به نام خدا

آزمایش پنجم آزمایشگاه مهندسی نرم افزار

آشنایی با نحوه پروفایل برنامه (Profiling)

علی‌رضا دیزجی ۹۶۱۰۵۷۴۵

حمیدرضا هدایتی ۹۶۱۰۹۹۳۹

دانشگاه صنعتی شریف

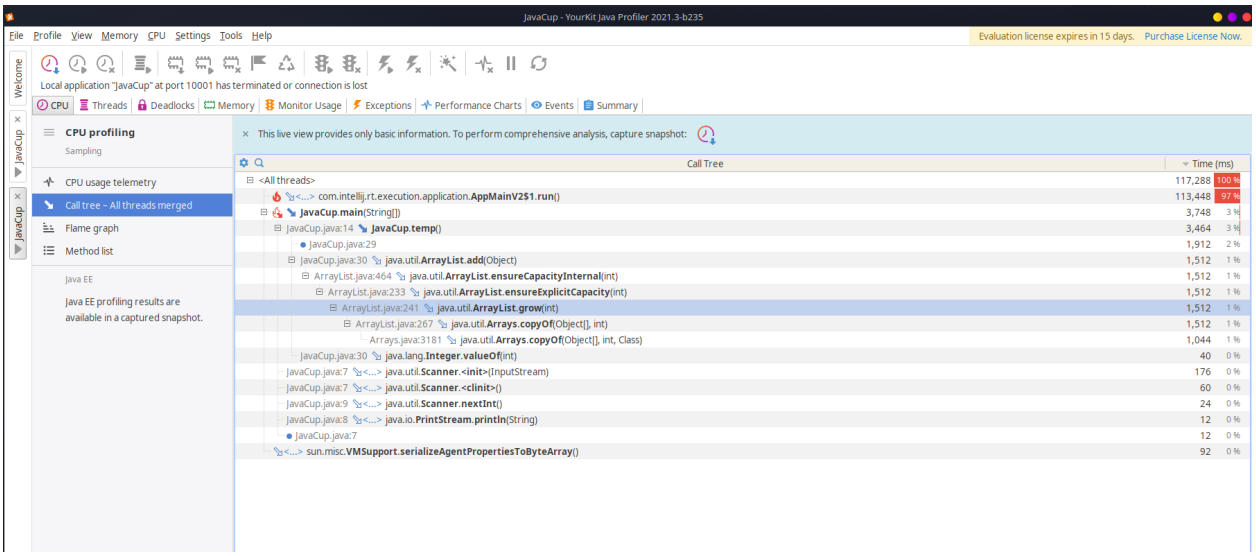
پاییز ۱۴۰۰

ابتدا همانطور که در صورت آزمایش اشاره شده بود نسخه trial برنامه yourkit بر روی ubuntu دانلود شده و نصب شد و سپس باز نگه داشته شد و برنامه intellij باز شد، yourkit در هنگام راه‌اندازی plugin خود را بر روی intellij نصب نمود.

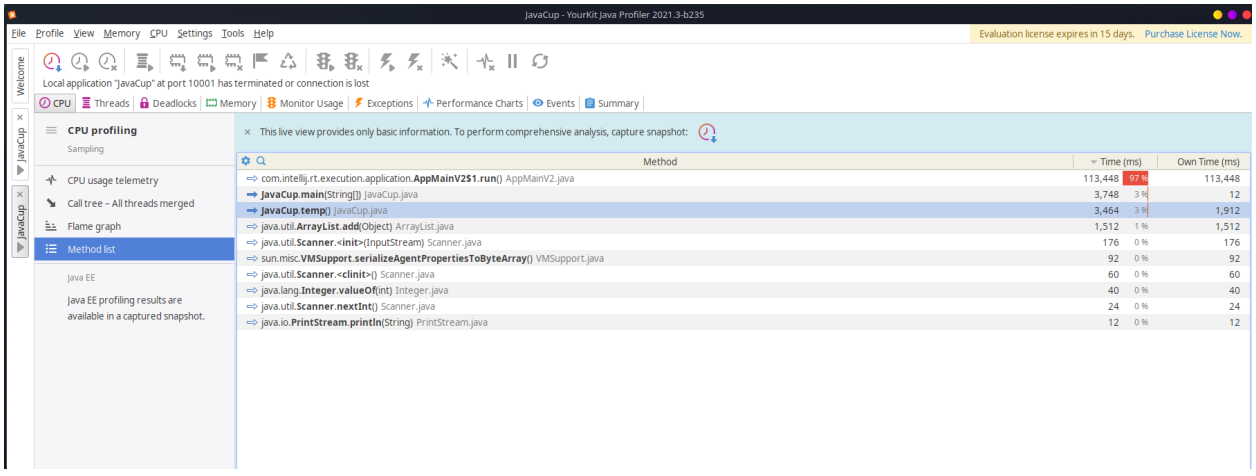
۱- انجام عمل پروفایلینگ بر روی پروژه داده شده

