Security Programming

Lab 3.1 & Lab 3.2

Wang Haoyuan

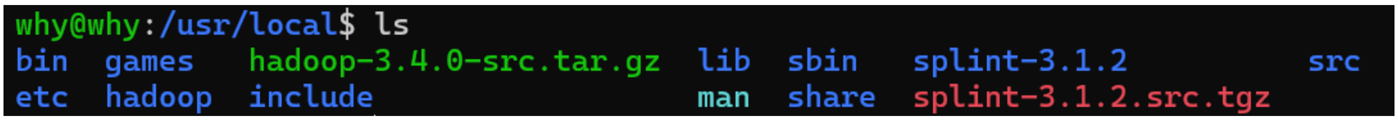
0. 实验目的概述：

第三部分的实验主要目的是通过相关的工具，对编程语言代码进行可能出现的编程漏洞的检测。

Lab 3.1 Using splint for C static analysis

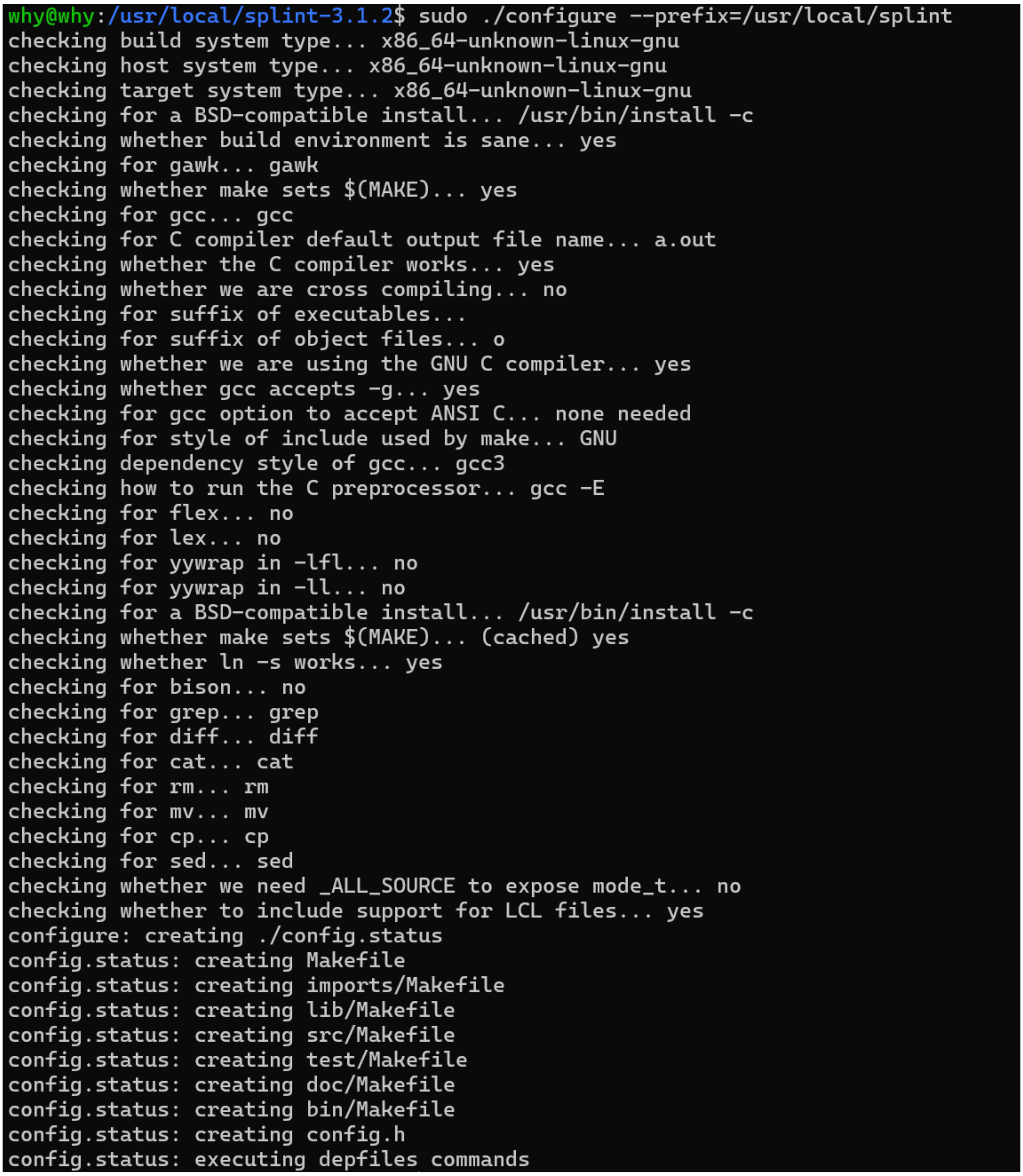
Step 1: 安装splint

通过给出的链接下载splint并在linux环境下解压，之后的文件夹结构如下：



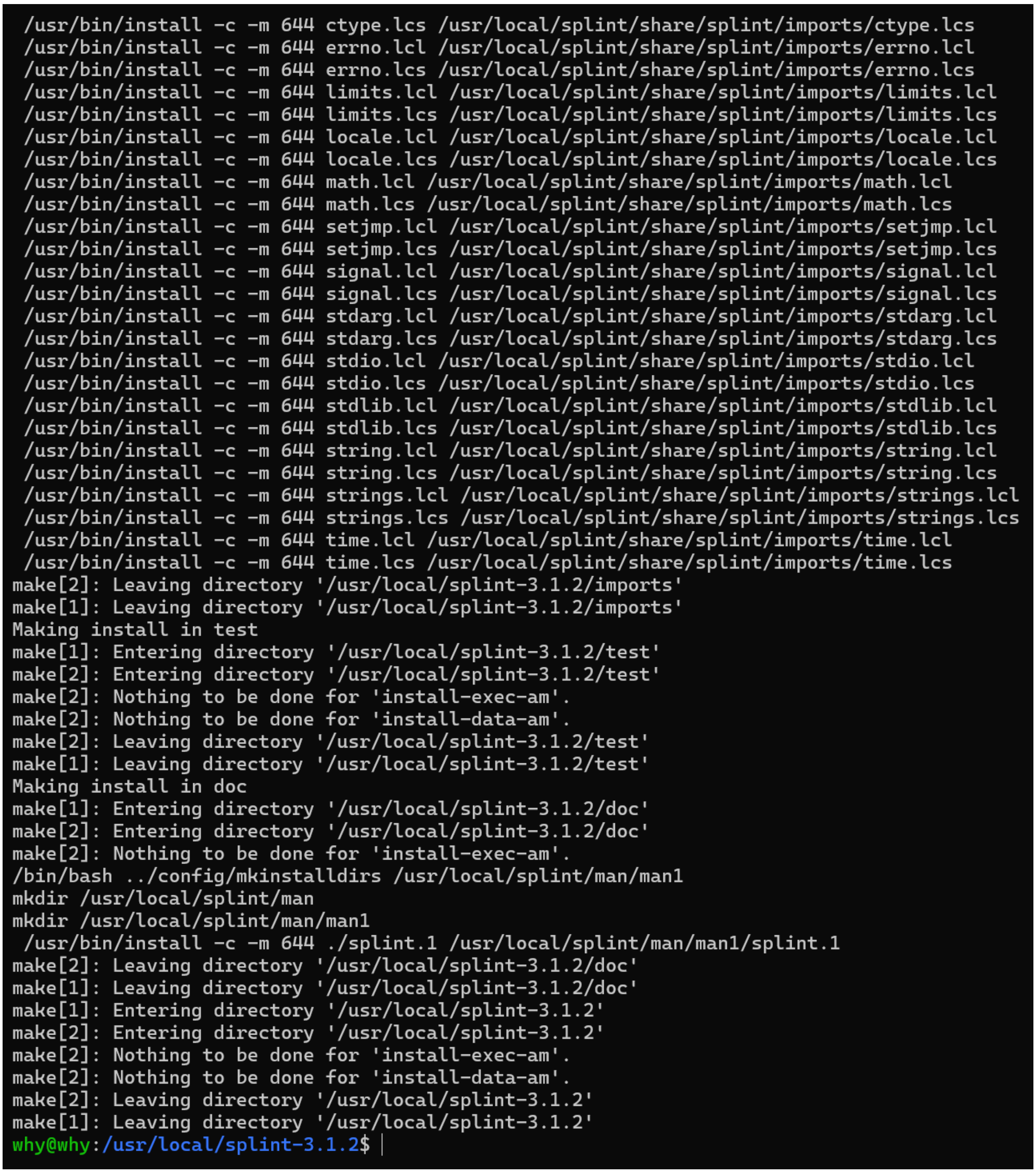
```
why@why:/usr/local$ ls
bin  games   hadoop-3.4.0-src.tar.gz  lib  sbin   splint-3.1.2          src
etc  hadoop  include                  man  share  splint-3.1.2.src.tgz
```

建立相关文件夹后，运行`configure`程序：



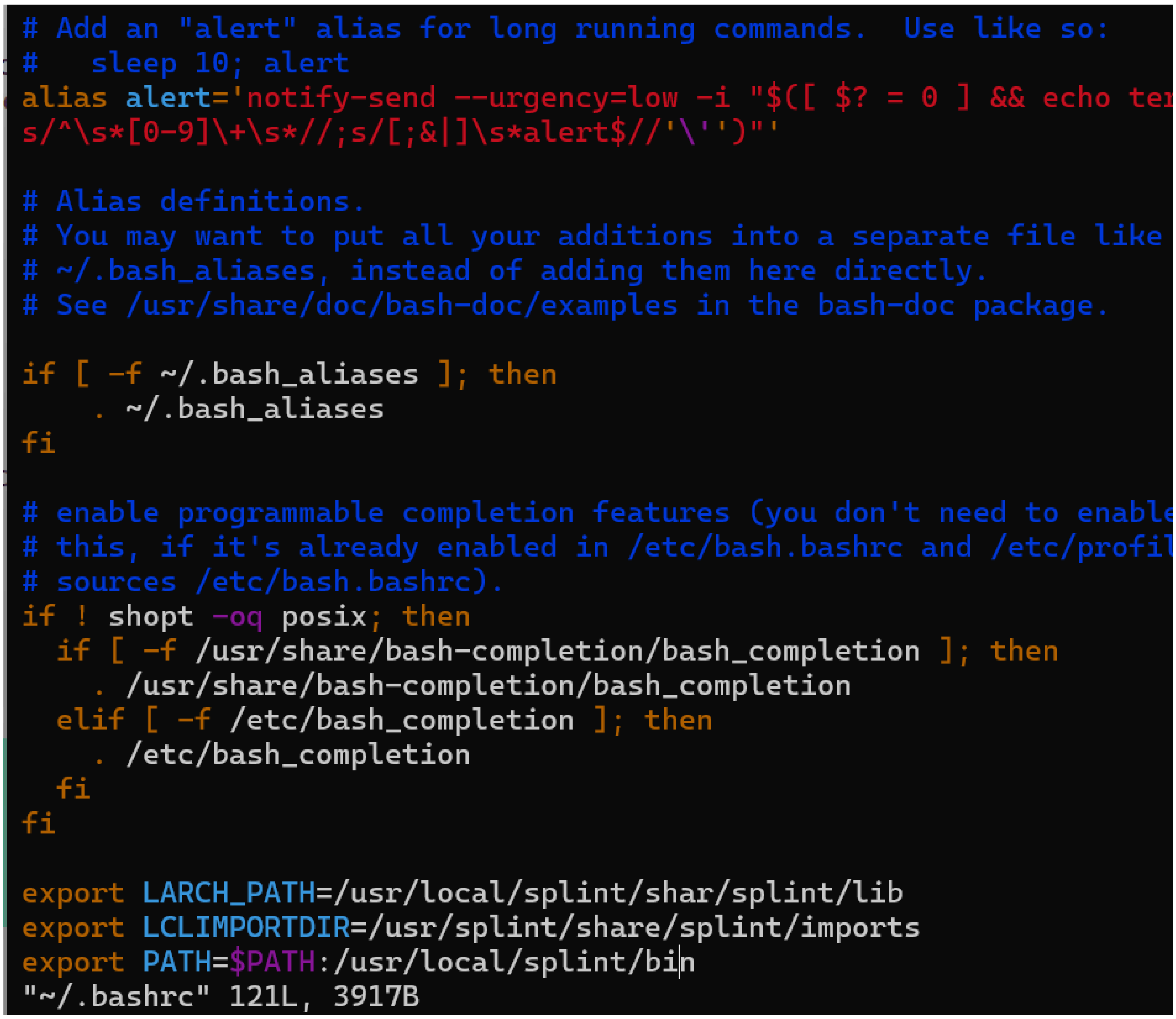
```
why@why:/usr/local/splint-3.1.2$ sudo ./configure --prefix=/usr/local/splint
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking target system type... x86_64-unknown-linux-gnu
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ANSI C... none needed
checking for style of include used by make... GNU
checking dependency style of gcc... gcc3
checking how to run the C preprocessor... gcc -E
checking for flex... no
checking for lex... no
checking for yywrap in -lfl... no
checking for yywrap in -ll... no
checking for a BSD-compatible install... /usr/bin/install -c
checking whether make sets $(MAKE)... (cached) yes
checking whether ln -s works... yes
checking for bison... no
checking for grep... grep
checking for diff... diff
checking for cat... cat
checking for rm... rm
checking for mv... mv
checking for cp... cp
checking for sed... sed
checking whether we need _ALL_SOURCE to expose mode_t... no
checking whether to include support for LCL files... yes
configure: creating ./config.status
config.status: creating Makefile
config.status: creating imports/Makefile
config.status: creating lib/Makefile
config.status: creating src/Makefile
config.status: creating test/Makefile
config.status: creating doc/Makefile
config.status: creating bin/Makefile
config.status: creating config.h
config.status: executing depfiles commands
```

