آزمون میان‌ترم درس اصول سیستم‌های عامل

دانشکده ریاضی – دانشگاه صنعتی خواجه نصیرالدین طوسی

اردیبهشت ماه ۱۴۰۳ – مدت زمان آزمون: ۷۰ دقیقه

۱- به پرسش‌های زیر پاسخ دهید.

الف) منظور از اینکه سامانه‌ای دارای ویژگی graceful degradation است چیست؟

ب) به کمک یک مثال در حوزه سیستم‌عامل مفهوم جداسازی policy از mechanism را توضیح دهید.

پ) وقتی می‌گوییم که شیوه synchronization بین فرستنده و گیرنده از نوع rendezvous است، منظور چیست؟

ت) وقتی یک interrupt handler، یک interrupt را clear می‌کند، منظور چیست؟

ث) وظیفه دستور ()exec که گاهی پس از دستور ()fork اجرا می‌گردد، چیست؟

ج) دلیل استفاده از شیوه Direct Memory Access (DMA) در مدیریت I/O در سیستم‌عامل را شرح دهید.

چ) مفهوم cache coherency در یک محیط multiprocessing را توضیح دهید.

ح) نقش local buffer ها و register های متعلق به یک device controller چیست؟

خ) منظور از اینکه ساختار یک سیستم‌عامل از نوع monolithic است چیست؟ یک مزیت و یک عیب این نوع از ساختار را ذکر نمایید.

د) هدف از memory dump یک برنامه چیست و چه زمانی از آن استفاده می‌شود؟

ذ) توضیح دهید که منظور از Big-endian بودن معماری یک cpu چیست؟

ر) یک trap وقفه‌ای (interrupt) سخت‌افزاری است یا نرم‌افزاری؟ مثالی از یک trap ذکر نمایید.

ز) تفاوت بین مفهوم‌های program و process را توضیح دهید.

۲- بدون ذکر دلیل، صرفاً درستی یا نادرستی هر یک از موردهای زیر را معین نمایید.

الف) پیش‌نیاز استفاده از named pipes در ارتباط بین دو process این است که رابطه‌ای همانند parent-child بین آن دو process برقرار باشد. همچنین، named pipes قوی‌تر از ordinary pipes هستند.

ب) سرعت دسترسی به حافظه cache نسبت به register ها کمتر است ولی سرعت دسترسی به حافظه cache نسبت به main memory بیشتر است.

پ) هنگامی که ارتباط بین دو process با استفاده از شیوه shared memory صورت می‌گیرد، مدیریت shared memory مستقلاً توسط دو process صورت می‌پذیرد و سیستم‌عامل در مدیریت آن نقشی ندارد.

ت) در سیستم‌عاملی که طراحی آن از نوع layered بوده و از لایه‌های صفر تا ۹ تشکیل شده است، اگر ایرادی در شیوه خدمت‌رسانی لایه ۵ وجود داشته باشد، برای ریشه‌یابی اشکال پیش آمده صرفاً باید لایه‌های صفر تا ۵ بررسی گردد. (صفر درونی‌ترین لایه و ۹ بیرونی‌ترین لایه است.)

ث) در روش طراحی سیستم‌عامل به شکل Microkernel تا جای ممکن kernel سیستم‌عامل بصورت Modular طراحی می‌گردد و ماجول‌های طراحی شده همگی در kernel mode اجرا می‌شوند.

ج) اگر یک process در حالت running باشد، در اینصورت، با رخداد timer interrupt، آن process در حالت waiting قرار می‌گیرد.

چ) وظیفه مدیریت main memory و cache توسط سیستم‌عامل و وظیفه مدیریت register ها توسط کامپایلر صورت می‌گیرد.

ح) ارتباط بین ماجول‌های یک سیستم‌عامل طراحی شده به شکل Microkernel از طریق shared memory صورت می‌پذیرد.

خ) در مدل NUMA یا همان Non-Uniform Memory Access، هر پردازنده یک حافظه محلی برای خود دارد و فضای آدرس‌دهی (address space) هر یک از این حافظه‌ها از یکدیگر جداست.

د) به ذخیره‌سازی بخش‌هایی از داده‌ها در یک ذخیره‌ساز سریع‌تر به منظور افزایش کارایی و دسترسی سریع‌تر به آن در اصطلاح buffering و به overlapping بین خروجی یک job با ورودی job های دیگر در اصلاح spooling گفته می‌شود.

ذ) اگر ساختارداده‌ای در kernel برای ذخیره‌سازی n کلید از نوع یک درخت جستجوی دودویی بالانس شده باشد، آنگاه، مدت زمان جستجوی یک کلید در آن ساختارداده در بدترین حالت از درجه $\mathcal{O}(n)$ است.

۳- با استدلال بیان کنید با اجرای شبه‌کد نوشته شده روبه‌رو حداکثر چه تعداد process به تعداد process های حاضر در حافظه اصلی اضافه خواهد داشت؟

```
#include <stdio.h>
#include <unistd.h>

int main()
{
    /* fork 1 */
    fork();

    /* fork 2 */
    fork();

    ....

    /* fork n */
    fork();

    return 0;
}
```

۴- برنامه روبه‌رو که به زبان C نوشته شده است را در نظر بگیرید و به پرسش‌های زیر پاسخ دهید:

الف) با اجرای برنامه، مقدارهای چاپ شده در خروجی به ترتیب چه خواهند بود؟

ب) با حذف wait(NULL) از خط A و انتقال آن به خط B و اجرای برنامه تغییر یافته، چه تغییری در ترتیب مقدارهای چاپ شده (نسبت به بخش الف) در خروجی رخ خواهد داد؟

پ) اگر علاوه بر wait(NULL) در خط A، در خط B نیز یک بار دیگر wait(NULL) را فراخوانی کنیم و برنامه تغییریافته را اجرا نماییم، چه تغییری (نسبت به بخش الف) در خروجی رخ خواهد داد؟

ت) فرض کنید سیستم‌عامل نتواند دستور fork را با موفقیت اجرا نماید. در اینصورت، قطعه کدی را پیشنهاد دهید که با قرار دادن آن در جای مناسبی از برنامه روبه‌رو، اجراکننده برنامه را از این امر (یعنی عدم موفقیت سیستم‌عامل) مطلع سازد.

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>

#define SIZE 3
int nums[SIZE] = {2,3,5};

int main()
{
    int i;
    pid_t pid;
    pid = fork();

    if (pid == 0) {
        for (i = 0; i < SIZE; i++) {
            nums[i] = nums[i] * nums[i];
            printf("%d\n",nums[i]);
        }
    }
    else if (pid > 0) {
        wait(NULL);     /* LINE A */
        for (i = 0; i < SIZE; i++) {
            printf("%d\n",(-1)*nums[i]);
        }
        /* LINE B */
    }

    return 0;
}
```