

به نام خدا



تمرین ۴  
آز نرم افزار

احمد نصرت بخش  
97110217

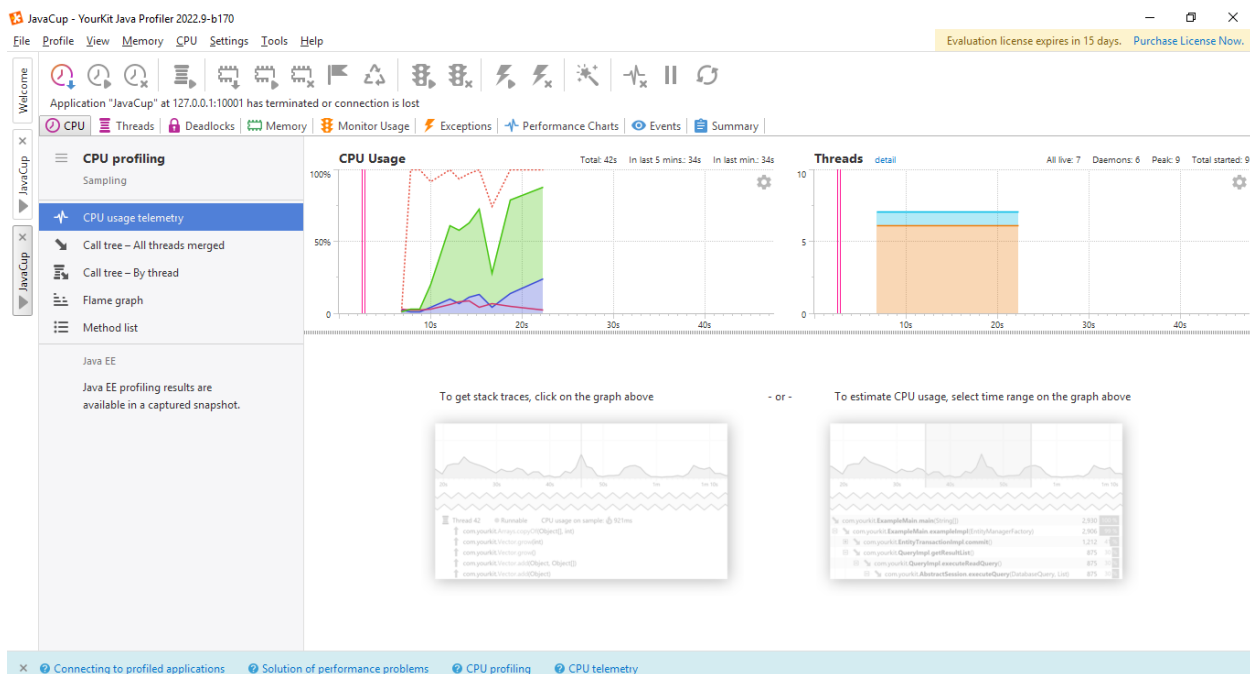
ابتدا به بررسی تصاویری از مراحل نصب برنامه **YourKit-JavaProfiler** میپردازیم. با مراجعه به وبسایت <https://www.yourkit.com/download>، آخرین نسخه مطابق با سیستم عامل (در اینجا ویندوز) را دانلود کرده و سپس آن را اجرا میکنیم. برای دریافت مجوز ۱۵ روزه ایمیل خود را وارد و **lisence key** را دریافت میکنیم. سپس کلید ارسال شده را در برنامه وارد کرده و مجوز استفاده از آن فعال میشود. در پروژه **ProfilingTest**، عملیات **Profiling** را بر روی کلاس **JavaCup** اجرا کرده و با استفاده از **YourKit** به بررسی مصرف منابع، تحلیل حافظه و **CPU** و **Thread** ها خواهیم پرداخت.

```

Run: JavaCup x
"C:\Program Files\Java\jdk-11.0.2\bin\java.exe" ...
[YourKit Java Profiler 2022.9-b170] Log file: C:\Users\ASuS\.yjp\log\JavaCup-14916.log
Press number1:
Press number2:
Press number3:
java.lang.OutOfMemoryError: Java heap space
Dumping heap to java_pid14916.hprof ...
Heap dump file created [2613936624 bytes in 44.881 secs]
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
    at java.base/java.util.Arrays.copyOfOf(Arrays.java:3689)
    at java.base/java.util.ArrayList.grow(ArrayList.java:237)
    at java.base/java.util.ArrayList.grow(ArrayList.java:242)
    at java.base/java.util.ArrayList.add(ArrayList.java:485)
    at java.base/java.util.ArrayList.add(ArrayList.java:498)
    at JavaCup.temp(JavaCup.java:30)
    at JavaCup.main(JavaCup.java:14)

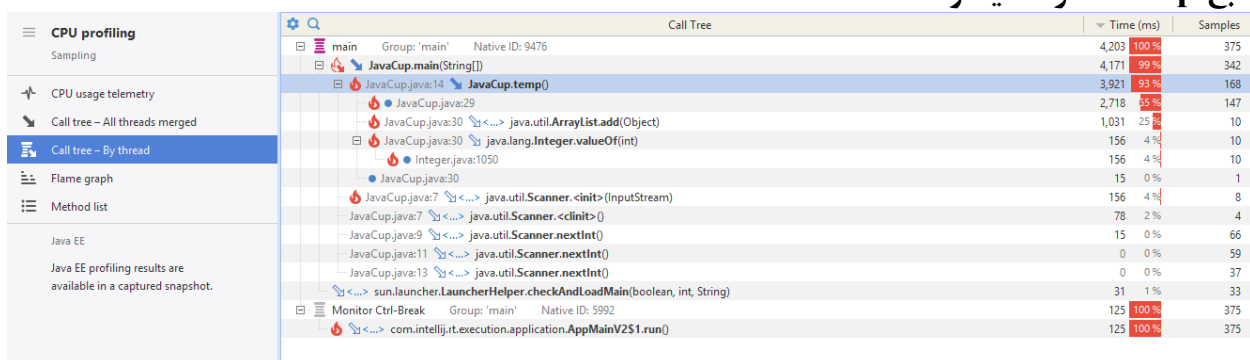
Process finished with exit code 1
  