之后在当前文件夹下进行`make`并且运行`sudo make install`进行安装：


```
 /usr/bin/install -c -m 644 ctype.lcs /usr/local/splint/share/splint/imports/ctype.lcs
 /usr/bin/install -c -m 644 errno.lcl /usr/local/splint/share/splint/imports/errno.lcl
 /usr/bin/install -c -m 644 errno.lcs /usr/local/splint/share/splint/imports/errno.lcs
 /usr/bin/install -c -m 644 limits.lcl /usr/local/splint/share/splint/imports/limits.lcl
 /usr/bin/install -c -m 644 limits.lcs /usr/local/splint/share/splint/imports/limits.lcs
 /usr/bin/install -c -m 644 locale.lcl /usr/local/splint/share/splint/imports/locale.lcl
 /usr/bin/install -c -m 644 locale.lcs /usr/local/splint/share/splint/imports/locale.lcs
 /usr/bin/install -c -m 644 math.lcl /usr/local/splint/share/splint/imports/math.lcl
 /usr/bin/install -c -m 644 math.lcs /usr/local/splint/share/splint/imports/math.lcs
 /usr/bin/install -c -m 644 setjmp.lcl /usr/local/splint/share/splint/imports/setjmp.lcl
 /usr/bin/install -c -m 644 setjmp.lcs /usr/local/splint/share/splint/imports/setjmp.lcs
 /usr/bin/install -c -m 644 signal.lcl /usr/local/splint/share/splint/imports/signal.lcl
 /usr/bin/install -c -m 644 signal.lcs /usr/local/splint/share/splint/imports/signal.lcs
 /usr/bin/install -c -m 644 stdarg.lcl /usr/local/splint/share/splint/imports/stdarg.lcl
 /usr/bin/install -c -m 644 stdarg.lcs /usr/local/splint/share/splint/imports/stdarg.lcs
 /usr/bin/install -c -m 644 stdio.lcl /usr/local/splint/share/splint/imports/stdio.lcl
 /usr/bin/install -c -m 644 stdio.lcs /usr/local/splint/share/splint/imports/stdio.lcs
 /usr/bin/install -c -m 644 stdlib.lcl /usr/local/splint/share/splint/imports/stdlib.lcl
 /usr/bin/install -c -m 644 stdlib.lcs /usr/local/splint/share/splint/imports/stdlib.lcs
 /usr/bin/install -c -m 644 string.lcl /usr/local/splint/share/splint/imports/string.lcl
 /usr/bin/install -c -m 644 string.lcs /usr/local/splint/share/splint/imports/string.lcs
 /usr/bin/install -c -m 644 strings.lcl /usr/local/splint/share/splint/imports/strings.lcl
 /usr/bin/install -c -m 644 strings.lcs /usr/local/splint/share/splint/imports/strings.lcs
 /usr/bin/install -c -m 644 time.lcl /usr/local/splint/share/splint/imports/time.lcl
 /usr/bin/install -c -m 644 time.lcs /usr/local/splint/share/splint/imports/time.lcs
make[2]: Leaving directory '/usr/local/splint-3.1.2/imports'
make[1]: Leaving directory '/usr/local/splint-3.1.2/imports'
Making install in test
make[1]: Entering directory '/usr/local/splint-3.1.2/test'
make[2]: Entering directory '/usr/local/splint-3.1.2/test'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/usr/local/splint-3.1.2/test'
make[1]: Leaving directory '/usr/local/splint-3.1.2/test'
Making install in doc
make[1]: Entering directory '/usr/local/splint-3.1.2/doc'
make[2]: Entering directory '/usr/local/splint-3.1.2/doc'
make[2]: Nothing to be done for 'install-exec-am'.
/bin/bash ../config/mkinstalldirs /usr/local/splint/man/man1
mkdir /usr/local/splint/man
mkdir /usr/local/splint/man/man1
 /usr/bin/install -c -m 644 ./splint.1 /usr/local/splint/man/man1/splint.1
make[2]: Leaving directory '/usr/local/splint-3.1.2/doc'
make[1]: Leaving directory '/usr/local/splint-3.1.2/doc'
make[1]: Entering directory '/usr/local/splint-3.1.2'
make[2]: Entering directory '/usr/local/splint-3.1.2'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/usr/local/splint-3.1.2'
make[1]: Leaving directory '/usr/local/splint-3.1.2'
why@why:/usr/local/splint-3.1.2$
```

