实验课程名称：__软件工程基础实验_____

实验项目名称	代码评审与程序性能优化			实验成绩	
实 验 者	王汉成	专业班级	软件 1804	组　　别	
同 组 者	王龙祥			实验日期	

第一部分：实验预习报告（包括实验目的、意义，实验基本原理与方法，主要仪器设备及耗材，实验方案与技术路线等）

一、实验目的

1）了解代码审查的含义；

2）了解如何对程序进行性能优化；

3）掌握配置工具的安装与使用。

二、实验内容及要求

1）针对前面实验中所完成的代码，进行代码评审(走查)和性能分析，从时间性能角度对代码进行优化；

2）练习代码评审的两个方面：静态分析、动态分析；

3）使用以下四个工具完成实验：

- Checkstyle
- FindBugs
- PMD
- JTest

三、实验意义

通过对程序进行代码评审和性能分析，能够了解已完成程序存在的不规范地方和一些问题，能够从时间性能角度对代码进行优化，通过静态分析和动态分析，对整体程序性能有一个大致的把握，对于后续的程序改进有着极大的帮助。

四、问题描述

（1）在 Eclipse 中配置代码审查与分析工具。要求学生采用屏幕截图的方式给出在 Eclipse 中配置 Checkstyle、PMD、Findbugs 和 JTest 的过程。

（2）分别使用这些工具对原始代码进行评审和性能分析，记录结果，期间不要有任何修改。

（3）对工具执行结果进行人工分析，对四种工具的分析结果进行对比，找到它们发现问题的能力差异。

（4）根据结果对源代码进行修正（代码规范、性能）；

（5）重新使用工具进行评审和性能分析，直到无法再改进为止

五、主要仪器设备及耗材

设备：PC

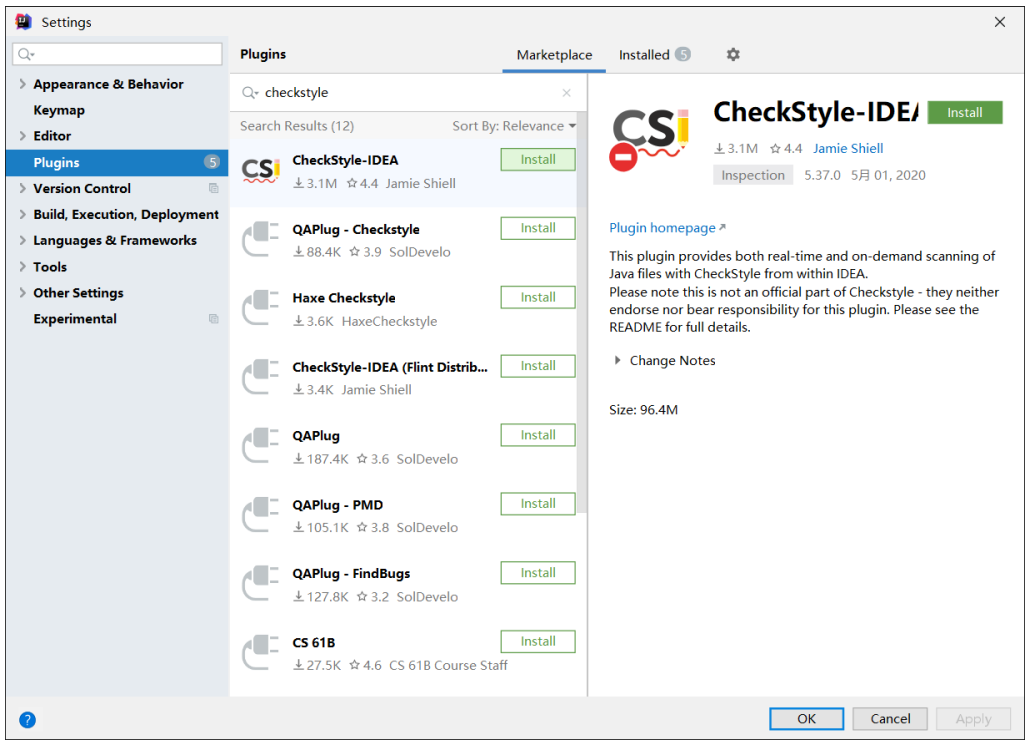
开发环境：Idea、Eclipse

第二部分：实验过程记录（可加页）（包括实验原始数据记录，实验现象记录，实验过程发现的问题等）

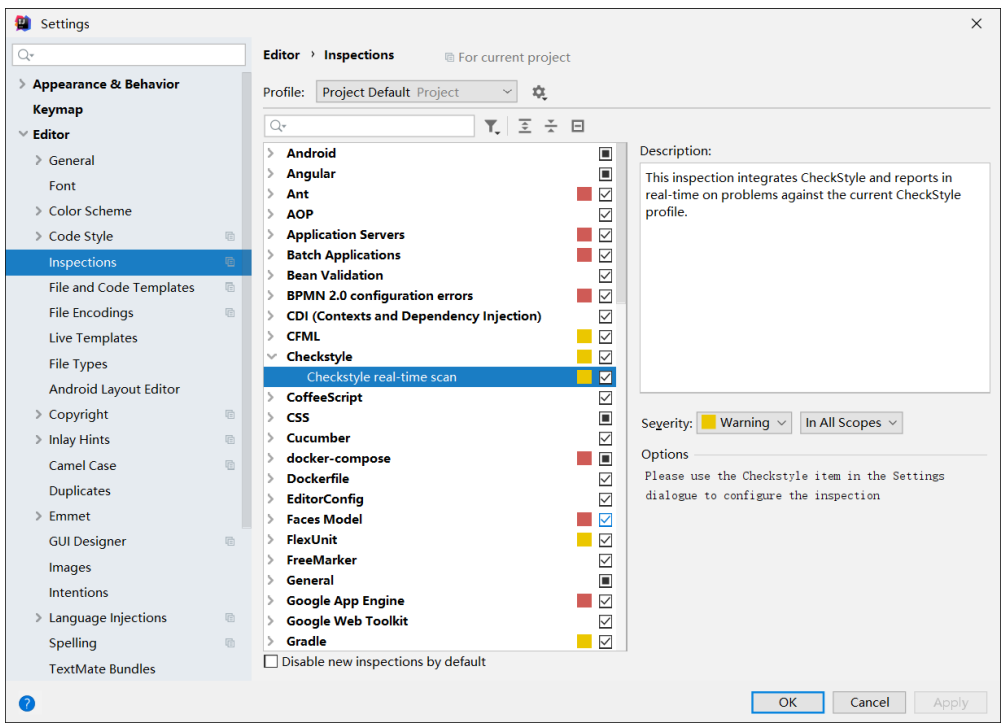
一、算法设计思路

（1）Checkstyle

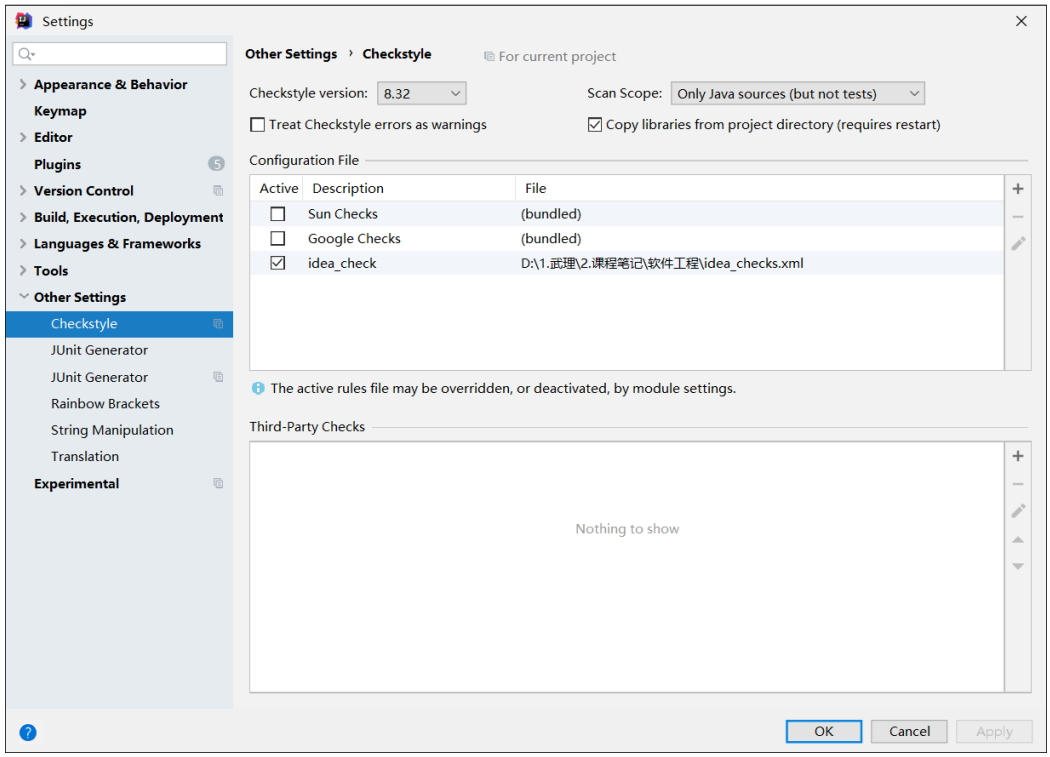
安装 Checkstyle 插件：在 Setting 中的 Plugins 模块直接搜索 Checkstyle 安装即可



