
# دانشکده مهندسی کامپیوتر

## درس سیستم‌های عامل

---

## تمرین سری سوم

---

مدرسان ..... دکتر رضا انتظاری ملکی، دکتر وحید ازهری

تیم طراح ..... عرفان زارع – هانا هاشمی

تاریخ انتشار ..... ۱۴۰۲/۰۹/۰۳

تاریخ تحویل ..... ۱۴۰۲/۰۹/۱۲

## در رابطه با تمرین

- نمره این تمرین از ۱۰۰ می‌باشد و بارم هر سوال روبه‌روی آن نوشته شده است.
- به هیچ وجه تمرینی را از دیگران **کپی** نکنید. در صورت مشاهده تقلب و کپی در تمرینات، نمره هر دو طرف **صفر** در نظر گرفته می‌شود.

**۱-** به سوالات زیر پاسخ دهید. **(۱۵ نمره)**

**الف)** تفاوت بین thread های سطح کاربر با thread های سطح kernel در چیست؟ تحت چه شرایطی بر یکدیگر برتری دارند؟

**ب)** هنگامی که یک thread ساخته می‌شود ، چه منابعی از سیستم را به خود اختصاص می‌دهد؟ این منابع اختصاص یافته با زمانی که وقتی یک process ساخته می‌شود چه تفاوتی دارد؟

**پ)** الگوریتم های زمان بندی FCFS,SRTF,RR را در نظر بگیرید .آیا گرسنگی (starvation) در هریک یا همه این الگوریتم‌ها ممکن است؟ در صورت وجود راه‌حل برای رفع این مشکل، آن را مطرح کنید.

**ت)** عمل switching بین دو یا چندین thread هزینه کمتری نسبت به process ها دارد. علت این امر چیست؟

**ث)** در مسئله غذاخوری فیلسوف‌ها، شرایطی را بیان کنید که منجر به گره (deadlock) شود.

**۲-** خروجی کد زیر در "Line A" چه خواهد بود؟ دلیل خود را با بررسی خط به خط کد بیان نمایید. **(۱۵ نمره)**

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
int value = 5;
int main(){
    pid_t pid;
    pid = fork();
    if (pid == 0) {
     value += 15;
     return 0;
    }
    else if (pid > 0) {
     wait(NULL);
     printf("PARENT: value = %d",value); /* Line A */
     return 0;
    }
}
```

۳- دو thread را مانند جدول زیر درنظر بگیرید که از یک حافظه مشترک استفاده می کنند. این دو thread به صورت همروند اجرا می شوند. مقدار اولیه x صفر است و هرباری که مقداری به x اضافه می شود، در یک ثبات ذخیره می شود. تمامی مقادیر احتمالی زمانی که دو thread کار خود را انجام دهند را بنویسید. **(۱۰ نمره)**

| Thread A | Thread B |
|---|---|
| `for i = 1 to 5 do`<br>`x = x + 1;` | `for j = 1 to 5 do`<br>`x = x + 1;` |

۴- ذهن خلاق و بهینه‌دوست شما که بسیار به بازی‌های کامپیوتری علاقه دارد و از محدودیت‌های پردازشی موجود نیز باخبر است، به تازگی با مفهوم پردازش موازی آشنا شده و از شما می پرسد چرا به جای تعداد محدود هسته‌های محاسباتی موجود در پردازنده‌های امروزی، از تعداد بیشتری هسته (مثلا ۱۰۰هسته!) استفاده نمی‌شود تا از سرعت‌های چند برابر بهره بگیریم؟ پاسخ شما چیست؟ (نرخ افزایش سرعت را برای ۲، ۴، ۸ و ۱۰۰هسته محاسبه کنید) **(۱۵ نمره)**

۵- با درنظرگرفتن مجموعه پردازه‌های زیر و با فرض اینکه پردازه ها به ترتیب P1,P2,P3,P4,P5 از زمان t=0 در دسترس CPU قرار گرفته باشند، به سوالات زیر پاسخ دهید: **(۲۵ نمره)**

| Process | Burst Time | Priority |
|---|---|---|
| $P_1$ | ۲ | ۲ |
| $P_2$ | ۱ | ۱ |
| $P_3$ | ۸ | ۴ |
| $P_4$ | ۴ | ۲ |
| $P_5$ | ۵ | ۳ |

**الف)** گانت چارت حاصل از اجرای هریک از الگوریتم‌های زمان‌بندی RR، FCFS با کوانتوم ۲، SJFو non-preemptive را رسم کنید. (فرض کنید پردازه‌ای که عدد اولویت آن بیشتر است، اولویت بالاتری دارد)

**ب)** مقدار زمان برگشت را برای هریک از پردازه‌ها در هریک از الگوریتم‌های زمان‌بندی قسمت (الف) محاسبه کنید.

**پ)** مقدار زمان انتظار را برای هریک از پردازه‌ها در هریک از الگوریتم‌های زمان‌بندی قسمت (الف) محاسبه کنید.

**ت)** کدام الگوریتم زمان‌بندی، کمترین میانگین زمان انتظار را برای تمامی پردازه‌ها دارد؟

**۶-** یک مسئله تولیدکننده – مصرف‌کننده با بافر محدود را در نظر بگیرید که در آن چندین تولیدکننده به صورت همزمان وجود دارند و اندازه آیتم‌های تولیدی و مصرفی با هم متفاوت است. ظرفیت کل بافر N است و این پردازه‌ها از دو سمافور شمارشی برای دسترسی به بافر استفاده می‌کنند. full دارای مقدار اولیه صفر و empty دارای مقدار اولیه N است. **(۲۰ نمره)**

هر تولیدکننده کد زیر را برای اضافه کردن آیتمی با اندازه k به بافر استفاده می‌کند:

```
for i from 1 to k do {
    P(empty)
}
//insert item of size k into buffer
V(full)
```

مصرف‌کننده نیز کد زیر را برای حذف آیتم از بافر اجرا می‌کند:

```
P(full)
//remove item of size k from the buffer
for i from 1 to k do {
    V(empty)
}
```

آیا با استفاده از این کد می‌توان اطمینان حاصل کرد که:

**الف)** تولیدکننده تا زمانی که در بافر ظرفیت کافی وجود نداشته باشد، آیتمی را اضافه نکند؟

**ب)** مصرف کننده تنها زمانی که آیتمی در بافر باشد اقدام به حذف کند؟

**پ)** تنها یک پردازه (تولیدکننده یا مصرف‌کننده) از بافر در یک زمان استفاده کند؟

**ت)** بن بست ایجاد نمی‌شود؟