Step 2: 配置环境变量：

在~/.bashrc文件末尾插入如下代码即可：

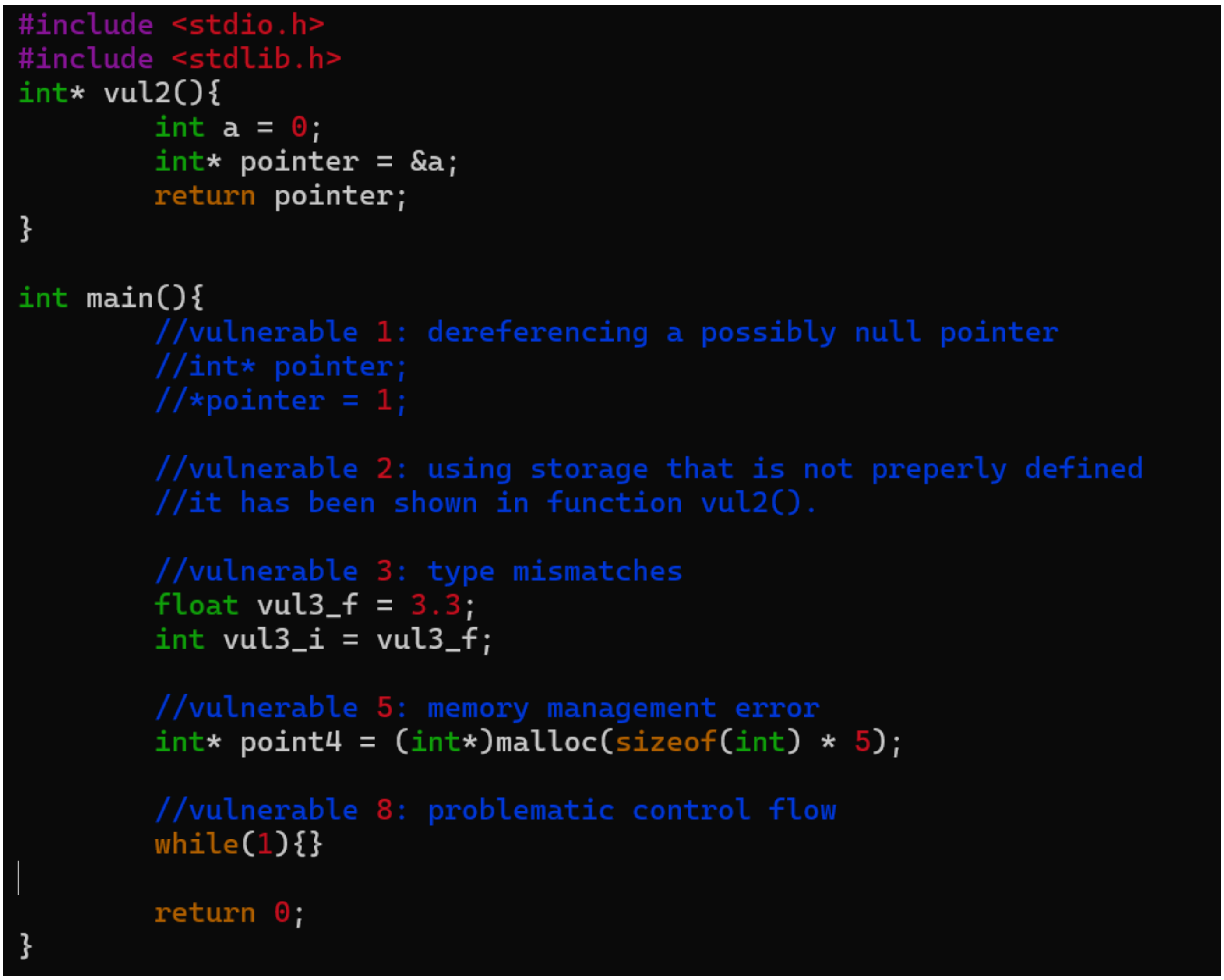


```
# Add an "alert" alias for long running commands.  Use like so:
#   sleep 10; alert
alias alert='notify-send --urgency=low -i "$([ $? = 0 ] && echo ter
s/^\s*[0-9]\+\s*//;s/[;&|]\s*alert$//'\''')"'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profil
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export LARCH_PATH=/usr/local/splint/shar/splint/lib
export LCLIMPORTDIR=/usr/splint/share/splint/imports
export PATH=$PATH:/usr/local/splint/bin
"~/.bashrc" 121L, 3917B
```

