

شروع	دوشنبه، 22 آذر 1400، 6:29 عصر
وضعیت	پایان یافته
پایان	دوشنبه، 22 آذر 1400، 6:59 عصر
زمان صرف شده	30 دقیقه 1 ثانیه
نمره	10.00 از 10.00 (100%)

استفاده از spin lock جهت جلوگیری از دسترسی همزمان به داده‌های کرنل، در هیچ شرایطی روش مناسبی نیست زیرا باعث busy waiting و کاهش بهره وری CPU میشود

یک گزینه را انتخاب کنید:

☐ صحیح

☒ غلط ✓

سؤال 1

درست

نمره 2.00 از 2.00

پاسخ درست گزینه «غلط» است.

## سؤال 2

درست

نمره 2.00 از 2.00

در مورد راه حل نرم افزاری زیر که جهت حل مسئله ناحیه بحرانی پیشنهاد شده است، کدام جمله یا جملات صحیح است مقدار اولیه flagها false است.

g\*/

```
p1{  
  
    while(true){  
  
        flag[1]=true;  
  
        while(flag[0]); /* do nothing */  
  
        /*critical_section*/  
  
        flag[1]=false;  
  
        /* remainder_section */  
  
    }  
  
}
```

☒ a. امکان رخداد بن بست وجود دارد ✓

☐ b. شرط انحصار متقابل برآورده شده ولی پیشرفت وجود ندارد

c. ☐ شرط انحصار متقابل برقرار نشده است

d. ☒ امکان رخداد گرسنگی بدین معنی که یکی از پروسسها نتواند وارد CS شود در حالیکه پروسس دیگر متناوبا میتواند وارد شود، وجود دارد.

پاسخ شما صحیح می باشد

پاسخ درست «امکان رخداد بن بست وجود دارد» است.

### سؤال 3

کامل

نمره 6.00 از 6.00

یک اتوبوس با ظرفیت  $k$  مسافر هر بار، مسافرانی را از یک ایستگاه اتوبوس سوار می‌کند. هر وقت اتوبوس به ایستگاه می‌رسد، حداکثر به  $k$  مسافر منتظر اجازه سوارشدن می‌دهد و پس از آن راه می‌افتد. مسافران باید صبر کنند تا اتوبوس برسد و سوار شوند. مسافرانی که بعد از رسیدن اتوبوس به ایستگاه، به ایستگاه می‌رسند، نباید بتوانند سوار شوند و باید برای دفعه بعدی در ایستگاه منتظر بمانند. یک thread برای سوارشدن مسافران وجود دارد که هر بار که مسافری سوار می‌شود تابع `savar` را فراخوانی می‌کند و یک thread برای اتوبوس وجود دارد که هر بار بعد از اینکه تعداد موردنظر مسافران را سوار کرد، با فراخوانی `harekat` شروع به حرکت می‌کند. در ادامه سمافور و متغیرهای لازم برای حل مسئله آمده است

```
semaphore mutex = 1
```

```
semaphore bus_arrived = 0
```

```
semaphore mosافر_savar = 0
```

```
int waiting_mosافر = 0
```

همچنین کد thread سوارشدن مسافران نیز به صورت زیر نوشته شده است. با استفاده از فقط همین سمافورها و متغیرهایی که تعریف شده و با توجه به کد thread سوارشدن مسافران، کد thread حرکت اتوبوس را بنویسید .

```
lock(mutex)
```

```
waiting_mosافر ++
```

```
unlock(mutex)
```

```
lock(bus_arrived)
```

```
savar()
```

