جلسه‌ی هشتم — کتابخانه‌ی PThreads

هر پردازه در سیستم عامل می‌تواند از تعدادی بند (Thread یا ریسمان) تشکیل شود که به صورت همروند اجرا می‌شوند. چون همه‌ی بندهای یک پردازه در فضای آدرس آن پردازه اجرا می‌گردند، آنها می‌توانند با استفاده از حافظه‌ی مشترک، برای مثال با متغیرهای سراسری (Global) با هم ارتباط داشته باشند.

در استاندارد POSIX یک کتابخانه‌ی بند معرفی شده است که «POSIX Threads» یا «PThreads» نامیده می‌شود. این کتابخانه در بسیاری از سیستم‌های عامل رایج از جمله لینوکس پیاده‌سازی شده است. در این جلسه با این کتابخانه آشنا می‌شوید. برای استفاده از کتابخانه‌ی PThreads در لینوکس باید پارامتر lpthread- را به Linker بدهید (در برخی از محیط‌ها باید پارامتر pthread- را هم به Linker و هم به مترجم فرستاد).

ساختن بندها

۱ اجرای یک بند با فراخوانی یک تابع آغاز می‌شود و با پایان اجرای تابع، بند از بین می‌رود. چنین تابعی باید به صورت زیر تعریف شود.

```
void *func(void *dat)                    تابعی که در بند جدید فراخوانی می‌شود
{
      printf("Thread started!\n");       این دستورات در بند جدید اجرا می‌شوند
      return NULL;
}
```

۲ با تابع pthread_create() می‌توان یک بند ایجاد نمود. ورودی اول این تابع یک اشاره‌گر به یک متغیر برای ذخیره‌سازی شناسه‌ی بند جدید است و ورودی سوم این تابع، تابعی است که بند جدید باید از آن اجرای خود را شروع کند. برای اطلاع بیشتر در مورد ورودی‌های این تابع، به صفحه‌ی راهنمای آن مراجعه نمایید.

```
#include <pthread.h>

pthread_t tid;
pthread_create(&tid, NULL, func, NULL);        ایجاد یک بند از تابع «func»
```

۳ برای انتظار برای اتمام یک بند می‌توان تابع pthread_join() را فراخوانی کرد (مشابه تابع wait() برای پردازه‌ها).

```
pthread_join(tid, NULL);          خروجی بند می‌تواند در آدرس پارامتر دوم قرار گیرد
```

۴ یک بند بسازید که هر دو ثانیه پیغامی را چاپ کند (پنج بار). بند اصلی پردازه نیز هر یک ثانیه پیغامی را چاپ کند (چهار بار) و پس از آن برای پایان بند جدید منتظر باشد.

۵ قفل‌های Mutex در کتابخانه‌ی Pthreads باید با نوع pthread_mutex_t تعریف شوند.

```
pthread_mutex_t lock;
pthread_mutex_init(&lock, NULL);          مقداردهی اولیه
pthread_mutex_lock(&lock);                فقل کردن
pthread_mutex_unlock(&lock);              باز کردن
pthread_mutex_destroy(&lock);             آزاد کردن منابع (در پایان استفاده)
```

۶ سمافورها (Semaphore) در کتابخانه‌ی Pthreads با نوع sem_t تعریف می‌شوند.

```
sem_t sem;
sem_init(&sem, 0, 1);                     مقدار دهی اولیه
sem_post(&sem);                           افزایش مقدار
sem_wait(&sem);                           کاهش مقدار (انتظار در صورت نیاز)
sem_destroy(&sem);                        آزاد کردن منابع (در پایان استفاده)
```

۷ برنامه‌ی زیر را کامل کنید تا هر تابع نام خود را چاپ کند. سپس این برنامه را تغییر دهید که تابع f() و h() در یک بند جدید فراخوانی شوند.

```
int main(void)
{
      f();
      g();
      h();
      return 0;
}
```

۸ برنامه‌ی تمرین قبل را تغییر دهید تا اجرای تابع h() در بند جدید پس از اجرای تابع g() در بند اصلی شروع شود.

۹ برنامه‌ی تمرین قبل را تغییر دهید تا اجرای تابع ()g در بند اصلی پس از اجرای تابع ()f در بند جدید شروع شود.

۱۰ قطعه کد زیر را کامل کنید تا هر یک از دو تابع توسط یک بند اجرا شوند. آیا وضعیت رقابتی رخ می‌دهد؟

```
int count;
void *thread1(void *dat)
{
	while (1) {
		int count1 = count;
		sleep(1);
		printf("thread1: %d\n", count1);
		count = count1 + 1;
	}
	return NULL;
}
void *thread2(void *dat)
{
	while (1) {
		int count2 = count;
		sleep(2);
		printf("thread2: %d\n", count2);
		count = count2 + 1;
	}
	return NULL;
}
```

۱۱ با استفاده از قفل، دسترسی‌های همزمان را مدیریت کنید.

مسئله‌ی تولید کننده و مصرف کننده (اختیاری)

۱۲ در درس سیستم عامل مسئله‌ی تولید کننده و مصرف کننده معرفی شده است. با استفاده از کتابخانه‌ی PThreads می‌توان این مسئله را با در نظر گرفتن دسترسی‌های همزمان حل کرد. هنگامی که اندازه‌ی بافر یک باشد، می‌توان با استفاده از دو سمافور داده‌های تولید شده توسط تولید کننده را به مصرف کننده انتقال داد.

```
sem_t full;                    مقدار یک به معنای وجود یک عنصر در بافر است
sem_t empty;                   مقدار یک به معنای خالی بودن بافر است
sem_init(&full, 0, 0);
sem_init(&empty, 0, 1);
```

۱۳ در ادامه قسمت مربوط به تولید کننده نشان داده شده است.

```
sem_wait(&empty);              انتظار برای خالی شدن بافر
...                            اضافه کردن یک عنصر به بافر
sem_post(&full);               تغییر مقدار سمافور «full» پس از پر شدن بافر
```

۱۴ برنامه‌ای بنویسید که دنباله‌ای از اعداد را از بند اصلی به بند جدید انتقال دهد و بند جدید با دریافت هر عدد آن را چاپ کند.