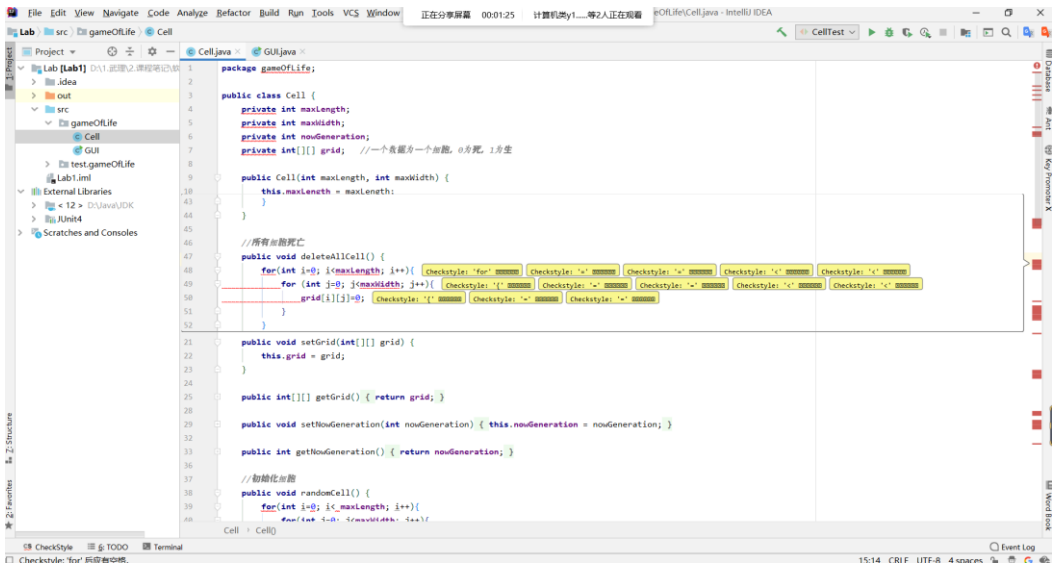
检查插件是否安装成功，在 Editor 中的 Inspections 查看，勾选即可



插件完成安装后，需要对 Checkstyle 进行配置，打开 Checkstyle 的相关配置，导入检查所用的 xml 文件，勾选应用即可



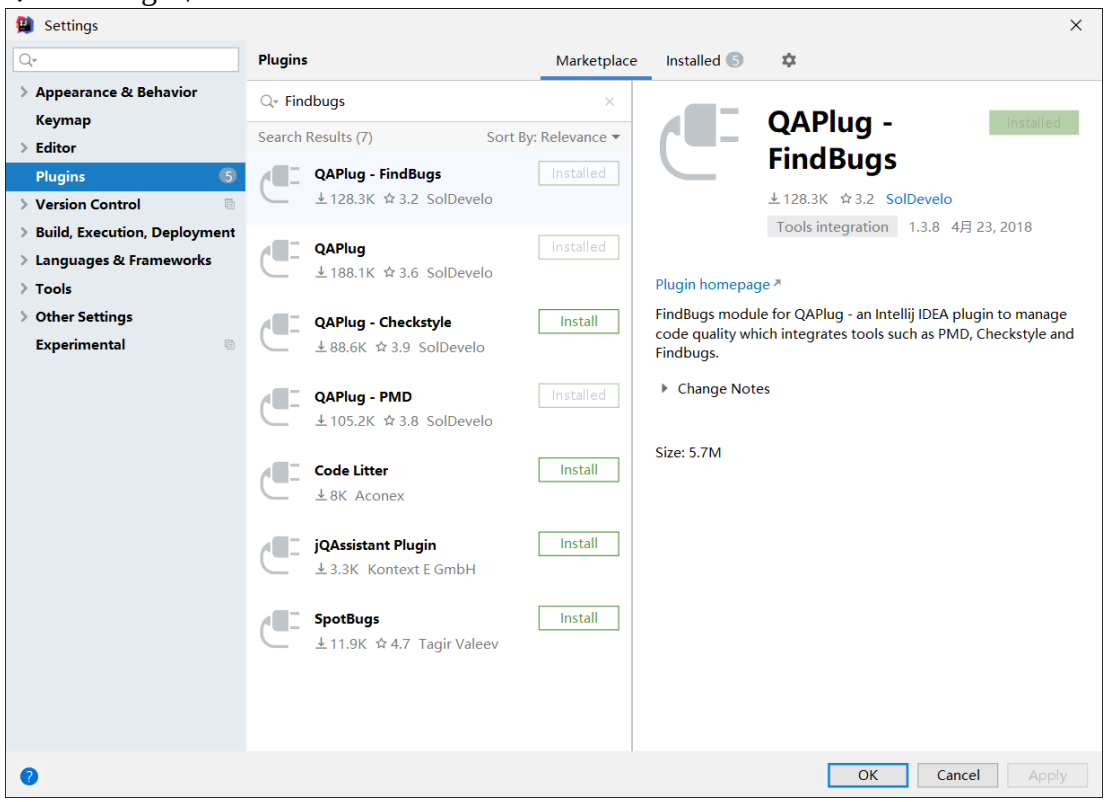
将插件应用之后，Idea 将自动把检查出来的问题现实在页面上，基本上都是编程格式的错误，例如大括号的位置不对，空格的位置不正确等。



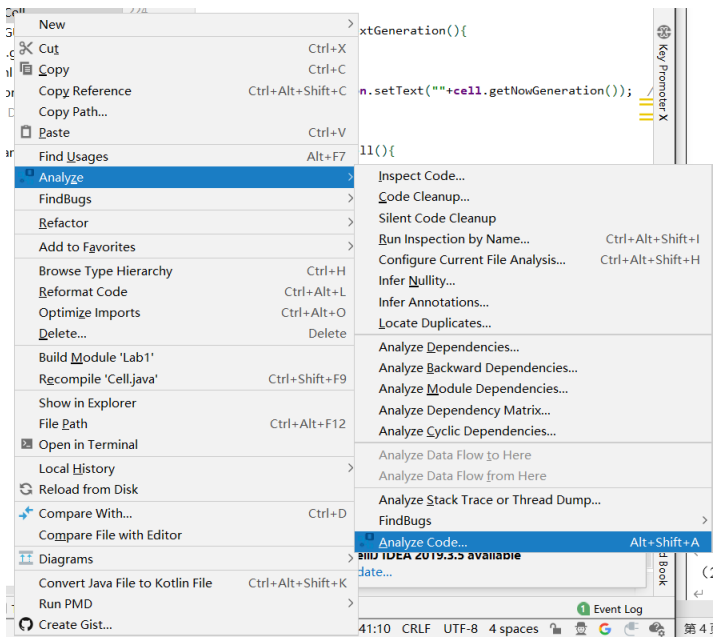
（2）FindBugs

FindBugs 在 Idea 的安装中有两种插件，一种是可以在插件仓库中直接搜索出来的，名为 QA-FindBugs；另一种无法在插件仓库中直接搜索到，需要在插件网上下载后使用，首先来介绍 QA-FindBugs 的安装方法：

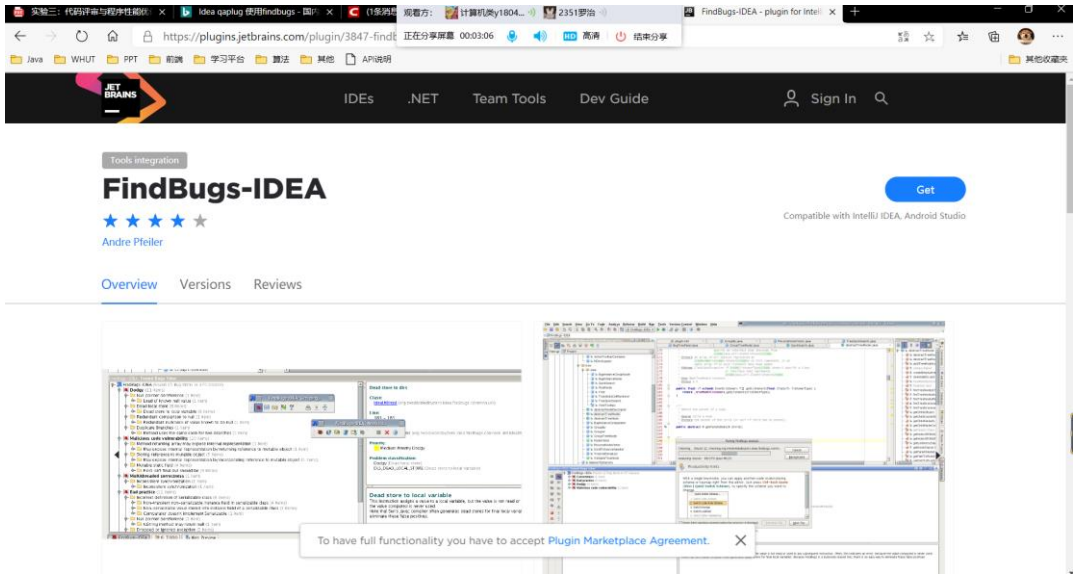
直接在搜索中搜索 FindBugs 即可，QA 是一款 Intellj 中集成的代码质量检测插件，包含了 FindBugs 和 PMD：