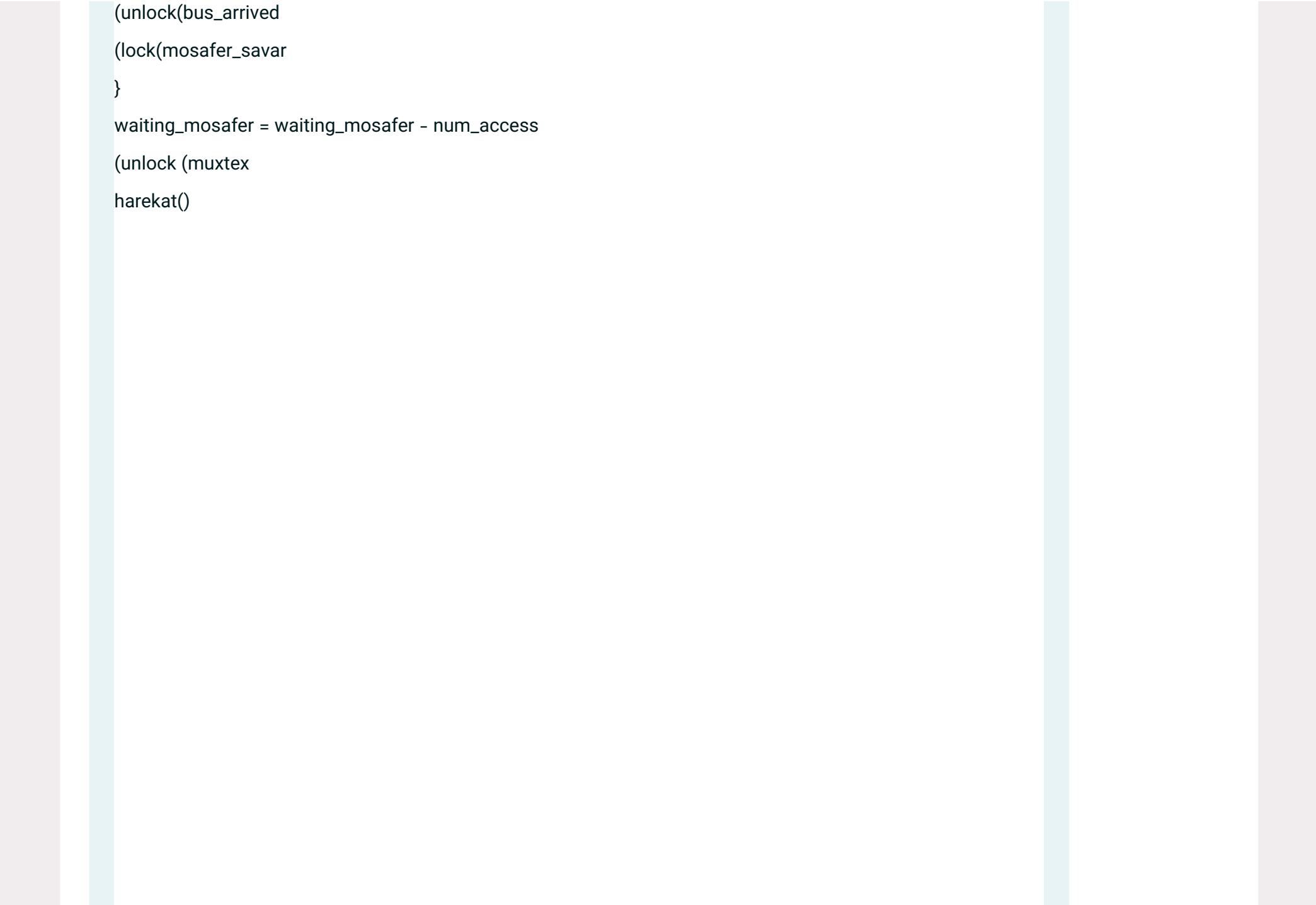
```
unlock(mosافر_savar)
```

```
(lock (mutex
```

```
(num_access = min(k,waiting_mosافر
```

```
(++for (i=0;i<num_access;i
```

```
{
```



```
(unlock(bus_arrived
(lock(mosafer_savar
}
waiting_mosafer = waiting_mosafer - num_access
(unlock (mutex
harekat()
```

```
lock(mutex)
m = min(waiting_mosafer, k)
for(i=0;i<m;i++){
    unlock(buss_arrived)
    lock(mosafer_savar)
    waiting_mosafer - -;
}
unlock(mutex)
harekat()
```

(اینجا lock و unlock منظور همان wait و signal است که از انجایی که تعریفها سمافور بوده مشخص است. )

دیدگاه:

