دانشگاه علم و صنعت ایران
دانشکده کامپیوتر

سیستم‌های عامل

ریسمان و ناحیه بحرانی

سوال اول

در برنامه های دارای چند ریسمان[۱] شرایطی پیش می‌آید که در آن نیاز است تردها منتظر بمانند تا کار دیگران نیز به مرحله‌ای مشخص برسد. به عنوان مثال یک برنامه را در نظر بگیرید که می‌خواهد ماتریسی را به توان پنج برساند. در این برنامه هر ریسمان مسئول محاسبه یک درایه است. برای سادگی فرض کنید یک ماتریس 2×2 را می‌خواهیم به توان پنج برسانیم. هنگامی که درایه a_{11} برای ماتریس A^2 توسط یکی از ریسمان‌ها محاسبه می‌شود، قبل از ادامه دادن برای محاسبه توان A^3 باید صبر کند تا دیگر ریسمان‌ها نیز مقادیر دیگر درایه‌ها را محاسبه کنند. توجه کنید که عملیات join در اینجا مطلوب نیست چرا که join صبر می‌کند تا کار یک ریسمان به صورت کامل تمام شود. در این مواقع از یک عامل هماهنگ‌سازی[۲] استفاده می‌شود با نام مانع[۳].

مانع به این صورت عمل می‌کند که بعد از ایجاد و در هنگام مقدار دهی اولیه، مشخص می‌شود که چند تا ریسمان باید به این مانع برسند تا اجازه عبور برای همه داده شود. به همین دلیل ریسمان‌هایی که زودتر به مانع می‌رسند متوقف[۴] می‌شوند تا زمانی که تعداد ریسمان‌های منتظر به شماره مورد نظر برسد. سپس تمام ریسمان‌ها اجازه عبور از مانع را خواهند داشت. کد زیر سعی می‌کند نحوه استفاده از یک مانع را در یک زبان فرضی نشان دهد.

---

[۱] Multi-threaded
[۲] Syncronisation
[۳] Barrier
[۴] Block

```
// define a barrier
barrier_t b

function main:
    init_barrier(&b, 10) // wait until 10 threads reach the block;
    for i to 10 do
        create_thread(worker_func)
    end
    ...

function worker_func():
    process()
    ...
    wait(b) // wait until all 10 threads reach this point
    // continue processing
    ...
```

از شما خواسته شده است تا فقط با استفاده از mutex این عامل هماهنگ‌سازی را پیاده کنید. استفاده از Semaphore و یا Conditional Variables برای این منظور مجاز نیست. در صورت امکان کد پیاده سازی را بنویسید و آن را تست کنید. درصورتی که امکان پیاده سازی این *مانع* با شرایط گفته شده وجود ندارد دلیل خود را به صورت مفصل و به همراه ذکر مثال شرح دهید.

سوال دوم

با استفاده mutex یک سمافور[۵] پیاده سازی کنید. سمافور دارای یک مقدار صحیح نا منفی است و مقدار اولیه آن در هنگام آماده‌سازی آن مشخص می‌شود. سپس هر بار که تابع wait برای آن فراخوانی می‌شود مقدار آن یک واحد کم می‌شود و هر بار که تابع post برای آن فراخوانی می‌شود مقدار آن یک واحد افزایش پیدا می‌کند. اگر در هنگام فراخوانی تابع wait مقدار سمافور صفر باشد آنگاه اجرای برنامه متوقف[۶] می‌شود تا زمانی که ریسمان دیگری post را برای سمافور فراخوانی کند. در هنگام فراخوانی post ریسمان از حالت توقف خارج می‌شود دو باره سعی می‌کند که مقدار سمافور را یک واحد کم کند. اگر مقدار (پیش از کاهش) صفر نباشد، اجازه ادامه پیدا می‌کند در غیر اینصورت متوقف می‌شود. کد زیر استفاده از سمافور را در یک زبان فرضی نشان می‌دهد.

```
// define a semaphore
sem_t s

function main:
    init_sem(&s, 1) // set semaphore value to 1
    for i to 10 do
        create_thread(worker_func)
    end
    ...

function worker_func():
    process()
    ...
    wait(s) // wait until all 10 threads reach this point
    critical_region()
    ...
```

از شما خواسته شده است تا فقط با استفاده از mutex یک سمافور پیاده کنید. استفاده از Semaphore و یا Conditional Variables برای این منظور مجاز نیست. در صورت امکان کد پیاده سازی را بنویسید و آن را تست کنید. اگر که امکان پیاده سازی سمافور با شرایط گفته شده وجود ندارد دلیل خود را به صورت مفصل و به همراه ذکر مثال شرح دهید.

---

[۵] Semaphore
[۶] Block