```

خروجی برنامه بعد از اجرای عملیات profiling

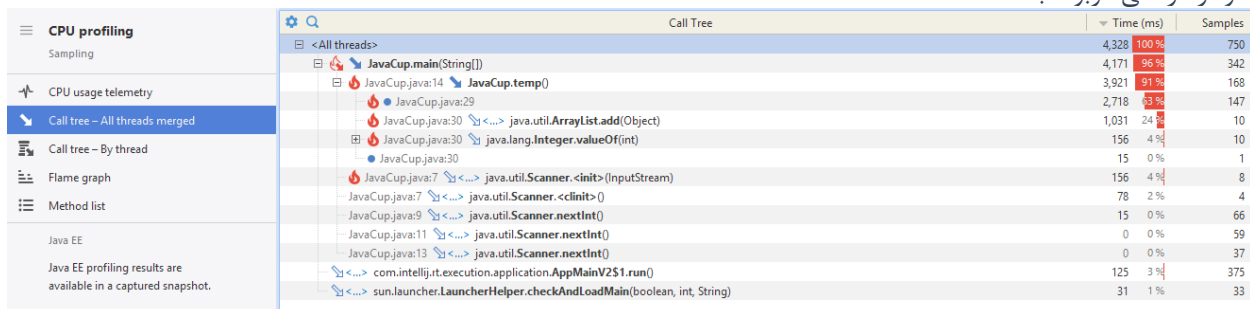


نمودار استفاده از پردازنده

طبق نمودار مربوط به thread ها و زمان اجرای توابع مشاهده میشود ۹۱ درصد زمان اجرایی توسط تابع temp مصرف میشود.



