آزمون میانی درس سیستم عامل

مجموع نمره‌ها: ۱۰۰ زمان آزمون: ۸۰ دقیقه

---

۱ (۱۰) وقفه‌ها (Interrupt) را می‌توان به سه دسته‌ی کلی تقسیم کرد که هر یک نقش مهمی در یک سیستم عامل بازی می‌کند. این سه نقش در سیستم عامل را توضیح دهید.

۲ (۸) دلایل معرفی چند برنامگی (Multiprogramming) و چند وظیفگی (Multitasking) را در سیستم عامل شرح دهید و تفاوت آنها را نیز مشخص کنید.

۳ (۵) فضای حافظه‌ی (Address space) ریسمان‌های یک پردازه (Process) مشترک هستند. دو راه برای تخصیص حافظه‌ی اختصاصی به هر ریسمان (که بتواند از آن به صورت خصوصی استفاده کند) را بیان کنید.

۴ (۱۰) مزیت‌های سیستم عامل‌هایی که از ماژول‌های هسته (Loadable Kernel Modules) استفاده می‌کنند نسبت به سیستم عامل‌های یکپارچه (Monolithic) چیست؟ آنها را از دید مقاومت در برابر خطاهای نرم‌افزاری در راه‌اندازها و سربار (Overhead) داخلی سیستم عامل نیز مقایسه نمایید.

۵ (۱۰) در تعویض متن (Context Switch)، سیستم عامل چه عملیاتی را انجام می‌دهد؟ همچنین، پردازه‌ی جدید از بین کدام پردازه‌ها انتخاب می‌شود؟

۶ (۱۰) در پیاده‌سازی قفل‌ها، با دلیل بیان کنید در چه شرایطی «Busy waiting» و در چه شرایطی «Blocking» راه حل مناسب‌تری هستند.

۷ (۱۵) دو ریسمان در نظر بگیرید: تابع «q()» در ریسمان دوم تنها وقتی باید فراخوانی شود که ریسمان اول تابع «p()» را فراخوانی کرده باشد. یک از دوستانتان برای همگام‌سازی این دو ریسمان، راه حل زیر را پیشنهاد داده است. آیا امکان بروز وضعیت رقابتی (و در نتیجه فراخوانی تابع «q()» قبل از «p()») وجود دارد؟ اگر بله، توضیح دهید در چه صورتی رخ می‌دهد و راه حل جایگزین پیشنهاد دهید.

متغیرهای مشترک در دو ریسمان

```
semaphore sem = 0;
int p_called = 0;
```

کد اجرا شده توسط ریسمان اول

```
p_called = 1;
p();
signal(sem);
```

کد اجرا شده توسط ریسمان دوم

```
if (p_called == 0)
      wait(sem);
q();
```

۸ (۱۲) در شبه کد زیر، تابع «fork()» فراخوانی می‌شود که پردازه‌ی فراخوانی کننده را تکثیر می‌کند و در پردازه‌ی پدر مقدار شناسه‌ی فرزند (عددی بیشتر از صفر) و در پردازه‌ی فرزند، عدد صفر را بر می‌گرداند. درخت پردازه‌های حاصل از اجرای این برنامه را بکشید (یک رأس به ازای هر پردازه و یک یال از پردازه‌ی پدر به هر فرزندش) و همچنین عدد چاپ شده توسط هر پردازه را در کنار رأس اختصاص یافته به آن پردازه نشان دهید.

```
int val = 0;
if (fork() > 0)
      val = val + 1;
if (fork() == 0) {
      val = val + 1;
      fork();
}
printf("%d\n", val);
```

۹ (۲۰) فرض کنید زمانبندی توسط الگوریتم صف چند رده‌ای (Multilevel Queue) انجام می‌شود که در آن دو صف وجود دارند: در صف اول از الگوریتم «Round Robin» با تکه‌ی زمانی (Time slice) ۱۰ میلی ثانیه و در صف دوم از الگوریتم «First-Come First-Served» (زمانبندی بر اساس زمان ورود) استفاده می‌شود و صف اول نسبت به صف دوم اولویت دارد (تنها پردازه‌های صف دوم هنگامی اجرا می‌شوند که صف اول خالی باشد).

پردازه	زمان ورود	CPU Burst	شماره صف
A	۰	۱۵	۱
B	۵	۱۵	۱
C	۱۰	۱۰	۲
D	۳۵	۱۵	۱
E	۳۵	۱۵	۲

نمودار Gantt (نمایش پردازه‌ی در حال اجرا در طول زمان) را بکشید و برای پردازه‌های A و C زمان انتظار (Waiting)، زمان برگشت (Turnaround) و زمان پاسخ (Response) را محاسبه نمایید.