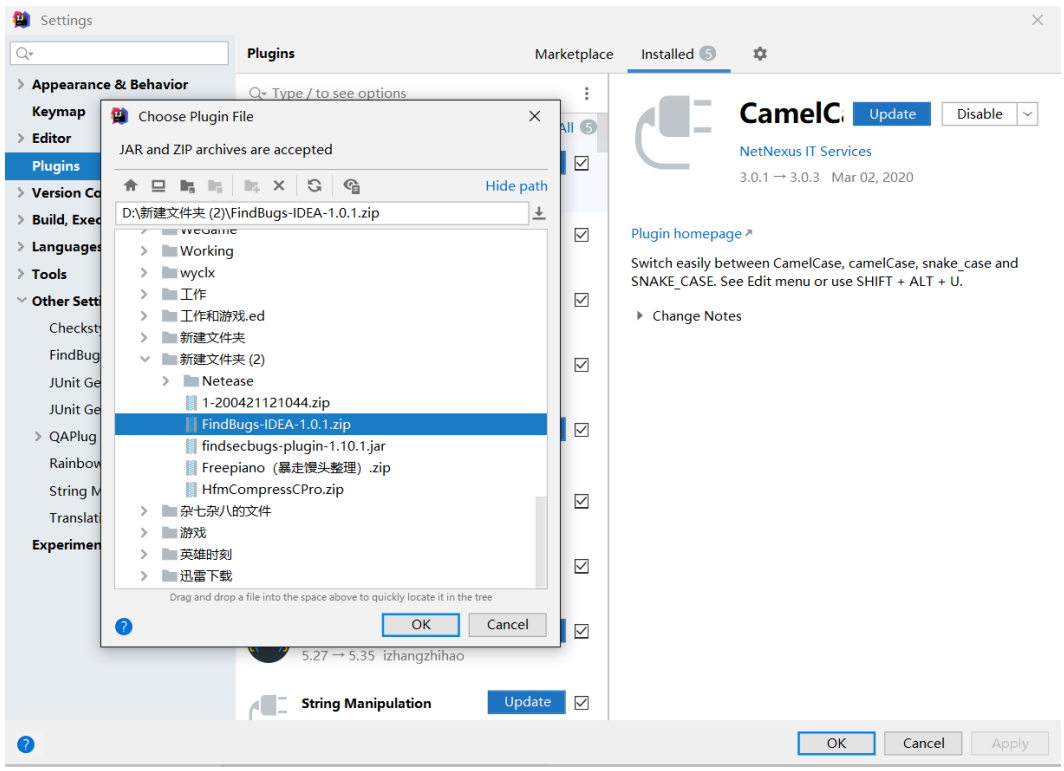
使用 QA-FindBugs 时，右键需要测试的类，点击 Analyze code 即可



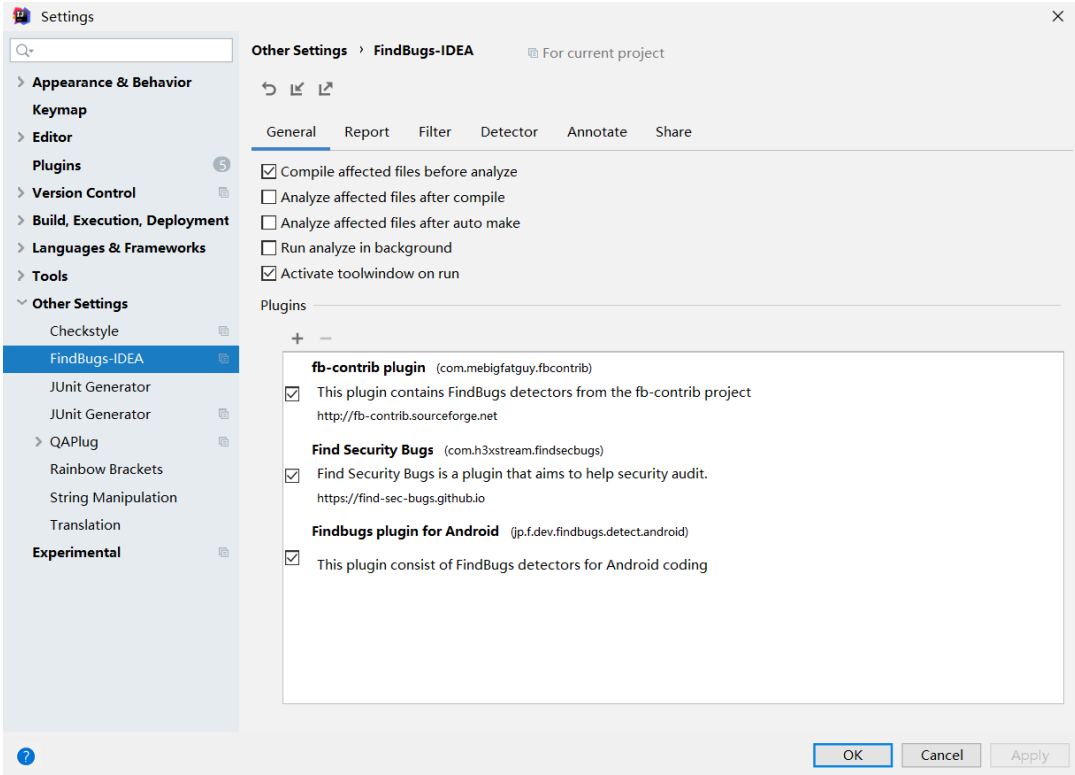
另一个 FindBugs 插件是使用插件网上的下载的 FindBugs-Idea，与 QA-FindBugs 不同，FindBugs-Idea 只有 FindBugs 的相关功能，没有集成其他的测试功能，首先需要在官网下载安装包:



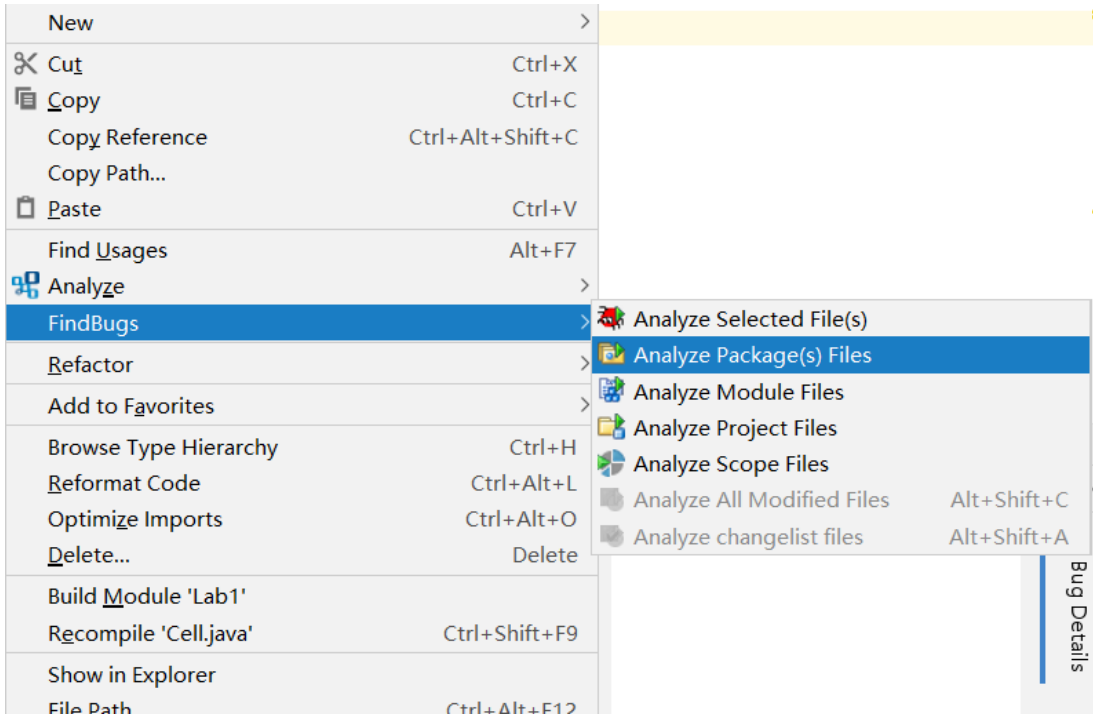
将 FindBugs-Idea 的压缩包下载好了之后需要导入到插件仓库中使用，具体配置方法如下：在 Plugins 中点击设置按钮，导入下载好的插件，确认即可。



