاده از آنها پیاده سازی کنید.

پاسخ: تابع north هم به صورت مشابه و متقارن نوشته میشود.

```
Semaphor mutex = 1, south_permit = 0, north_permit = 0;
int north_passing = 0, north_passed = 0, north_waiting = 0;
int south_passing = 0, south_passed = 0, south_waiting = 0;

void south()
{
    wait(mutex);
    if (north_passing == 0 && north_waiting == 0)
    {
        south_passing++;
        signal(south_permit);
    }
    else if (north_waiting > 0 && north_passing == 0 && south_passing + south_passed < 5)
    {
        south_passing++;
        signal(south_permit);
    }
    else
        south_waiting++;

    signal(mutex);
    wait(south_permit);

    // pass the bridge

    wait(mutex);
    south_passing--;
    south_passed++;

    if (south_passing == 0 && south_waiting == 0 && north_waiting > 0)
    {
        south_passed = 0;
        n = min(5, north_waiting);

        for (i = 0; i < n; i++)
        {
            signal(north_permit);
            north_passing++;
            north_waiting--;
        }
    }
    else if (north_waiting == 0 && north_passing == 0 && south_waiting > 0)
    {
        signal(south_permit);
        south_waiting--;
        south_passing++;
    }

    signal(mutex);
}
```

سوال ۴

یک غار تاریخی با نقاشی‌های دیواری باشکوه دارای ورودی بسیار باریکی است که تنها می‌تواند به یک بازدیدکننده اجازه ورود/خروج در هر زمان بدهد.

برای اطمینان از اینکه هرگز بیش از 15 نفر در غار نباشند، شبه کد روبرو را تکمیل کنید. کد توسط هر فرآیند بازدید کننده غار اجرا می شود. بخش های مشخص شده را با مقادیر اولیه سمافور یا فراخوانی های سمافور (wait() یا signal()) پرکنید.

```
semaphore s1 = // your answer here;
semaphore s2 = // your answer here;

Cave_exploration ( ) {
// your answer here(may contain one or two calls)
Enter_the_cave ( );
// your answer here(may contain one or two calls)
Look_at_the_paintings ( );
// your answer here(may contain one or two calls)
Exit_the_cave ( );
// your answer here(may contain one or two calls)
}
```

پاسخ:

```
semaphore s1 = 1; // Semaphore to control access to the cave entrance
semaphore s2 = 15; // Semaphore to limit the number of people inside the cave

procedure Cave_exploration():
    Enter_the_cave();
    Look_at_the_paintings();
    Exit_the_cave();

procedure Enter_the_cave():
    wait(s1); // Acquire the semaphore to enter the cave
    wait(s2); // Wait if the maximum number of people inside the cave is reached
    signal(s1); // Release the semaphore to allow the next visitor to enter

procedure Look_at_the_paintings():
    // Critical section: Actions inside the cave (e.g., admire paintings)

procedure Exit_the_cave():
    wait(s1);  //Acquire the semaphore to exit the cave
    signal(s2); // Release the semaphore to indicate that a visitor has left the cave
    signal(s1); // Release the semaphore to allow the next visitor to pass
```

موفق باشید :)