دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

درس آز مهندسی نرم‌افزار

استاد سیدحسن میریان حسین‌آبادی

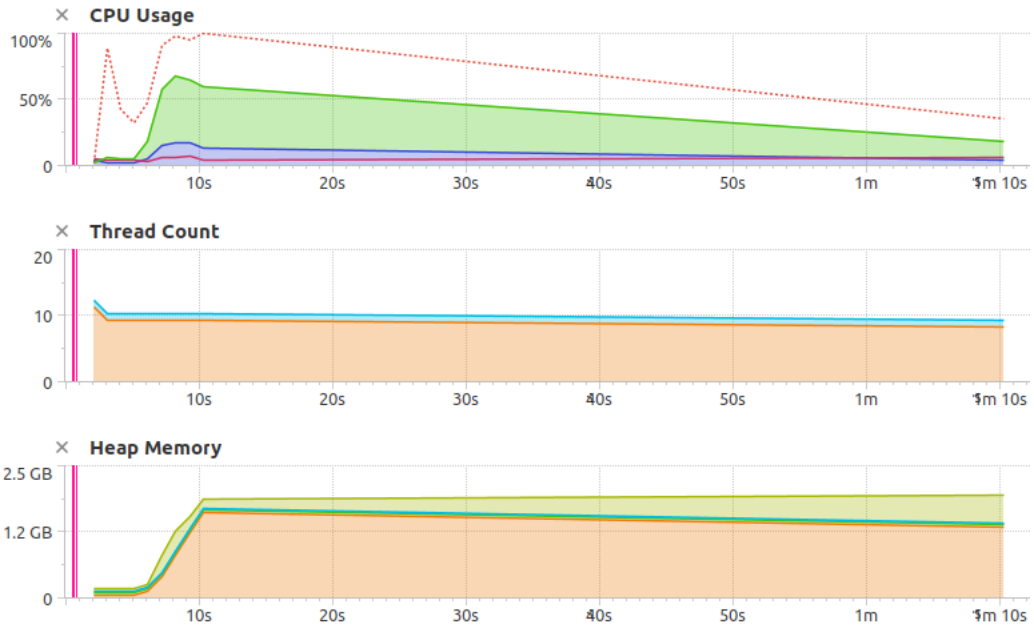
گزارش آزمایش هشتم | Profiling | حسین ذاکری‌نیا، مصطفی قدیمی

سؤال ۱. قبل از بهبود عمل‌کرد

پس از اجرای برنامه و تابع main، با استفاده از ابزار Yourkit وضعیت برنامه و منابع استفاده شده به شکل زیر است:

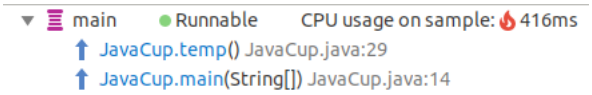
```
/usr/lib/jvm/java-1.11.0-openjdk-amd64/bin/java -agentpath:/home/mostafa/Downloads/Compressed/YourKit-JavaProfiler-2020.9-b412/
[YourKit Java Profiler 2020.9-b412] Log file: /home/mostafa/.yjp/log/JavaCup-7084.log
Press number1:
1
Press number2:
2
Press number3:
3
java.lang.OutOfMemoryError: Java heap space
Dumping heap to java_pid7084.hprof ...
Heap dump file created [2598739354 bytes in 52.949 secs]
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
    at java.base/java.util.Arrays.copyOf(Arrays.java:3689)
    at java.base/java.util.ArrayList.grow(ArrayList.java:238)
    at java.base/java.util.ArrayList.grow(ArrayList.java:243)
    at java.base/java.util.ArrayList.add(ArrayList.java:486)
    at java.base/java.util.ArrayList.add(ArrayList.java:499)
    at JavaCup.temp(JavaCup.java:38)
    at JavaCup.main(JavaCup.java:14)

Process finished with exit code 1
```

