«به نام خدا »

درس سیستم‌های عامل

تمرین سوم تئوری

مدرس : دکتر انتظاری

طراح : الناز رضایی

قوانین

دانشجویان محترم لطفاً نکات تکمیلی زیر را در تمامی تمرین‌ها در نظر بگیرید.

•در مجموع تمام تمرین‌ها، ۲۴۰ ساعت تأخیر در ارسال پاسخ‌ها مجاز است به همین جهت زمان اعلام شده به هیچ وجه قابل تمدید نیست و در صورت نیاز می‌توانید از این زمان استفاده کنید، با پایان یافتن این زمان نمره شما ۰ لحاظ خواهد شد.

•پروژه های درسی مهلت تاخیر ندارند .

•انجام تمرین و کوییز به صورت یک‌نفره می‌باشد.

• فرمت ارسالی تمرین تئوری حتماً باید به صورت latex یا word باشد.

• در صورت مشورت کردن در مورد سؤال خاصی ذکر اسامی همراه با سؤال موردنظر، در گزارش کار طرفین الزامی هست غیر این صورت مشاهده هرگونه تشابه در گزارش کار یا کدهای پیاده‌سازی، به منزله تقلب برای طرفین در نظر گرفته می‌شود.

• تمامی منابع استفاده شده به صورت دقیق ذکر شوند . همچنین استفاده از کدهای آماده بدون ذکر منبع و بدون تغییر به منزله تقلب خواهد بود و نمره تمرین شما صفر در نظر گرفته می‌شود .

•بخش زیادی از نمره شما مربوط به گزارش کار و روند حل مسئله است. لطفاً به موارد خواسته شده به صورت کامل، دقیق و بدون ابهام پاسخ دهید.

•لطفاً گزارش، فایل کدها و سایر ضمائم موردنیاز را با فرمت زیر در سامانه مدیریت دروس بارگذاری نماید به جای n شماره تمرین قرار داده می‌شود HWn_[studentNumber].zip .

•درصد تشابه بالای ۳۰ درصد از دید quera تقلب لحاظ می‌شود و بعد از تکرار برای بار دوم، نمره این تمرین و تمرین قبلی که تقلب آن گرفته نشده، ۰ لحاظ خواهد شد.

•زمان ارسال کوییزهای کلاس حل تمرین تا ساعت ۱۲ همان شب خواهد بود و این مورد به هیچ وجه تمدید نخواهد شد.

•فرض کوییزهای کلاس حل تمرین این است که این امتحان به صورت book open برگزار نمی‌شود و اجازه استفاده از کدهای آماده در سایت ها را نخواهید داشت.

1. دو پردازه همروند P_0 و P_1 کد زیر را اجرا می‌کنند و برای ورود به ناحیه بحرانی دارای رقابت هستند. i می‌تواند صفر یا یک را اختیار کند. آیا در کد زیر انحصار متقابل برقرار است؟ توضیح دهید.

```
bool flag[2];
int turn;
process i
        flag[i] = true
        while (turn != i){
                while (flag[j])
                ;
                Turn = I;
        }
        /* Critical Section */
        flag[i] = false;
```

2. راه‌حل زیر را برای مسئله‌ی انحصار متقابل شامل دو فرآیند P_0 و P_1 در نظر بگیرید. فرض کنید که مقدار اولیه turn، صفر می‌باشد. کد Process P_0 در ادامه آمده است.

```
/* Other code */
while (turn != 0) /* Do nothing and wait*/
/* Critical Section */
turn = 0;
/* Other code */
```

برای فرآیند P_1، 0 را با 1 در کد بالا عوض کنید. تعیین کنید که آیا راه‌حل ارائه شده تمام شرایط مورد نیاز برای یک راه‌حل انحصار متقابل صحیح را دارد یا خیر.

3. کد زیر را برای تخصیص (allocating) و آزادسازی (releasing) فرآیندها در نظر بگیرید.

```
while (true){
        while (true){
                flag[i] = want_in;
                j = turn;
                while (j != i){
                        if (flag[j] != idle)
                                j = turn;
                        else
                                j = (j + 1) % n;
                }
                flag[i] = in_cs;
                j = 0;
                while ((j < n) && (j == i || flag[j] != in_cs) )
                        j++;
                if ((j >= n ) && (turn == i || flag[turn] == idle))
                        break;
        }
        /* Critical Section */
        j = (turn + 1) % n;
        while (flag[j] == idle)
                j = (j + 1) % n;
        turn = j;
```

```
        flag[j] = idle;
        /* Remainder Section*/
}
```

الف) race condition(s) را شناسایی کنید.

ب) فرض کنید یک mutex lock با نام mutex با عملیات ()acquire و ()release دارید. محل قرار دادن قفل را برای جلوگیری از race condition(s) مشخص کنید.

پ) آیا می‌توانیم با جابه‌جا کردن integer variable (int number_of_processes = 0) با atomic integer (atomic_t number_of_processes =0) از race condition(s) جلوگیری کنیم؟

4. توضیح دهید که چرا interruptها برای اجرای اصول synchronization در سیستم‌های چند پردازنده‌ای مناسب نیستند.

5. توضیح دهید deadlock چگونه در مسئله dining-philosophers امکان‌پذیر است.

6. دو فرآیند P_1 و P_2 به صورت زیر هستند اگر فرآیندها بتوانند به صورت همروند اجرا شوند و امکان اجرای آن‌ها به صورت چند در میان نیز وجود داشته باشد، در صورتی که مقدار اولیه متغیر مشترک a برابر صفر باشد، بعد از اجرای کامل رو فرآیند، مقادیر a، b و c چه تغییری می‌کنند؟

P_1	P_2
b = a;	a = 2;
c = a + 1;	

موفق باشید. ☺