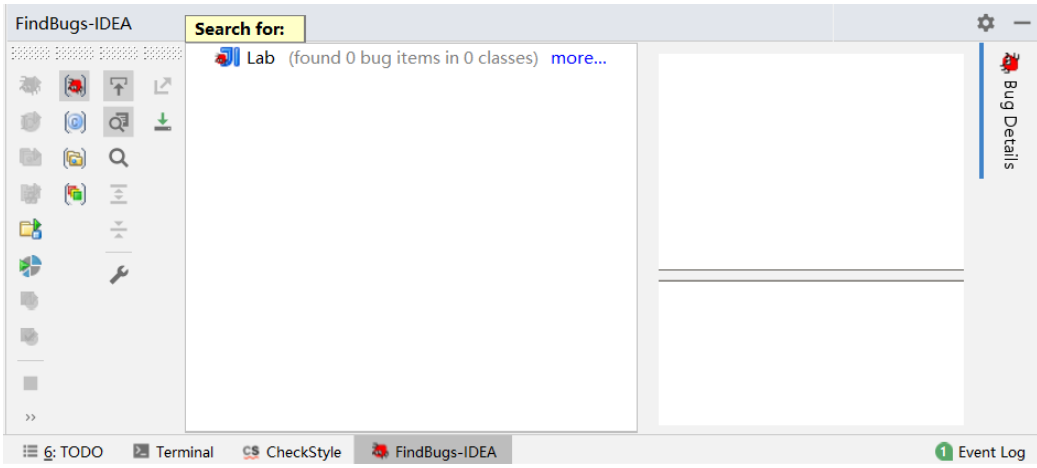
导入之后同样需要检查是否安装成功，同时导入相关检查插件，我这里导入了 fb-contrib plugin、Find Security Bugs 和 FindBugs plugin for Android：

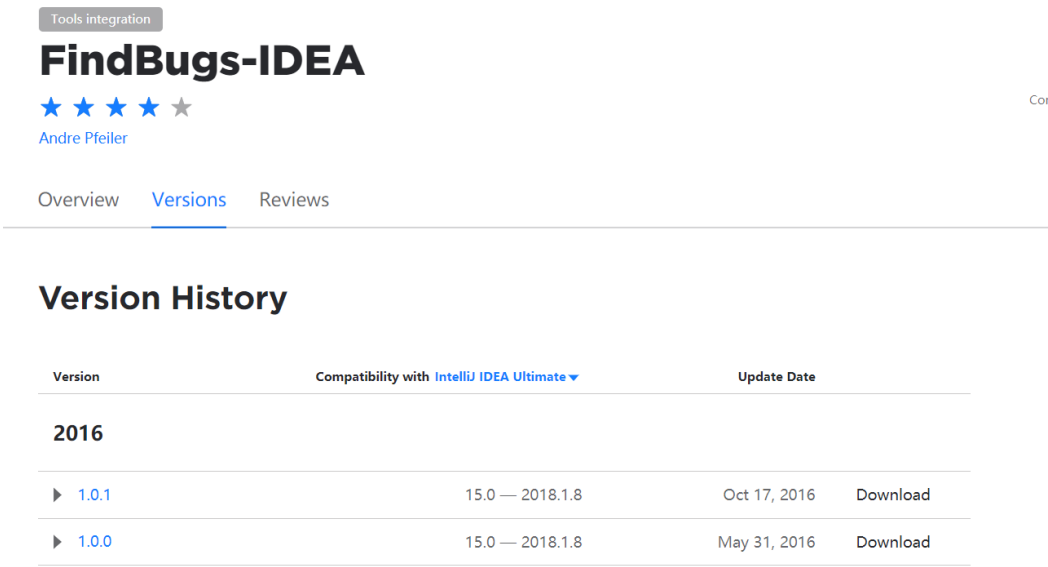


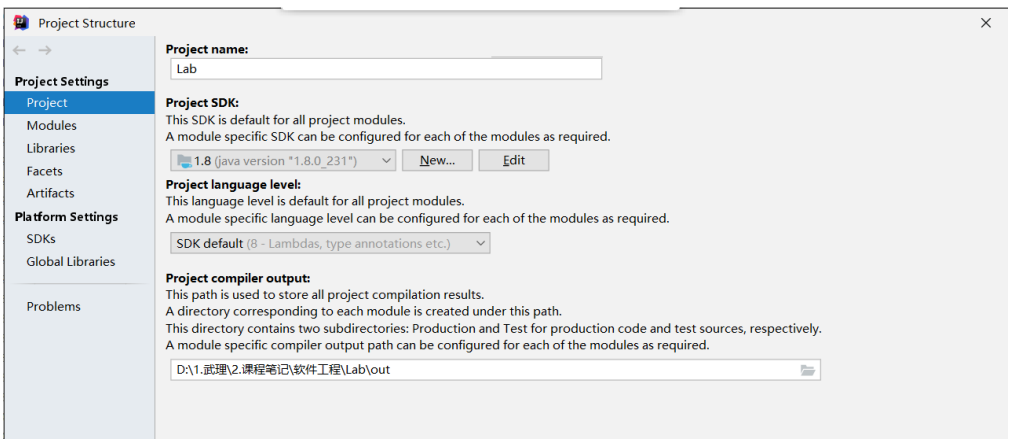
配置完成后会在 Idea 左下角中出现相关的模块，也可右键点击 Java 类，使用 FindBugs 功能:



需要注意的是，在实验过程中，我们发现 FindBugs-Idea 无法查找到 Bug，提示“found 0 bug item in 0 classes”，在查找了相关资料后发现，FindBugs 最后一个版本更新时间为 16 年，而那时候的 JDK 版本为 8，我们使用的是 JDK12，所以怀疑是 JDK 版本的问题，在重新安装了 JDK8 之后，FindBugs 功能可以正常使用了。

