# پروژه نهایی درس سیستم عامل

## سیستم مدیریت رستوران

نیمسال اول ۱۴۰۱-۱۴۰۲

در این پروژه، قصد داریم تا سیستم مدیریت یک رستوران را بنویسیم. این رستوران، رستوران شلوغی است و گاهی اوقات ممکن است که میزها پر بشوند. در این صورت، ما احتیاج داریم تا یکی از مشتری‌ها را بیرون بکنیم(!) تا مشتری جدید وارد رستوران بشود. مدیریت رستوران از شما خواسته تا سیستمی برای مدیریت این که بفهمیم کدام مشتری را از رستوران بیرون بکنیم بنویسید.

شما بین چندین الگوریتم برای پیاده‌سازی این مسئله مردد هستید. همانطور که در درس سیستم عامل یاد گرفته بودید، سه الگوریتم مهم را در ذهن خود دارید:

FIFO, LRU, Second-chance

در نهایت شما در تست این پروژه می‌خواهید بفهمید کدام الگوریتم بیشتر به درد رستوران می‌خورد. بنابراین در آخر تست، باید یک گزارش از تعداد مشتری‌هایی که با هر الگوریتم از رستوران بیرون انداخته می‌شوند را به مدیر رستوران بگویید.

در ابتدا که هر مشتری وارد رستوران می‌شود، یک شماره مخصوص همان مشتری به او داده می‌شود. سر یک میز نشانده می‌شود. زمانی که یک مشتری دیگر وارد بشود، با توجه به الگوریتم، یکی از مشتری‌ها بیرون انداخته می‌شود و مشتری جدید سر میز می‌نشیند.

اگر مشتری قبلی که بیرون انداخته شده دوباره سفارش داشته باشد، باز می‌تواند وارد رستوران بشود و در این مورد منعی وجود ندارد. همچنین مشتری‌ها می‌توانند داخل رستوران باشند و چند بار دیگر سفارش بدهند.

اسامی مشتری‌هایی که در حال حاضر مایل به سفارش هستند، هر ۱ میلی ثانیه تا ۲ ثانیه یکبار توسط مدیر رستوران به اطلاع ما می‌رسد. مدیر رستوران، اطلاع‌رسانی‌هایش را از طریق سرور شخصی‌ای که خریده و نصب کرده، به اطلاع ما می‌رساند. بدین ترتیب شما باید پیام‌های مدیر رستوران را از سرور او دریافت کنید و تعداد مشتری‌هایی که در نهایت از رستوران بیرون انداخته می‌شوند را در آخر روز کاری به برگردانید. زمانی که مدیر رستوران پیام ”0“ را فرستاد، به معنای پایان روز کاری است.

**طریقه پیاده‌سازی پروژه**

پیام‌های مدیر رستوران، توسط فایل javaای که به پروژه شما پیوست شده است، به دست شما می‌رسد. ارتباط بین برنامه مدیر رستوران و برنامه شما، به صورت سوکتی است.

فایل مورد نظر را در سیستم عامل خود با javac و java (یا هر روش دیگری که بلد هستید)، کامپایل کرده و اجرا کنید.

به طور خلاصه، ارتباط سوکت بین دو برنامه اینگونه است که برنامه سرور (مدیر رستوران) منتظر می‌ماند تا کلاینت (برنامه شما) به آن متصل شود. زمانی که اتصال به درستی برقرار شد، سرور پیغام

“Client connected”

را در کنسول خود چاپ می‌کند. با مشاهده این پیام در کنسول سرور می‌توانید بفهمید که اتصال اولیه را به درستی برقرار کرده‌اید. بعد از آن، ابتدا سرور مقدار n یا همان تعداد میزهای رستوران را می‌فرستد. سپس طی زمان‌های رندوم بین ۱ میلی ثانیه تا ۲ ثانیه، برای شما شماره مشتری‌ای که مایل به سفارش است را ارسال می‌کند.

شما **درست بعد از دریافت این پیام‌ها**، باید اقدام به جاگذاری مشتری سر میزش بکنید. به عبارت دیگر، باید هر سه الگوریتم را بعد از دریافت هر پیغام از سرور اجرا بکنید و لیست افرادی که در حال حاضر سر هر میز هستند برای هر الگوریتم در کنسول برنامه خودتان چاپ بکنید.

در آخر، سرور برای شما پیغام “0” (بدون “”) را ارسال می‌کند و شما درست بعد از این که این پیغام را دریافت کردید، page fault برای هر سه الگوریتم را با فرمت زیر در کنسول خودتان چاپ می‌کنید:

LRU:<page_fault>,FIFO:<page_fault>,Second-chance:<page_fault>

**نکات مهم:**

- برای اجرای بهتر برنامه، بهتر است که از Multi Threading استفاده بکنید. به این شکل که خواندن پیغام‌های سرور در یک thread و اجرای الگوریتم‌ها در یک thread دیگر باشد.
- برای پیاده‌سازی کدتان، از هر زبانی می‌توانید استفاده بکنید.
- پورتی که سوکت در برنامه شما به آن متصل می‌شود، در داخل برنامه سرور به صورت

static final int PORT = 8080;

است. مقدار پیش‌فرض آن پورت ۸۰۸۰ است. شما می‌توانید اگر دوست داشتید مقدار پورت دیفالت را تغییر بدهید.

- استفاده از Thread واجب نیست. ولی استفاده از آن نمره امتیازی دارد.
- به هیچ‌وجه کد سرور را مگر برای تغییر پورت آن، تغییر ندهید. وگرنه در هنگام تحویل دچار مشکل خواهید شد.
- در زبان c، اگر برای استفاده از thread دچار مشکل شدید، می‌توانید از processها هم استفاده کنید.
- کانکشن از نوع TCP است.

**آن چه باید آپلود کنید:**

یک فایل زیپ شامل کدهای زده شده و یک گزارش درباره چگونگی هندل کردن اتصال با سوکت، توضیحاتی درباره پیاده‌سازی الگوریتم‌ها، در صورت استفاده از Thread، توضیحات مختصری در مورد آن، نوشته شود. همچنین چند خروجی نمونه به همراه ورودی‌های آن را هم داخل گزارش خود بگذارید.