

به نام خدا

آزمایش ۵

نام خانوادگی: روژینا کاشفی

نام استاد: سرکار خانم علیزاده

تاریخ: ۱۴۰۰/۹/۱۰

(۱)

اگر برنامه به صورت سریال اجرا شود خروجی مطابق زیر خواهد بود.

```
rojina@ubuntu:~/Desktop/OSLab/Lab5$ make all
rojina@ubuntu:~/Desktop/OSLab/Lab5$ time ./serial.out 5000
real    0m0.003s
user    0m0.003s
sys     0m0.000s
rojina@ubuntu:~/Desktop/OSLab/Lab5$ time ./serial.out 50000
real    0m0.017s
user    0m0.016s
sys     0m0.000s
rojina@ubuntu:~/Desktop/OSLab/Lab5$ time ./serial.out 500000
real    0m0.127s
user    0m0.127s
sys     0m0.000s
```

تعداد نمونه	5000	50000	500000
زمان اجرا (ثانیه)	۰/۰۰۴	۰/۰۱۸	۰/۲۵

(۲)

اگر از رابطه پدر فرزندی استفاده کنیم کارها به صورت موازی انجام میشوند و منتظر دیگری نیستند و خروجی مطابق زیر میشود.

```
rojina@ubuntu:~/Desktop/OSLab/Lab5$ make all
rojina@ubuntu:~/Desktop/OSLab/Lab5$ time ./concurrent.out 5000
real    0m0.003s
user    0m0.004s
sys     0m0.000s
rojina@ubuntu:~/Desktop/OSLab/Lab5$ time ./concurrent.out 50000
real    0m0.010s
user    0m0.016s
sys     0m0.000s
rojina@ubuntu:~/Desktop/OSLab/Lab5$ time ./concurrent.out 500000
real    0m0.074s
user    0m0.142s
sys     0m0.000s
```

تعداد نمونه	5000	50000	500000
زمان اجرا (ثانیه)	۰/۰۰۳	۰/۰۱۲	۰/۰۸۲

(۳)

بله؛ متغیر `hist[counter + 12]` میتواند باعث `race condition` برای اینکه این اتفاق نیفتد میتوان با استفاده از `spin lock` یا سمافور، انحصار متقابل در حین دسترسی به آرایشی `hist` ایجاد کرد. برای مثال میتوان به ازای هر خانه `hist` یک سمافور با مقدار اولیه یک ایجاد کرد و در هنگام ایجاد تغییر در آن، به شکل زیر کد را بازنویسی کرد:

```
sem_wait(&sem[counter + 12]);
```

```
hist[counter + 12]++;
```

```
sem_post(&sem[counter + 12]);
```

(۴)

500000	50000	5000	تعداد نمونه
۰/۱۶۸	۰/۰۰۶	۰/۰۰۱	زمان اجرا (ثانیه)