با استفاده از "profile "java cup در زیر شاخه run برنامه موجود را profile کردیم که به نتیجه زیر رسیدیم.

Call tree:



Method list:



که از این دو نتیجه می‌گیریم که بیشترین زمان مصرف شده بر روی تابع temp برنامه است که با ریز شدن در call tree به تابع grow در arraylist بر می‌خوریم.

۲- حل کردن مشکل با استفاده از profiling:

- استفاده از initial size در array list:

از آنجا که بیشتر برنامه pend قسمت grow تابع arraylist.add می‌شد با استفاده از یک initial size مشخص شده از اجرای تابع grow جلوگیری کردیم که در نتیجه تکه کد به عکس زیر تبدیل شد:

```
        int k = scanner.nextInt();
        temp();
        eval(i, j, k);
    }
    public static void eval(int i, int j, int k)
    {
        if (i * i + j * j == k * k || i * i == j * j + k * k || j * j == i * i + k * k
        {
            System.out.println("YES");
        }
        else { System.out.println("NO"); }
    }
    public static void temp() {
        ArrayList a = new ArrayList( i 10000*20000+2);
        for (int i = 0; i < 10000; i++)
        {
            for (int j = 0; j < 20000; j++) {
                a.add(i + j);
            }
        }
    }
}
```

با اینکه این مورد باعث شد در اندازه‌های کوچک‌تر برنامه بسیار سریع‌تر اجرا شود ولی باز هم به اندازه کافی خوب نبود و در تابع initial برنامه گیر می‌کرد.

- استفاده از array به جای arraylist:

به این دلیل که باز هم کد در قسمت initial کردن arraylist دچار مشکل می‌شد به دنبال آن رفتیم که به طور کلی arraylist را حذف کنیم و به جای آن از array استفاده کنیم که مشکلات size داینامیک را نداشته باشد و همچنین نیاز ما را برطرف کند.

