بررسی سرعت اجرای برنامه با چند روش پیاده سازی

مشخصات فایل ورودی :

فایل ورودی شامل 26994 بایت و تعداد کلمه های موردنظر برای جستجو 28 هستند.

- استفاده از یک نخ :

در این صورت تمام فایل ورودی به یک نخ داده می شود و عملا از thread استفاده ای نمی شود و مانند یک تابع عادی رفتار می کند.

```
Enter number of threads ...
1
Enter file name :
test2_input.txt
Enter word's file name :
test2_words.txt
File size in bytes 26994
Word's file size in bytes 191
Output file created ... result.txt
thread_start = 104457176840400
thread_end = 104457182067200
total_time = 0.0052268
```

- استفاده از 20 نخ:

در این صورت ، درابتدای کار 20 نخ ایجاد می شود و هر نخ بخش مربوط به خود را بررسی می کند.

تقسیم کردن یک کار ثابت بین تعداد زیادی نخ ، کار بسیار کمی را به هر نخ می دهد و در عوض سربار ایجاد هرنخ و terminate کردن هر نخ بسیار زیاد می شود.

همان که در شکل مشخص است ، زمان اجرای برنامه در این حالت نسبت به حالت قبل که فقط از یک thread استفاده شده بود ، بیشتر است.

```
Enter number of threads ...
20
Enter file name :
test2_input.txt
Enter word's file name :
test2_words.txt
File size in bytes 26994
Word's file size in bytes 191
Output file created ... result.txt
thread_start = 104393474183800
thread_end = 104393484440400
total_time = 0.0102566
```

- استفاده از 4 thread و mutex lock

```
Enter number of threads ...
4
Enter file name :
test2_input.txt
Enter word's file name :
test2_words.txt
File size in bytes 26994
Word's file size in bytes 191
Output file created ... result.txt
mutex_start = 104576418413200
mutex_end = 104576424613200
total_time = 0.0062
```

- استفاده از 4 thread و semaphore

```
Enter number of threads ...
4
Enter file name :
test2_input.txt
Enter word's file name :
test2_words.txt
File size in bytes 26994
Word's file size in bytes 191
Output file created ... result.txt
semaphore_start = 104694281420300
semaphore_end = 104694287759700
total_time = 0.0063394
```