执行命令`source ~/.bashrc`以应用。

Step 3: 编写可能出错的C语言文件

这里给出了11条示例错误中的5条错误（由于在splint检测中，vulnerable 1对应错误会影响到splint对应的parser，因此在图片中被注释掉）：

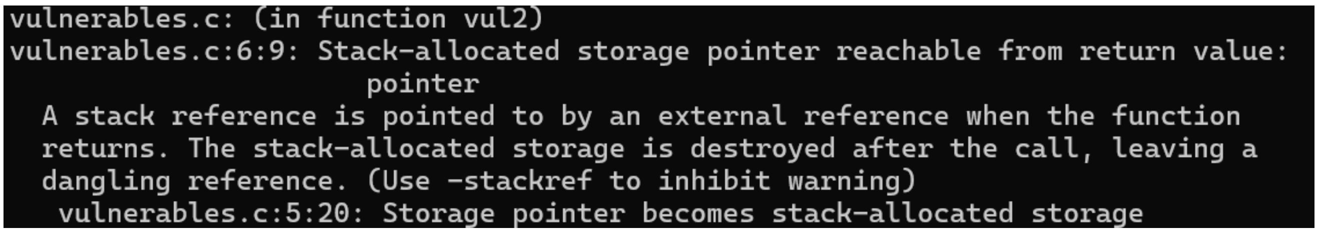


```c
#include <stdio.h>
#include <stdlib.h>
int* vul2(){
        int a = 0;
        int* pointer = &a;
        return pointer;
}

int main(){
        //vulnerable 1: dereferencing a possibly null pointer
        //int* pointer;
        //*pointer = 1;

        //vulnerable 2: using storage that is not preperly defined
        //it has been shown in function vul2().

        //vulnerable 3: type mismatches
        float vul3_f = 3.3;
        int vul3_i = vul3_f;

        //vulnerable 5: memory management error
        int* point4 = (int*)malloc(sizeof(int) * 5);

        //vulnerable 8: problematic control flow
        while(1){}

        return 0;
}
```

Step 4: 利用splint检测代码漏洞

在运行`splint vulnerables.c`后出现一系列警报信息，这里将对其分类进行解释：

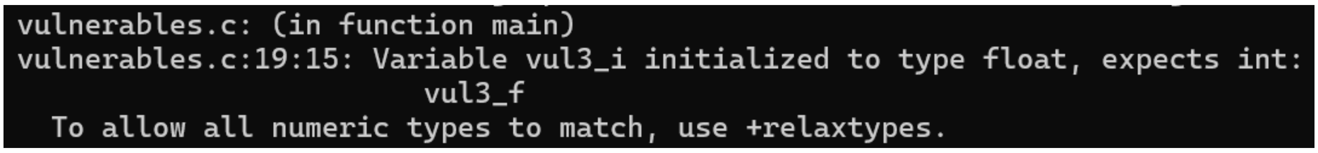
1. 漏洞2：



```
vulnerables.c: (in function vul2)
vulnerables.c:6:9: Stack-allocated storage pointer reachable from return value:
                      pointer
  A stack reference is pointed to by an external reference when the function
  returns. The stack-allocated storage is destroyed after the call, leaving a
  dangling reference. (Use -stackref to inhibit warning)
   vulnerables.c:5:20: Storage pointer becomes stack-allocated storage
```

这是在函数中的警报，原因是在函数栈中分配空间的局部变量在函数执行完毕后空间会被释放，而此时如果返回指向这个局部变量的指针是危险的。

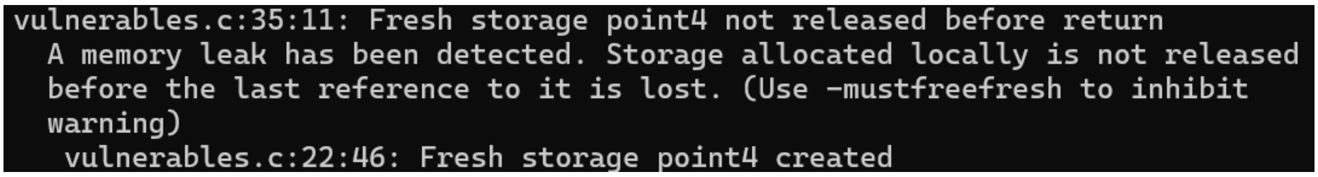
2. 漏洞3：



```
vulnerables.c: (in function main)
vulnerables.c:19:15: Variable vul3_i initialized to type float, expects int:
                        vul3_f
  To allow all numeric types to match, use +relaxtypes.
```

该警报是因为我们将float类型的变量赋值给了int，因此出现类型错误（但相对不严重，splint也提示表示可以通过`+relaxtypes`忽略这类警报。

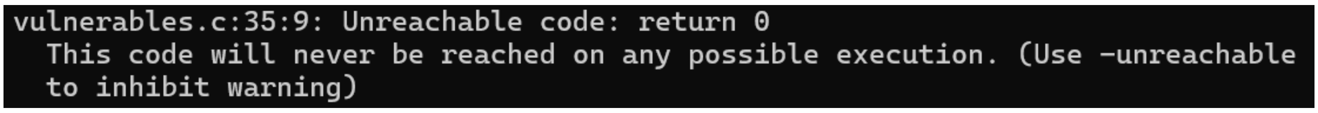
3. 漏洞5：



```
vulnerables.c:35:11: Fresh storage point4 not released before return
  A memory leak has been detected. Storage allocated locally is not released
  before the last reference to it is lost. (Use -mustfreefresh to inhibit
  warning)
   vulnerables.c:22:46: Fresh storage point4 created
```

该警报是因为我们在动态内存分配后到程序末尾都没有释放空间，因此造成了内存泄漏。

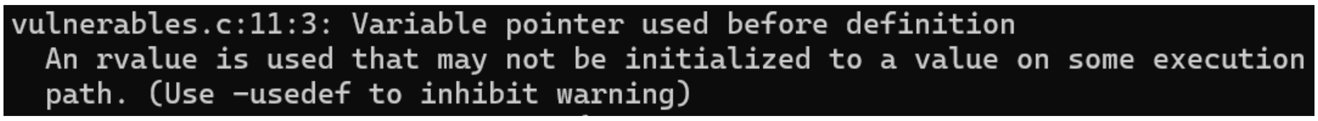
4. 漏洞8：



```
vulnerables.c:35:9: Unreachable code: return 0
  This code will never be reached on any possible execution. (Use -unreachable
  to inhibit warning)
```