نمودار درختی مربوط به thread ها

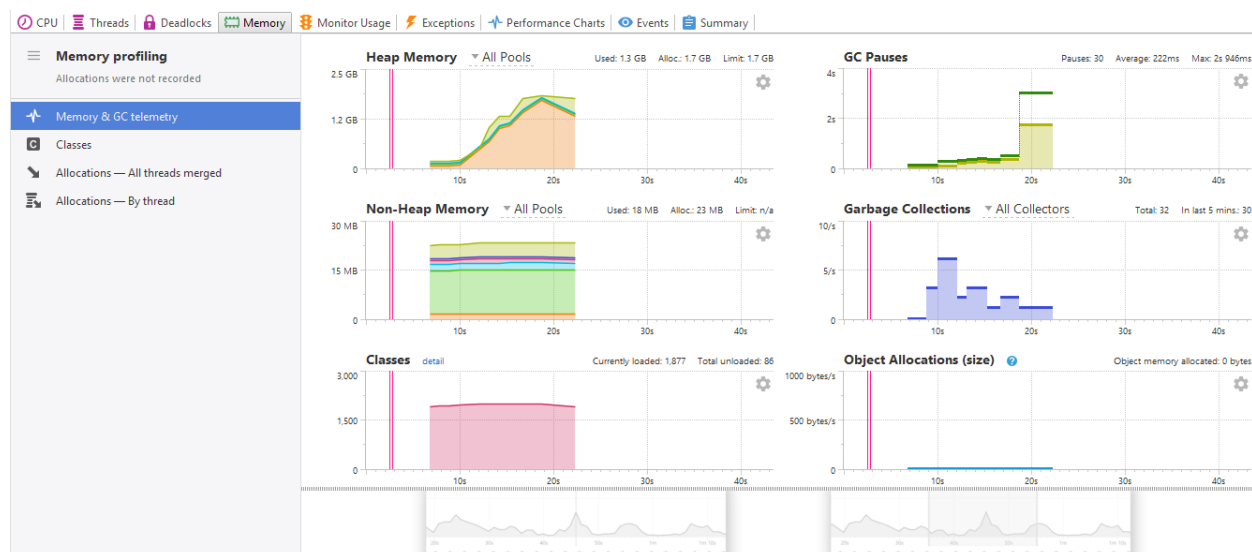


نمودار درختی thread ها و زمان اجرای توابع

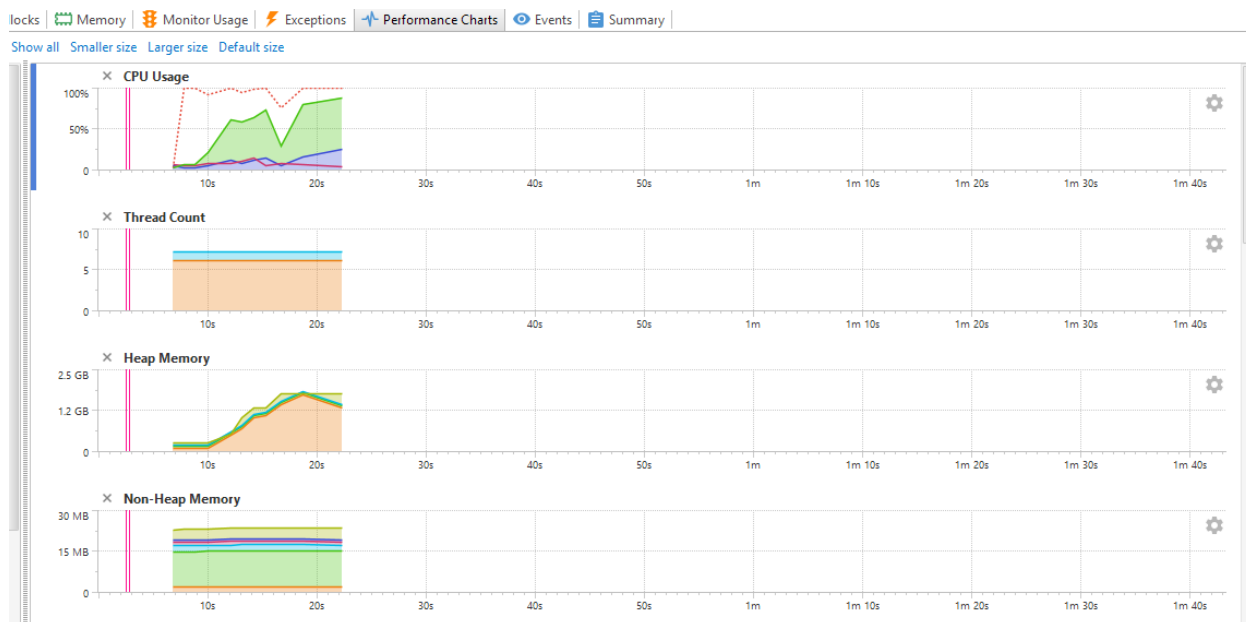
Method	Time (ms)	Own Time (ms)	Samples
JavaCup.main(String[]) JavaCup.java	4,171 96 %	0	342
JavaCup.temp() JavaCup.java	3,921 91 %	2,734	168
java.util.ArrayList.add(Object) ArrayList.java	1,031 24 %	1,031	10
java.lang.Integer.valueOf(int) Integer.java	156 4 %	156	10
java.util.Scanner.<init>(InputStream) Scanner.java	156 4 %	156	8
com.intellij.rt.execution.application.AppMainV2\$1.run() AppMainV2.java	125 3 %	125	375
java.util.Scanner.<clinit>() Scanner.java	78 2 %	78	4
sun.launcher.LauncherHelper.checkAndLoadMain(boolean, int, String) LauncherHelper.java	31 1 %	31	33
java.util.Scanner.nextInt() Scanner.java	15 0 %	15	162

نمودار زمان مصرفی و درصد زمانی مربوط به توابع

همچنین طبق نمودارهای مربوط به حافظه مصرفی داریم:



نمودار حافظه مصرفی



نمودار کارایی پردازنده و حافظه و thread ها

طبق بررسی نمودارهای بالا و حجم بالای مصرف منابع و زمان اجرایی توسط تابع temp، به بررسی دقیقتر آن میپردازیم.

```

JavaCup.java
20 {
21     System.out.println("YES");
22 }
23 else { System.out.println("NO"); }
24 }
25 public static void temp() {
26     ArrayList a = new ArrayList();
27     for (int i = 0; i < 10000; i++)
28     {
29         for (int j = 0; j < 20000; j++) {
30             a.add(i + j);
31         }
32     }
33 }
34 }
35

```

تابع temp

در این تابع، از **ArrayList** جاوا استفاده شده و  $10000 * 20000$  مرتبه به آن عضو اضافه میکند. این داده ساختار به صورت فضای پویا پیاده‌سازی شده و هر بار که فضای آن تکمیل شود، سایز خود را دوبرابر میکند. به همین علت منابع و زمان اجرایی بالایی را مصرف کرده و **heap** پر میشود. (خطای پس از اجرای برنامه نیز به همین علت بود).