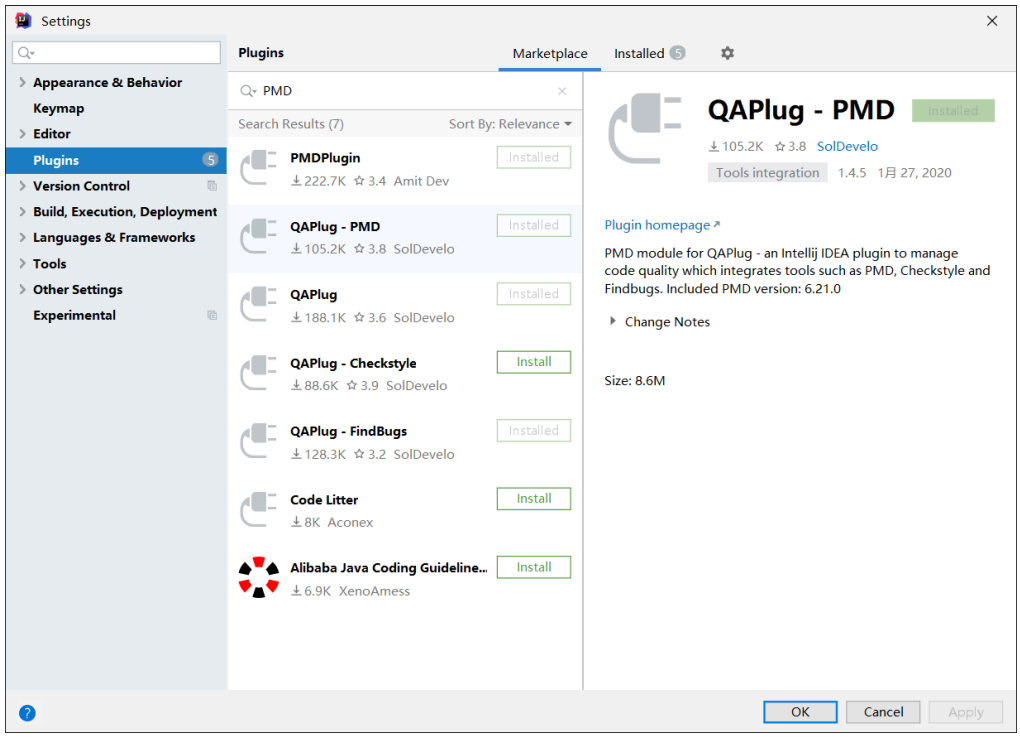




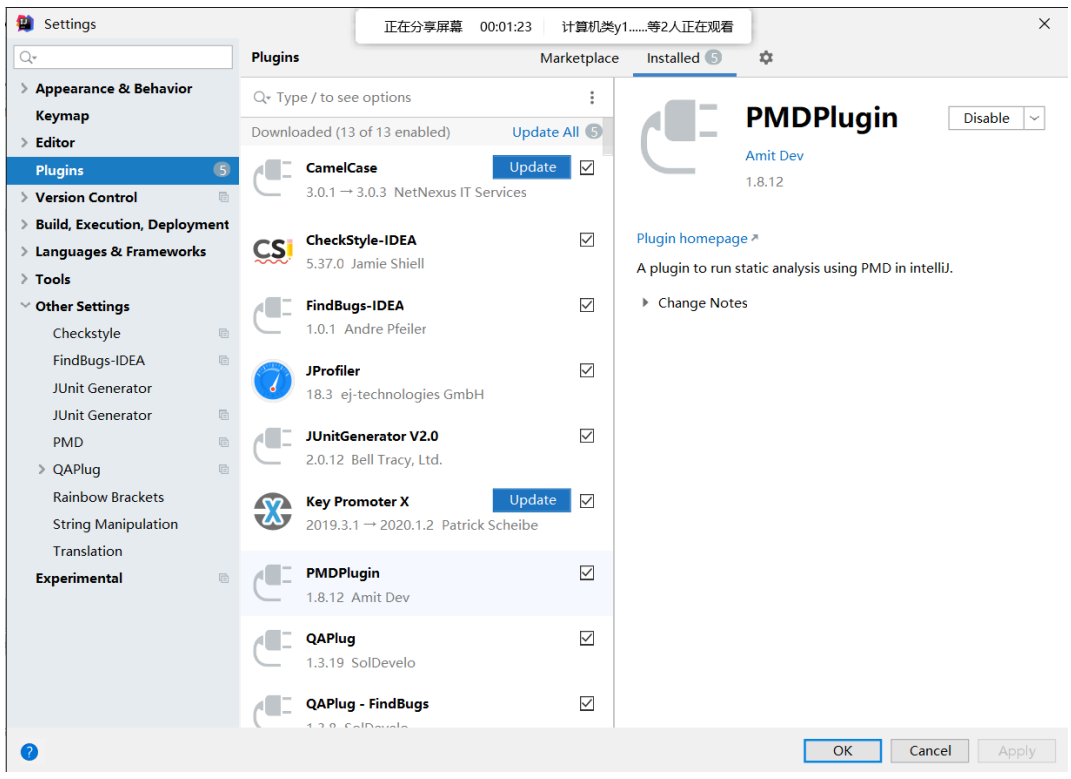


（3）PMD

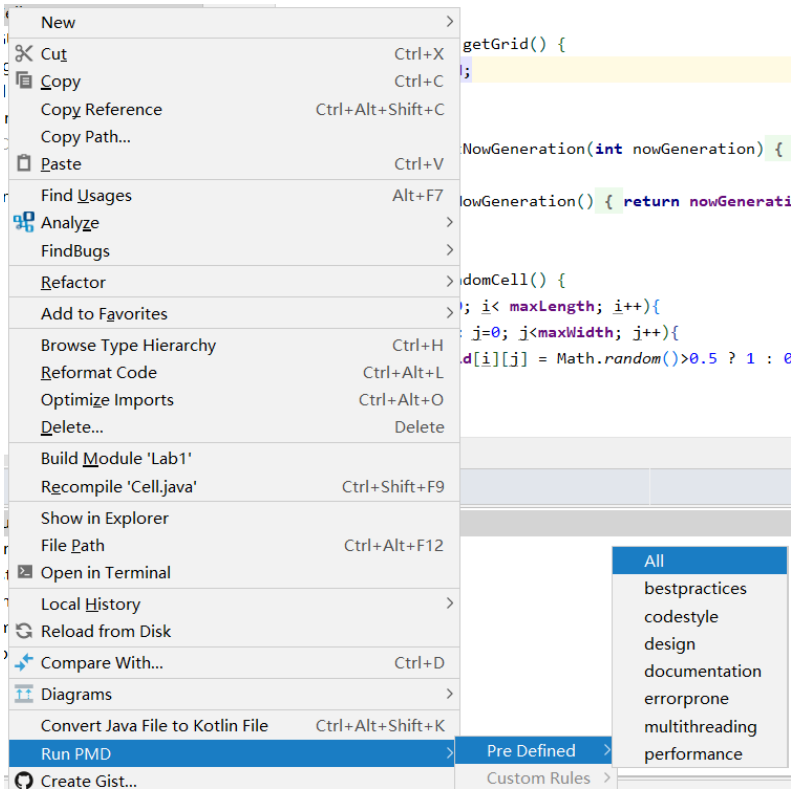
PMD 的安装同样有两个插件，我在实验中也分别进行了配置和比较，首先是 QAPlug-PMD，安装方法与 QAPlug-FindBugs 相同，安装完即可使用