该警报是因为`return 0`将永远不会被执行。由于控制流警报并没有找到splint发出的对应警报，因此用这种方式`unreachable code`来代替。由于该代码之前有死循环，因此函数永远不会终止。

5. 漏洞1：（去掉注释代码后可见）



```
vulnerables.c:11:3: Variable pointer used before definition
  An rvalue is used that may not be initialized to a value on some execution
  path. (Use -usedef to inhibit warning)
```

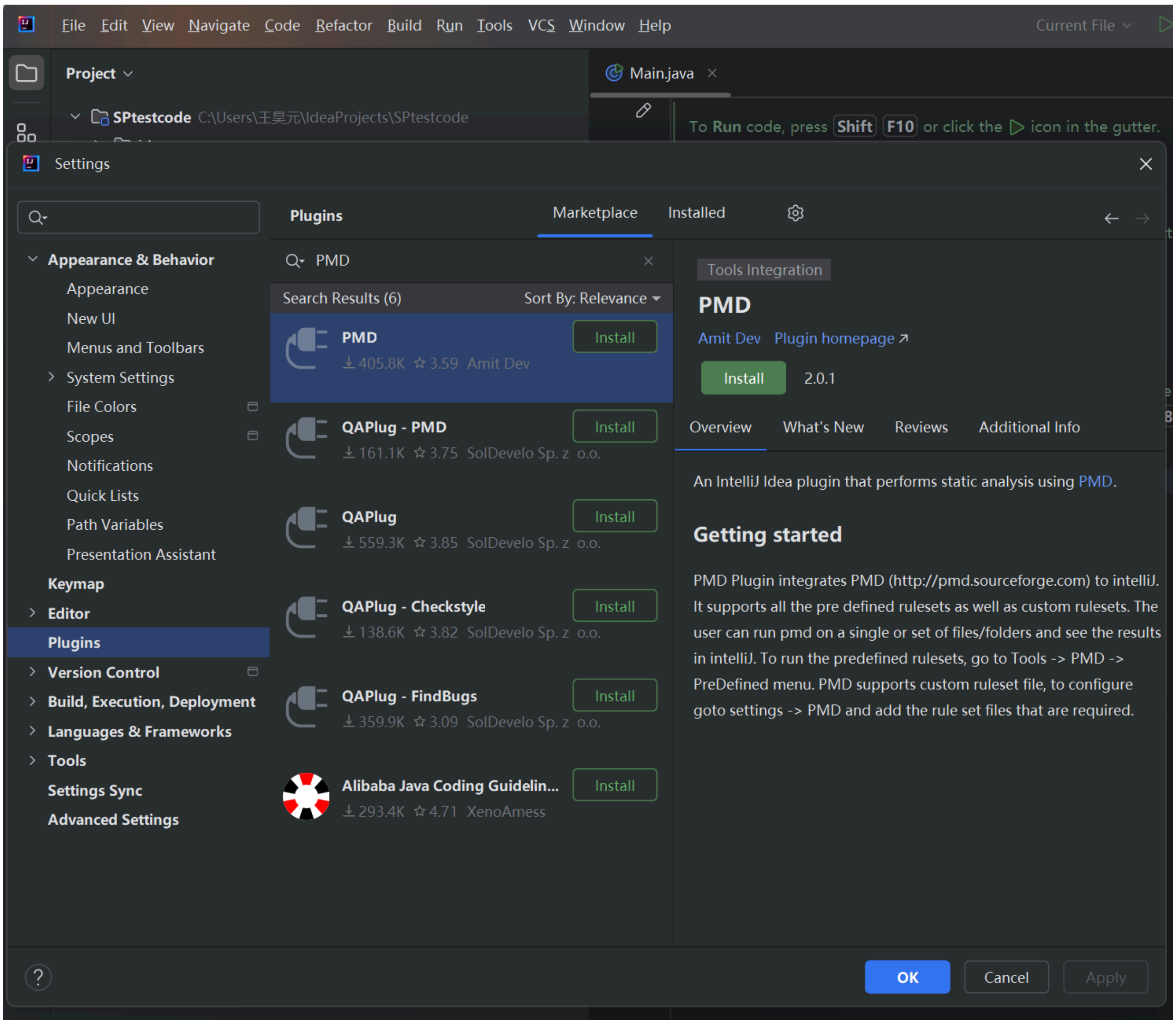
该警报是因为我们给一个未被定义的指针所指向的地址赋值。这会导致不可知的错误。

Lab 3.2 Using eclipse for java static analysis

由于本实验中在尝试实验报告给出的三个插件时，一个（FindBugs）检测无任何反应，其余两个链接直接失效，因此本实验采用IDEA平台提供的PMD接口进行PMD插件的下载，并直接在IDEA中完成本实验。如若希望看到在Eclipse的尝试可以移步Appendix。

Step 1: 在IDEA中下载PMD插件

在plugins中搜索`PMD`，即可找到对应的插件，下载之。



Plugin	Downloads	Rating	Vendor	
PMD	405.8K	3.59	Amit Dev	Install
QAPlug - PMD	161.1K	3.75	SolDevelo Sp. z o.o.	Install
QAPlug	559.3K	3.85	SolDevelo Sp. z o.o.	Install
QAPlug - Checkstyle	138.6K	3.82	SolDevelo Sp. z o.o.	Install
QAPlug - FindBugs	359.9K	3.09	SolDevelo Sp. z o.o.	Install
Alibaba Java Coding Guidelin...	293.4K	4.71	XenoAmess	Install