شکل ۱: برنامه بعد از اجرا قبل از بهبود عمل‌کرد



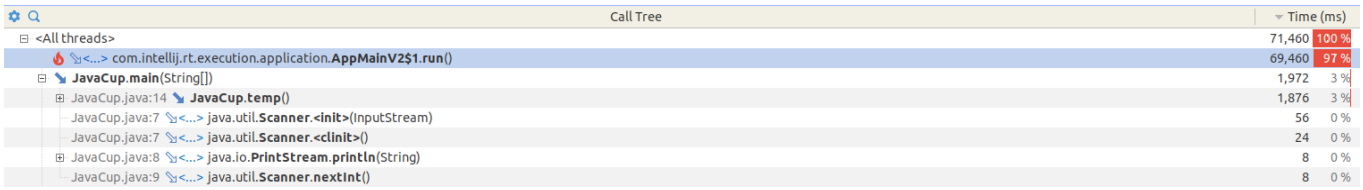
شکل ۲: نمودار منابع استفاده شده قبل از بهبود عمل‌کرد

و همان‌طور که مشاهده و برداشت می‌شود، تابع temp بیش‌ترین مصرف منابع را دارد و حافظه و پردازنده را به شدت درگیر می‌کند.



شکل ۳: توابع استفاده شده قبل از بهبود عملکرد

شکل ۴: میزان heap استفاده شده قبل از بهبود عملکرد



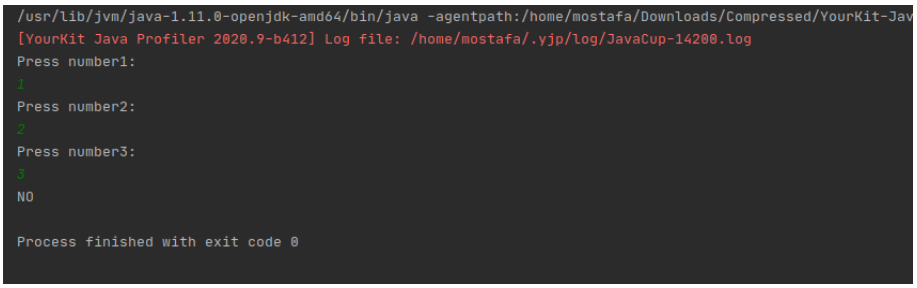
شکل ۵: زمان اجرای توابع قبل از بهبود عملکرد

سؤال ۲. بعد از بهبود عمل‌کرد

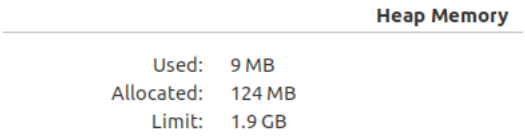
برای بهبود عمل‌کرد، باید پیاده‌سازی آن را با قطعه کد زیر جای‌گزین کنیم:

```
public static void temp() {
    int[] a = new int[200000000];
    int counter = 0;
    for (int i = 0; i < 10000; i++)
    {
        for (int j = 0; j < 20000; j++) {
        a[counter] = i + j;
        counter += 1;
        }
    }
}
```

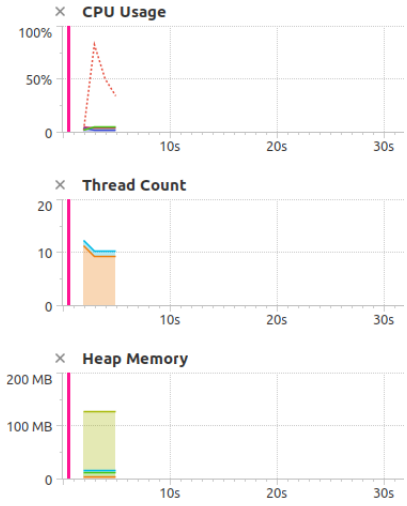
دلیل پر شدن heap، استفاده از ArrayList است. زیرا این داده‌ساختار فضای پویا می‌گیرد و هر بار که فضای آن پر می‌شود، مقدار فضای اختصاص‌داده را دو برابر می‌کند. به همین علت باعث مصرف بالای منابع می‌شود. اما در این قسمت چون تعداد خانه‌هایی که لازم است اختصاص دهیم مشخص است، می‌توانیم در ابتدا آن را مشخص کنیم و جلوی مشکل پیش‌آمده را بگیریم.
در نهایت پس از اعمال تغییرات، اگر بار دیگر با استفاده از ابزار Yourkit برنامه را اجرا کنیم، وضعیت استفاده از منابع به شکل زیر می‌شود.



شکل ۶: نتیجه اجرای برنامه بعد از بهبود عمل‌کرد



شکل ۷: میزان heap استفاده شده بعد از بهبود عمل‌کرد



شکل ۸: نمودار منابع استفاده شده بعد از بهبود عملکرد

شکل ۹: زمان اجرای توابع بعد از بهبود عملکرد