آزمون میانی درس سیستم عامل

مجموع نمره‌ها: ۵ + ۱۰۰ زمان آزمون: ۸۰ دقیقه

---

۱ (۵) یکی از دوستانتان ادعا می‌کند که هیچ‌گاه در برنامه‌هایش از فراخوانی سیستمی استفاده نمی‌کند و برای اثبات این ادعا کد یکی از برنامه‌هایش که با زبان جاوا نوشته شده است را نشان می‌دهد که فایلی را از فایل سیستم می‌خواند (که واقعا در آن فراخوانی سیستمی دیده نمی‌شود). آیا این برنامه فرخوانی سیستمی انجام نمی‌دهد؟ توضیح دهید.

۲ (۱۰) یکی از دوستانتان برای کاهش بار (Load) روی یک سرور پیشنهاد می‌کند به جای استفاده از مدل سرویس گیرنده-سرویس دهنده (Client-Server)، از معماری نظیر به نظیر (Peer-to-Peer) استفاده شود. آیا این پیشنهاد خوبی است؟ دلیل بیاورید.

۳ (۱۰) آیا در یونیکس پس از مرگ فرزند یک پردازه (Process)، سیستم عامل اطلاعات آن را کاملا دور می‌اندازد؟ اگر خیر، چه اطلاعاتی را نگه می‌دارد، تا چه هنگام نگه می‌دارد و به چه دلیلی نگه می‌دارد؟

۴ (۱۰) در یک سرور برای پاسخ به هر درخواست، یک بند (ریسمان یا Thread) ایجاد می‌شود. در صورتی که نرخ درخواست‌ها زیاد باشد چه خطری وجود دارد (فرض کنید درخواست‌ها به ترتیب بررسی می‌شوند، بین درخواست‌ها وابستگی وجود ندارد و وضعیت رقابتی رخ نمی‌دهد)؟ برای حل آن چه راهی را پیشنهاد می‌کنید؟ چرا فکر می‌کنید راه شما مؤثر است؟

۵ (۲۰) فرض کنید result یک متغیر سراسری با مقدار اولیه‌ی صفر باشد. تابع finished() بزرگ‌ترین مقدار فرستاده شده به آن را به متغیر result نسبت می‌دهد. اگر این تابع توسط چند بند به صورت همروند فراخوانی شود، با مثالی نشان دهید وضعیت رقابتی (Race condition) چگونه رخ می‌دهد. این مشکل را با استفاده از قفل یا سمافور برطرف کنید.

```
void finished(int value)
{
      if (value > result)
            result = value;
}
```

۶ (۳۰) فرض کنید سیستم عامل از الگوریتم زمانبندی چند صفه (Multi-level Queue) با سه صف برای زمانبندی پردازنده استفاده می‌کند. در صف اول از الگوریتم Round Robin با برش زمانی ۱۰ میلی ثانیه، در صف دوم از الگوریتم Shortest Job First (در حالت Preemptive) و در صف سوم از الگوریتم First-Come First-Served استفاده می‌شود. بین صف‌ها نیز از زمانبندی اولویت استفاده می‌شود (تنها وقتی پردازه‌های یک صف اجرا می‌شوند که صف‌های قبلی خالی باشند). با در نظر گرفتن اطلاعات جدول زیر، نمودار Gantt را برای زمانبندی بکشید و زمان پاسخ (Response) و انتظار (Waiting) را برای پردازه‌ها محاسبه نمایید.

پردازه	شماره‌ی صف	زمان ورود	زمان پردازش
A	۲	۰	۲۵
B	۳	۵	۳
C	۲	۱۰	۱۰
D	۱	۱۵	۱۵
E	۱	۲۰	۵

۷ (۲۰) دو بند را در نظر بگیرید. این دو بند تابع ()synch را فراخوانی می‌کنند (ترتیب فراخوانی شدن این تابع توسط بندها مشخص نیست). اولین بندی که این تابع را فراخوانی می‌کند باید منتظر شود تا وقتی که بند دیگر نیز تابع را صدا بزند. سپس اجرای هر دو بند ادامه پیدا می‌کند (دو بند از تابع خارج می‌شوند). این تابع را به کمک سمافور یا مانیتور پیاده‌سازی کنید.