另外一个插件为 PMDPlugin，该插件也是单独的一个 PMD 测试插件：

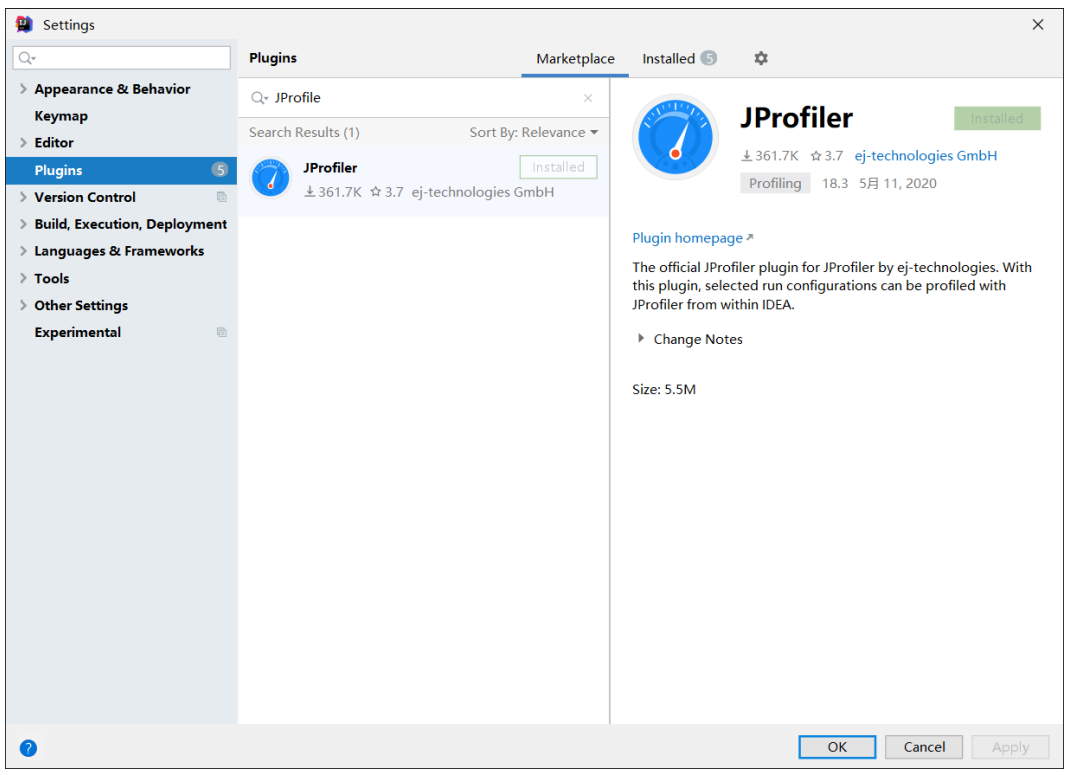


安装完成之后，右键点击项目，选择 Run PMD 即可运行。

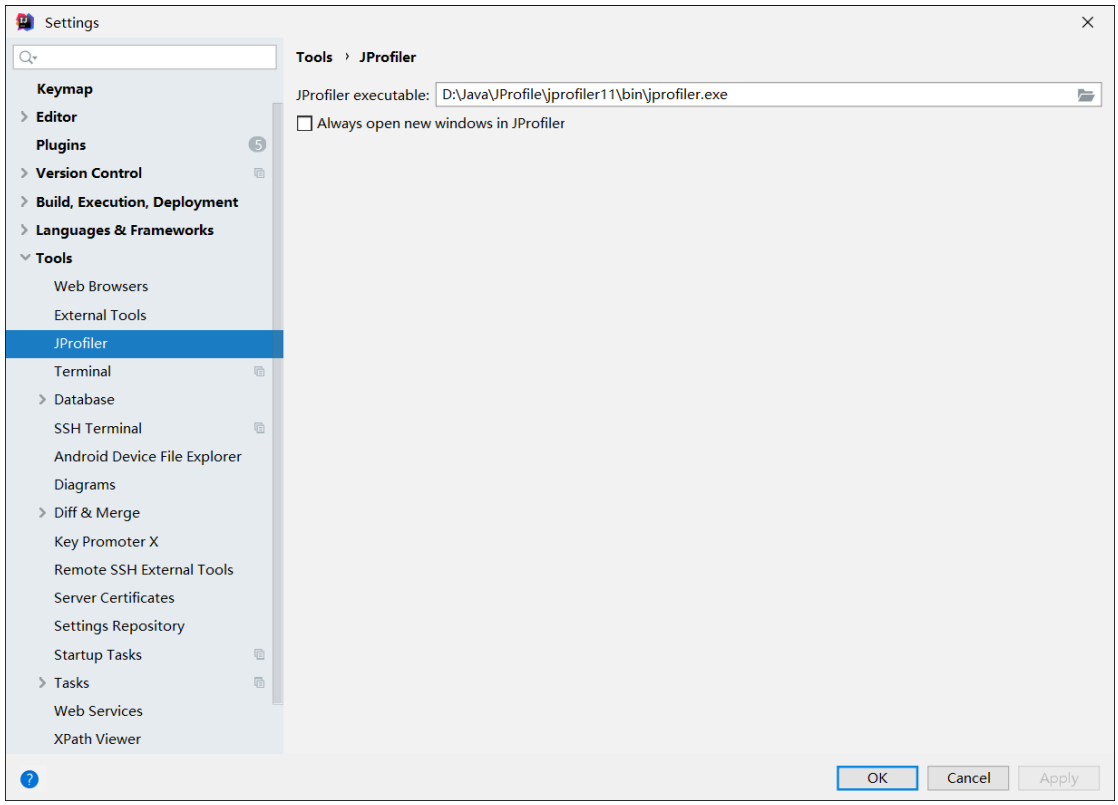


（4）JProfile

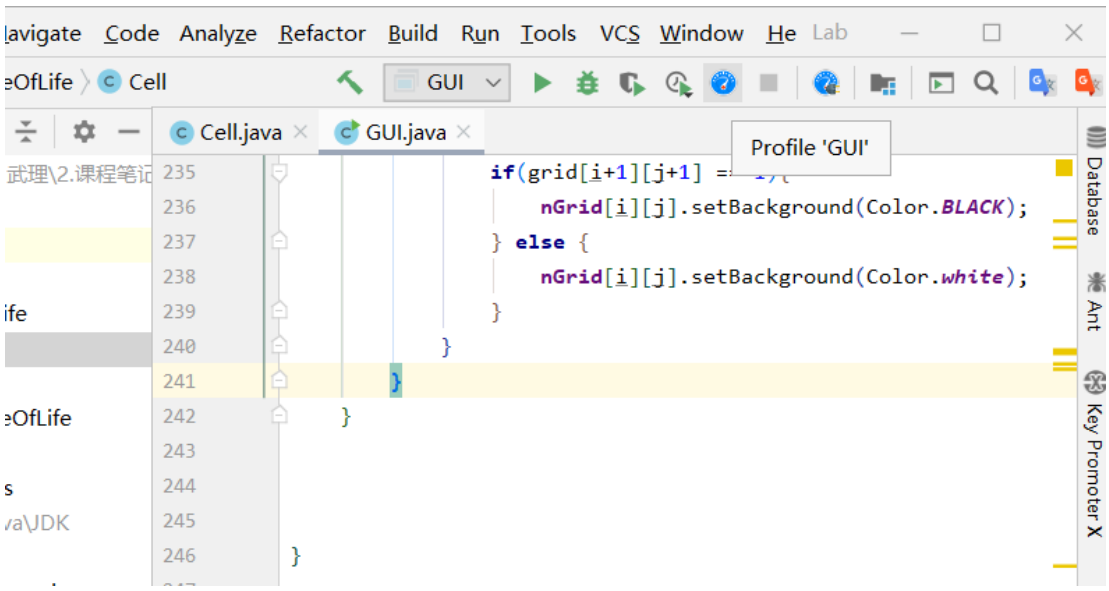
JProfile 的安装同样是在插件仓库中直接搜索安装即可：



在安装完成之后需要对 JProfile 进行配置，在设置中的 Tools 模块下，导入下载好的 JProfile 的应用，以实现在进行性能优化的时候打开外部的 JProfile



安装完成后，在 Idea 右上角，运行程序的地方会出现一个优化程序按钮，点击进入即可

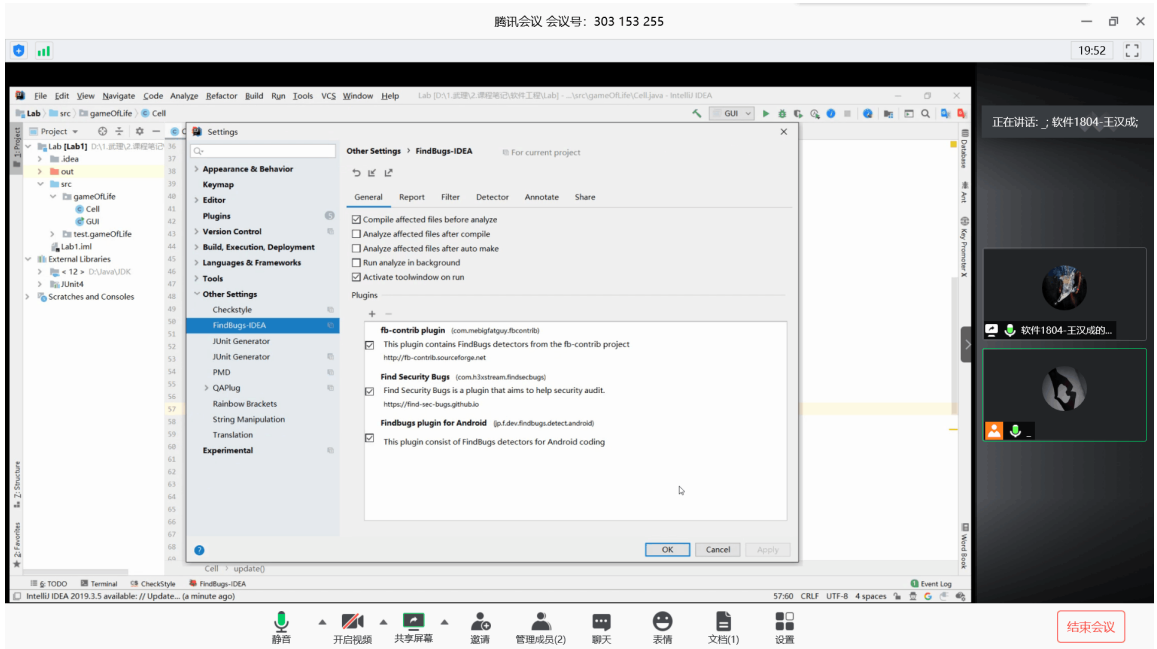


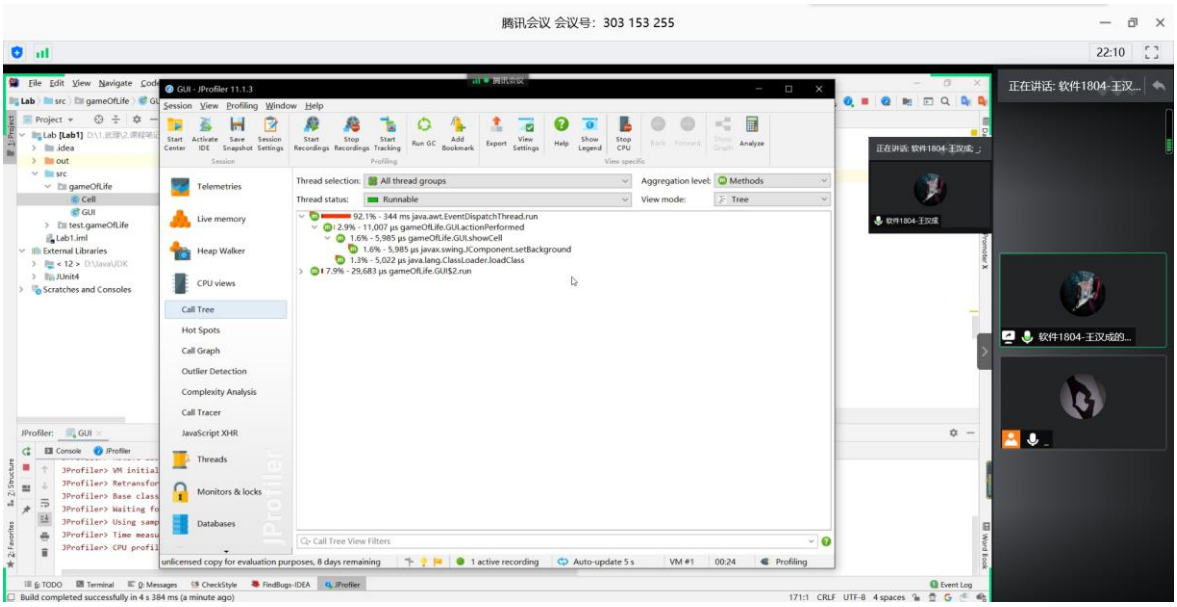
二、结对编程过程

（1）任务分工表

时间	Driver	Observer	主要工作
2020.05.27　20：30-21：30	王汉成	王龙祥	对与 Idea 的相关插件进行配置使用
2020.05.30　15：00-17：00	王龙祥	王汉成	对于 Eclipse 的相关插件进行配置使用