Tools Integration

PMD

Amit Dev Plugin homepage ↗

Install 2.0.1

Overview What's New Reviews Additional Info

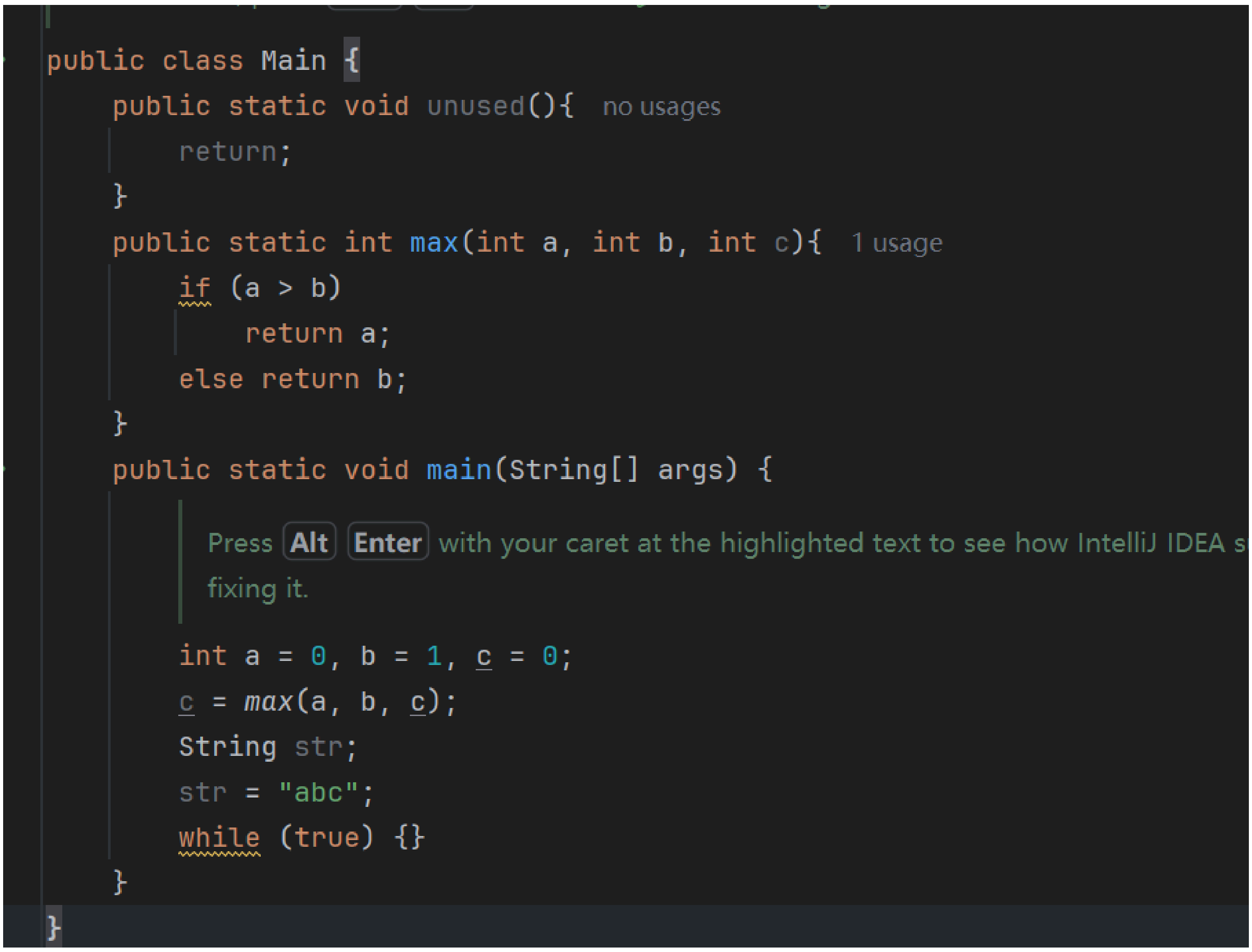
An IntelliJ Idea plugin that performs static analysis using PMD.

Getting started

PMD Plugin integrates PMD (http://pmd.sourceforge.com) to intelliJ. It supports all the pre defined rulesets as well as custom rulesets. The user can run pmd on a single or set of files/folders and see the results in intelliJ. To run the predefined rulesets, go to Tools -> PMD -> PreDefined menu. PMD supports custom ruleset file, to configure goto settings -> PMD and add the rule set files that are required.