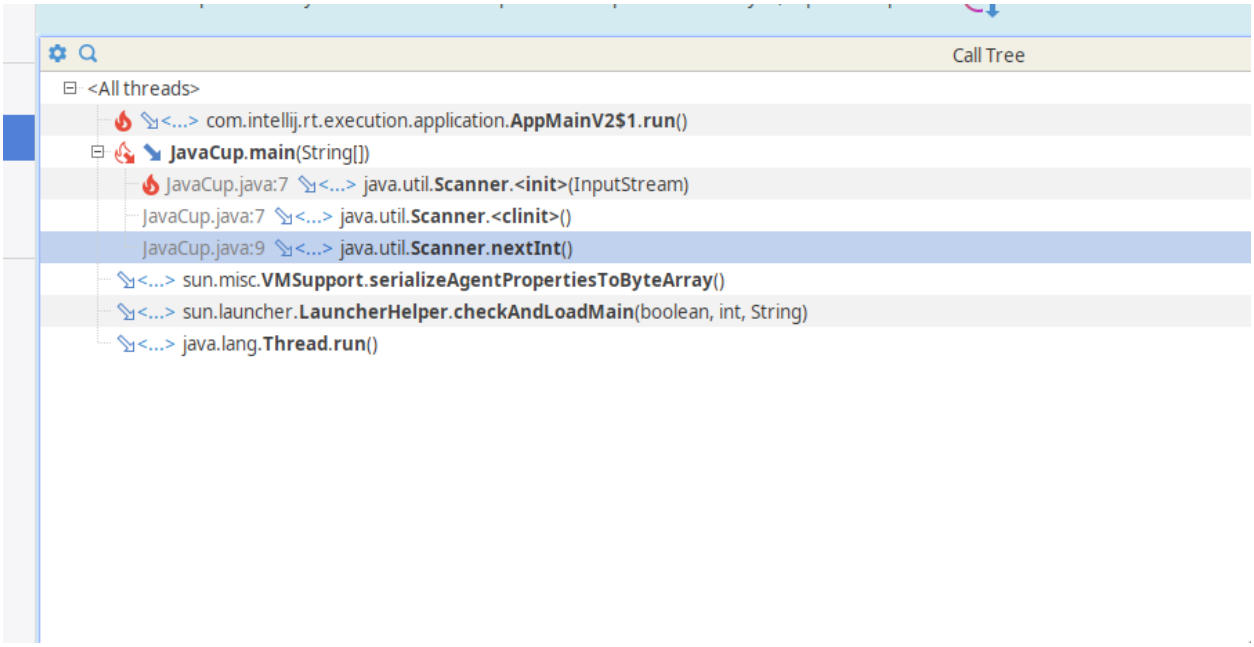
با انجام این عمل کد به شکل زیر تغییر کرد:

```
    }
    public static void eval(int i, int j, int k)
    {
        if (i * i + j * j == k * k || i * i == j * j + k * k || j * j == i * i + k * k)
        {
            System.out.println("YES");
        }
        else { System.out.println("NO"); }
    }
    public static void temp() {
        int[] a = new int[10000 * 20000 + 2];
        for (int i = 0; i < 10000; i++)
        {
            for (int j = 0; j < 20000; j++) {
                a[i * 20000 + j] = i + j;
            }
        }
    }
}
```

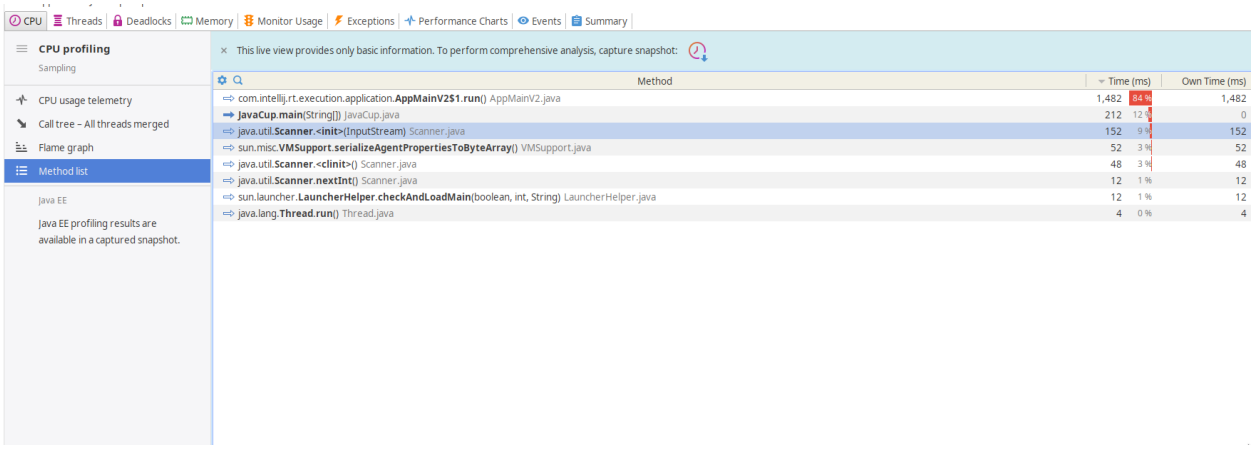
```
JavaCup
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
[YourKit Java Profiler 2021.3-b235] Log file: /home/hamidreza/.yjp/log/JavaCup-14936.log
Press number1:
3
Press number2:
4
Press number3:
5
YES

Process finished with exit code 0
```

که همانطور که می‌بینیم دیگر برنامه بیشترین زمان مصرفی‌اش در زمان ورودی گرفتن است و باقی کد در زمان اندکی به پایان می‌رسد و همچنین برای ورودی 3 4 5 خروجی yes را می‌دهد.

Call tree:



Method list:



کدها در ریپو گروه قرار گرفته است و همچنین به پیوست تقدیم شده.