همگام‌سازی بندها

تابع f1() توسط بند T1، تابع f2() توسط بند T2، تابع f3() توسط بند T3 و تابع f4() توسط بند T4 فراخوانی می‌شوند. با استفاده از دو سمافور مشخص کنید هر بند چه کدی اجرا نماید تا دو تابع f2() و f3() پس از تابع f1() فراخوانی شوند و تابع f4() پس از فراخوانی f2() و f3() فراخوانی شود.

جواب

دو سمافور s1 و s2 با مقدار اولیه‌ی صفر را در نظر بگیرید. کد اجرا شده توسط هر بند در ادامه نمایش داده می‌شود.

کد T1	کد T2	کد T3	کد T4

`f1();`
`signal(s1);`
`signal(s1);`
	`wait(s1);`
	`f2();`
	`signal(s2);`
		`wait(s1);`
		`f3();`
		`signal(s2);`
			`wait(s2);`
			`wait(s2);`
			`f4();`

بندها و پردازه‌ها

توضیح دهید در شبه کد زیر، چه خروجی‌هایی ممکن است چاپ شوند. در این شبه کد، تابع ()fork یک کپی از پردازه‌ی فراخوانی کننده ایجاد می‌کند و تابع ()thread_create یک بند (Thread) می‌سازد که اجرای خود را با فراخوانی تابعی که به عنوان پارامتر داده می‌شود شروع می‌کند و با اتمام فراخوانی این تابع، اجرایش خاتمه می‌یابد. اجرای پردازه‌ی اصلی با فراخوانی تابع ()main آغاز می‌گردد.

```
int gvar = 0;

void f1()
{
     gvar = 1;
     print(gvar);
}

void f2()
{
     gvar = 2;
     print(gvar);
}

int main(void)
{
     if (fork()) {
          thread_create(f1);
          gvar = 3;
     } else {
          gvar = 4;
          thread_create(f2);
     }
     print(gvar);
     return 0;
}
```

ترتیب در مانیتور

مانیتور Mon مطابق شبه کد زیر تعریف شده است. فرض کنید mon یک نمونه از این مانیتور باشد. ابتدا تابع mon.f() توسط بند اول فراخوانی می‌شود. سپس، بند دوم تابع mon.g() را فراخوانی می‌کند. چه مقداری چاپ می‌شود؟ توضیح دهید.

```
Monitor Mon {
      Condition c;
      int x = 0;

      void f() {
            x = 1;
            c.wait()
            x = 2;
      }

      void g() {
            x = 3;
            c.signal()
            print x
      }
}
```

پیاده‌سازی قفل

شبه کد زیر پیاده‌سازی یک قفل را با استفاده از عمل اتمی test_and_set() نشان می‌دهد (تابع test_and_set() کلمه‌ی اشاره شده توسط ورودی را به یک تغییر می‌دهد و مقدار قدیمی آن را بر می‌گرداند). سه متغیر تعریف شده در حافظه‌ی اشتراکی قرار دارند و در پیاده‌سازی تابع res_lock متغیر TID با شناسه‌ی بند فراخوانی کننده (عدد صحیح مثبت) جایگزین می‌شود. بررسی کنید که آیا هر یک از سه شرط راه‌حل ناحیه‌ی بحرانی در این پیاده‌سازی برقرار هستند یا خیر. اگر خیر، آیا می‌توانید به شکلی آن را تغییر دهید تا هر سه شرط برقرار شوند؟

```
int mutex = 0;
int turn = -1;
int waiting = -1;

void res_lock(void)
{
     waiting = TID;
     while (turn != TID && test_and_set(&mutex) == 1)
          ;
     waiting = -1;
     turn = -1;
}

void res_unlock(void)
{
     if (waiting != -1)
          turn = waiting;
     else
          mutex = 0;
}
```