（2）工作照片

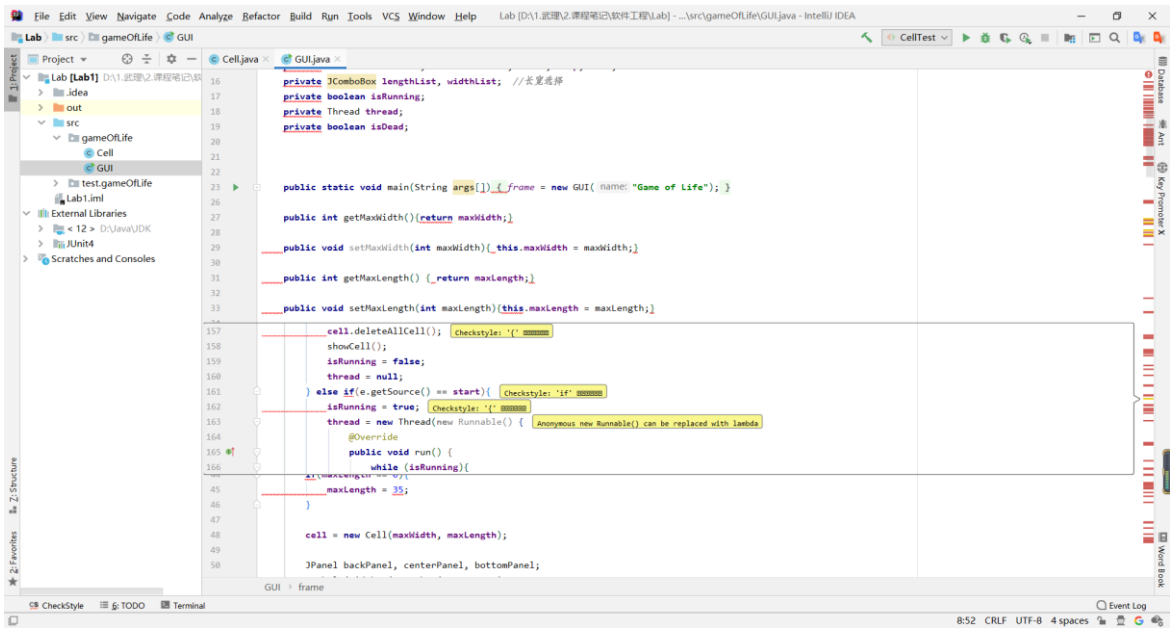




第三部分　结果与讨论（可加页）

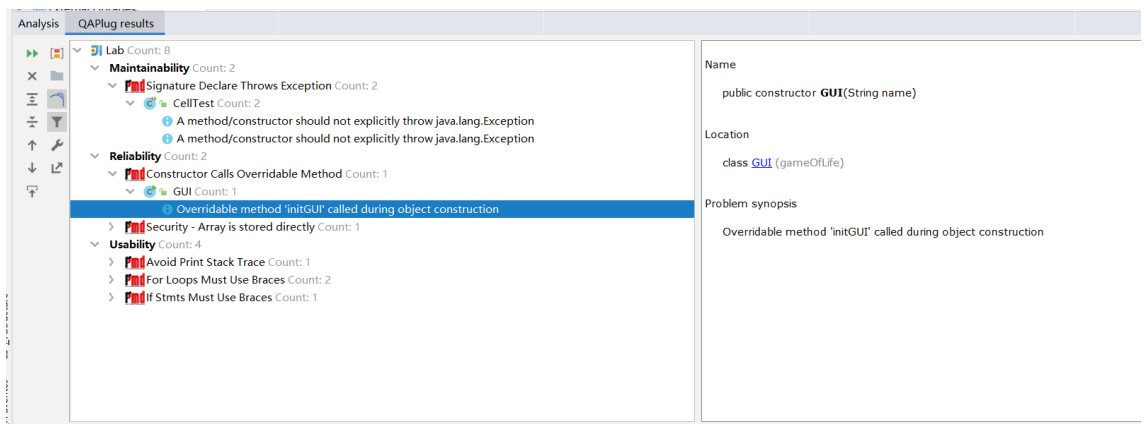
一、实验结果分析（包括数据处理、实验现象分析、影响因素讨论、综合分析和结论等）

（1）Checkstyle 运行结果：

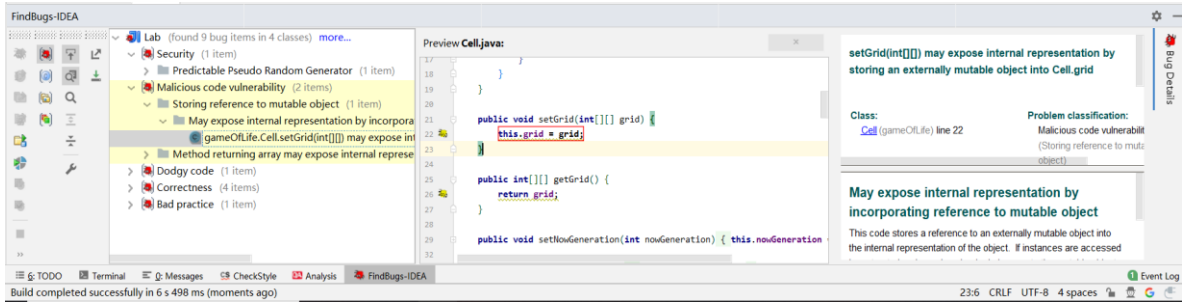


（2）FindBugs 运行结果：

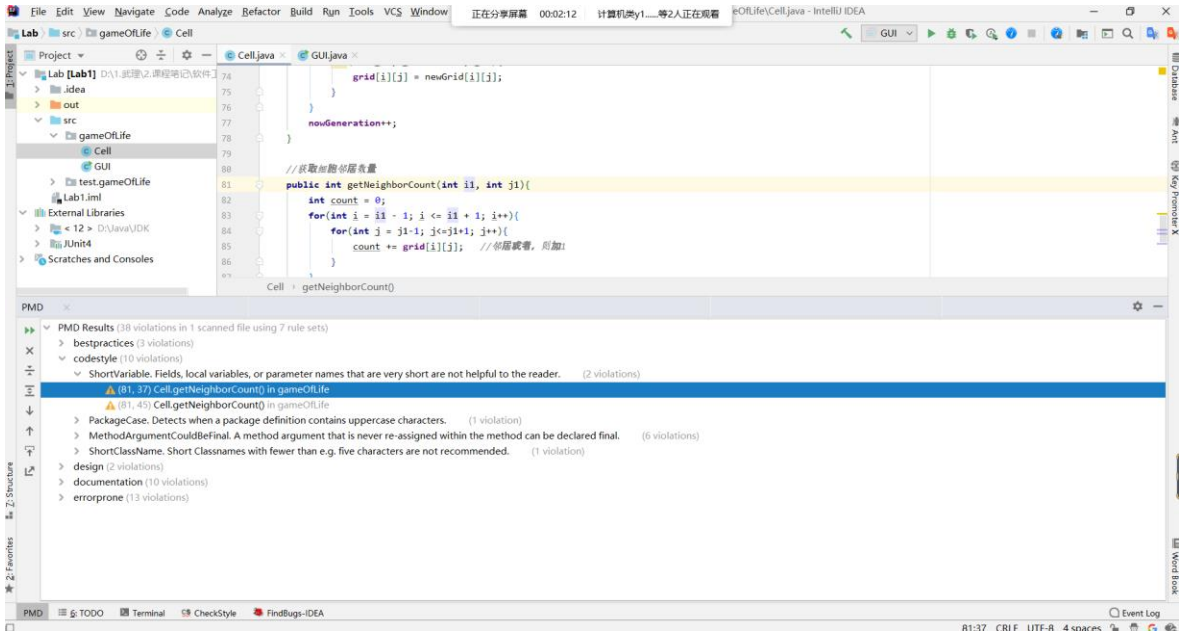
QAPlugin-FindBugs：



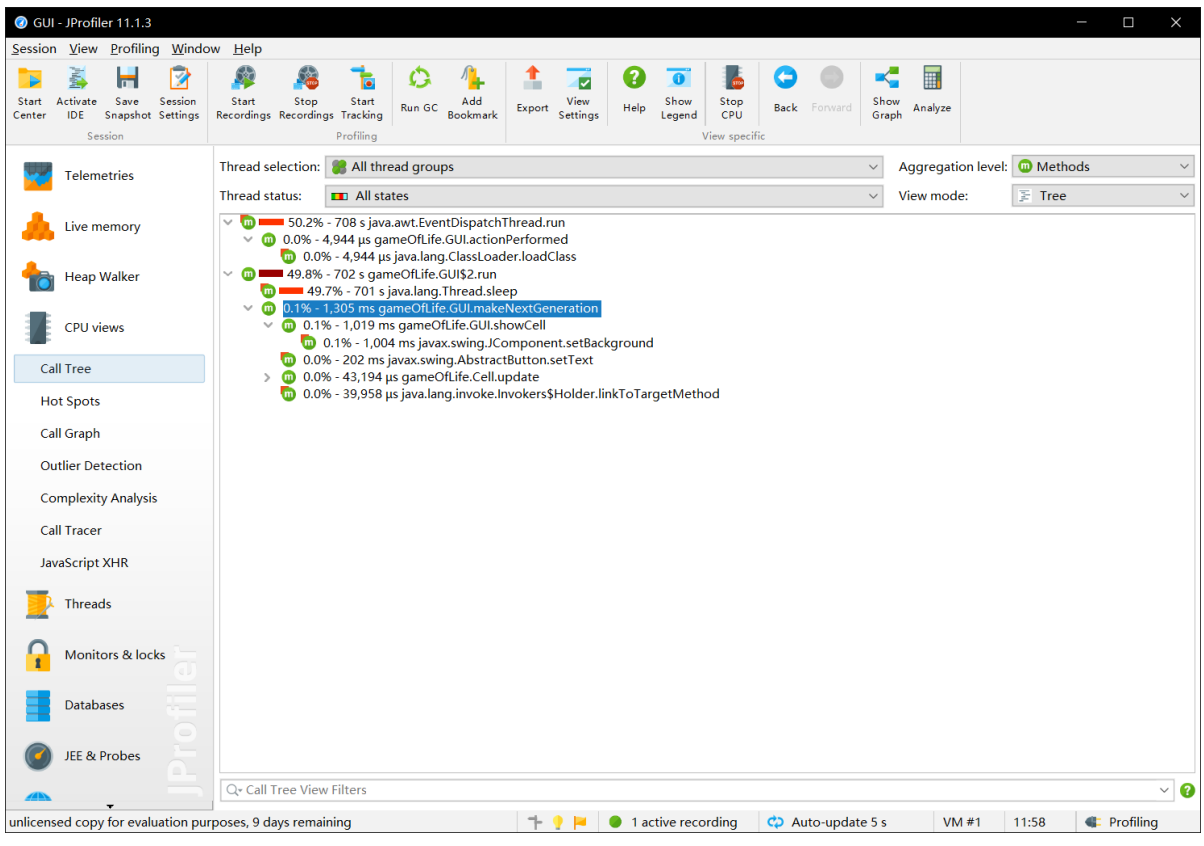
FindBugs-Idea：



（3）PMD 运行结果：



（4）JProfile 运行结果：



经过评审，发现耗时占比 50.2%的代码块没有修改的空间，选择优化 makeNextGeneration 函数中的 update 方法，该方法修改前耗时 1305ms，代码如下：

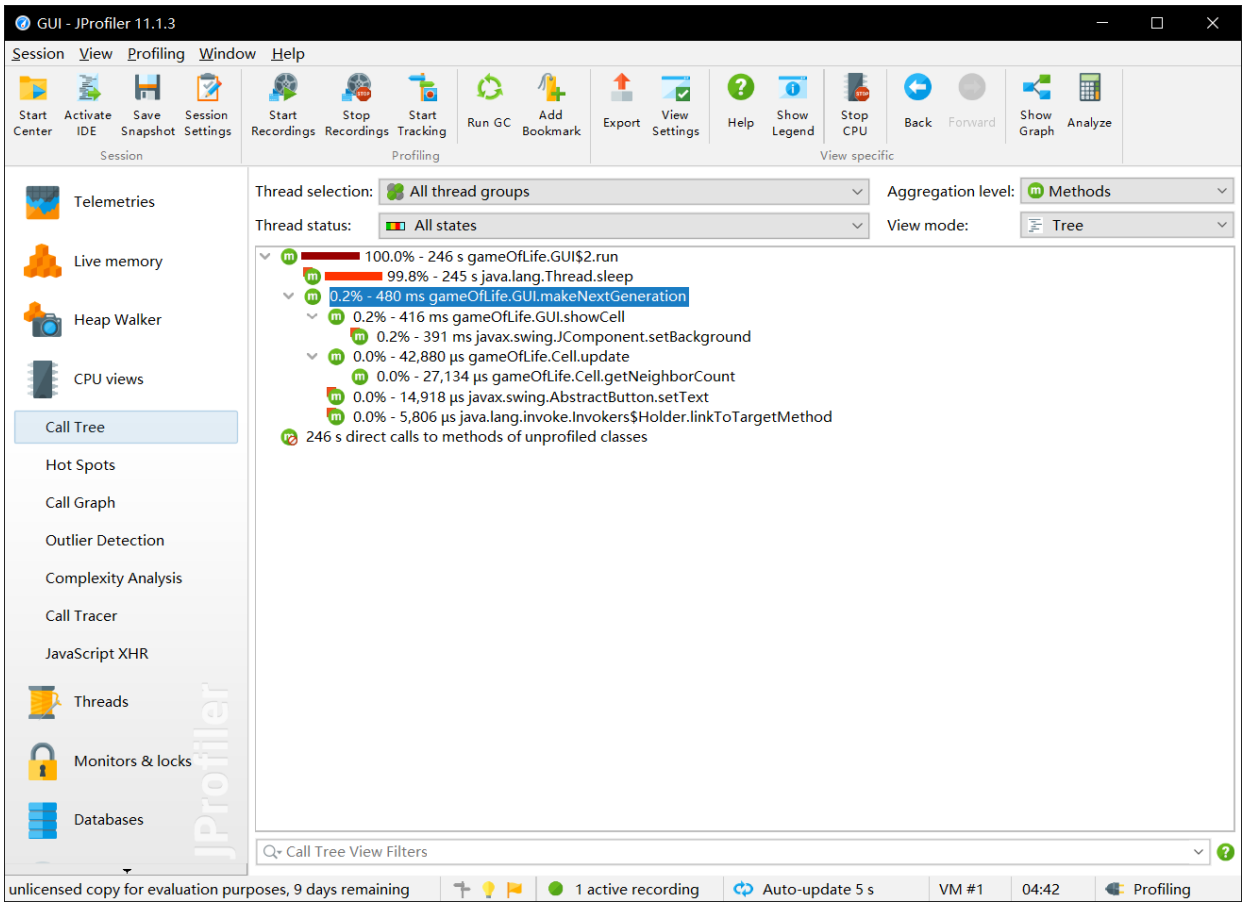
```
public void update() {
    int[][] newGrid = new int[maxLength+2][maxWidth+2];
```

```
    for(int i=1; i<=maxLength; i++){
        for(int j=1; j<=maxWidth; j++){
            switch (getNeighborCount(i, j)){
                case 2:
                    newGrid[i][j] = grid[i][j];   //细胞状态保持不变
                    break;
                case 3:
                    newGrid[i][j] = 1;      //细胞活着
                    break;
                default:
                    newGrid[i][j] = 0;      //细胞死亡
            }
        }
    }
    for(int i=1; i<=maxLength; i++){
        for(int j=1; j<=maxWidth; j++){
            grid[i][j] = newGrid[i][j];
        }
    }
    nowGeneration++;
}
```

评审后发现，第二个二层循环没有存在的必要，完全可以放在第一个循环中运行，修改后的代码如下：

```
public void update() {
    int[][] newGrid = new int[maxLength+2][maxWidth+2];
    for(int i=1; i<=maxLength; i++){
        for(int j=1; j<=maxWidth; j++){
            switch (getNeighborCount(i, j)){
                case 2:
                    newGrid[i][j] = grid[i][j];   //细胞状态保持不变
                    break;
                case 3:
                    newGrid[i][j] = 1;      //细胞活着
                    break;
                default:
                    newGrid[i][j] = 0;      //细胞死亡
            }
            grid[i][j] = newGrid[i][j];
        }
    }
    nowGeneration++;
}
```

修改后的 JProfile 时间检查：



明显可以看出来 makeNextGeneration 方法的占用时间变短，程序性能提高了。

二、实验小结及体会

本次实验让我了解了代码审查的含义，了解了如何对程序进行性能优化，同时掌握了配置代码审查的相关工具的安装及使用，通过对代码审查工具的安装和使用，让我掌握了编译环境下插件的使用方法，通过不同的插件集成的功能，能够快速便捷地实现某些辅助功能，能够帮助我们更好地优化代码。

在审查工具的安装过程中，我遇到了许多问题，主要是不同插件的功能对比，在本次实验的 FindBugs 和 PMD 插件中，我用了两种不同的插件，得到的结果大同小异，但是仍然存在着区别，如何有效利用这些不同的插件实现我们所需要的功能，对于提升我们写代码的效率和审查效率都有着极大的帮助。

通过审查工具找到的问题修改代码，提高代码效率从而缩短运行时间，有利于我们改进和优化代码，习惯使用不同的审查工具后也能够让我们的代码书写的越来越规范，对于自身未来的发展也大有好处。

成绩评定表：

序号	评分项目	满分	实得分
1	**实验报告格式规范**	**2**	
2	**实验报告过程清晰，内容详实**	**4**	
3	**实验报告结果正确性**	**2**	
4	**实验分析与总结详尽**	**2**	
	总得分	**10**	