Step 2: 编写一段有待优化的代码

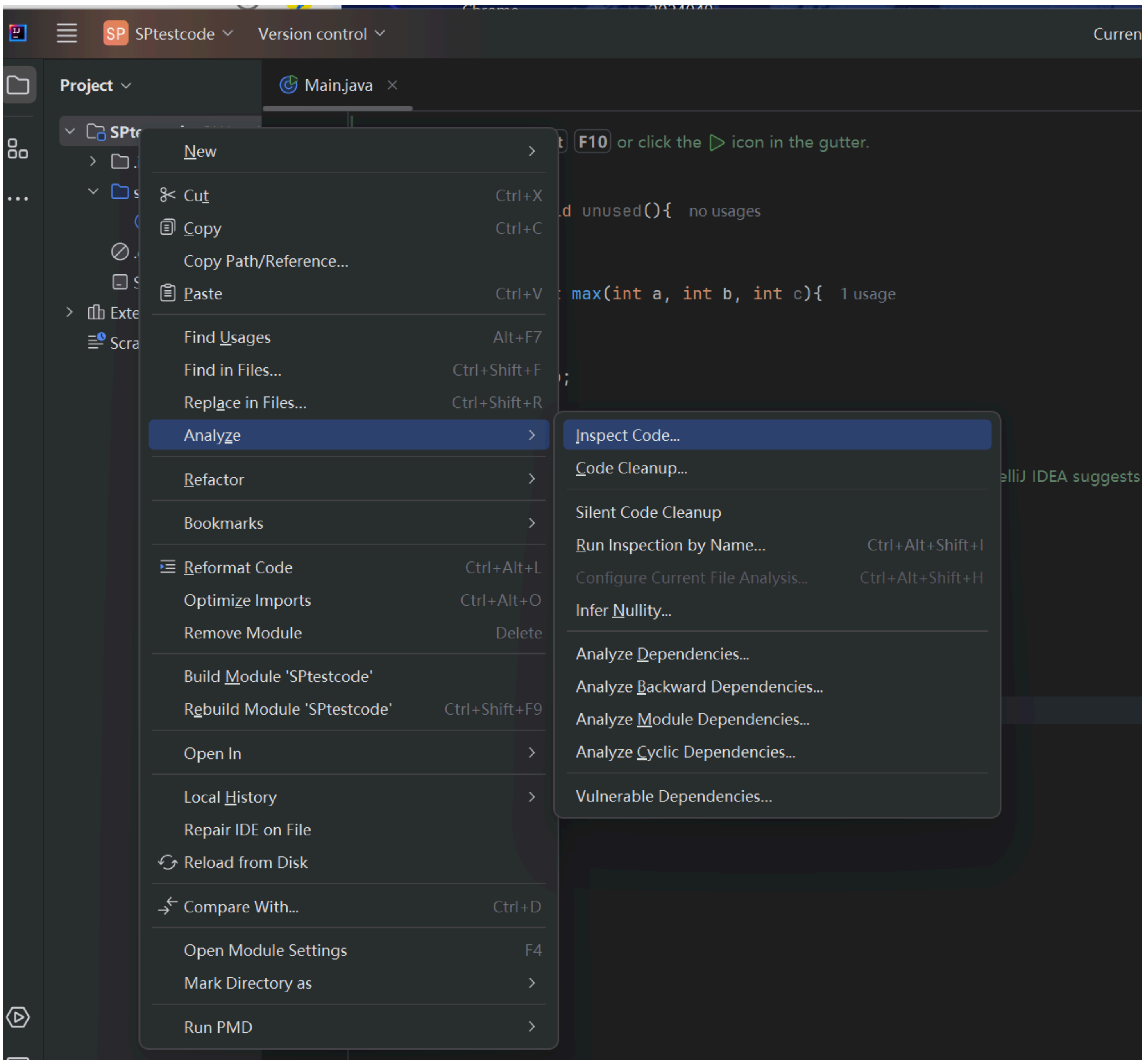
如下图所示，编写了一段尚待优化的代码，其中包括死循环问题、参量未使用、变量未使用、方法未使用等问题：



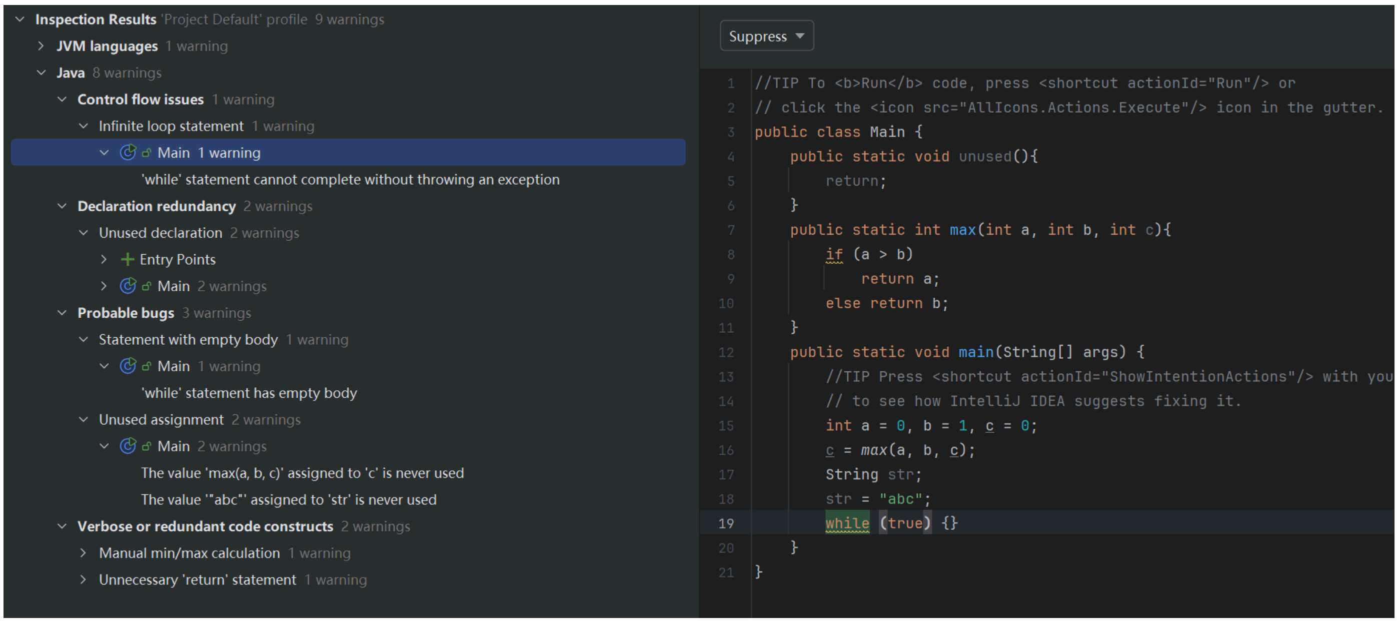
```
public class Main {
    public static void unused(){  no usages
        return;
    }
    public static int max(int a, int b, int c){  1 usage
        if (a > b)
            return a;
        else return b;
    }
    public static void main(String[] args) {
        Press Alt Enter with your caret at the highlighted text to see how IntelliJ IDEA s
        fixing it.
        int a = 0, b = 1, c = 0;
        c = max(a, b, c);
        String str;
        str = "abc";
        while (true) {}
    }
}
```

Step 3: 使用PMD对其进行检测

右键项目，通过`analyze->inspect code`进行查验：



于是可以看到所有的warning，并且点击其中一个可以看到右侧甚至给出了对应截取的段落：



```
Inspection Results 'Project Default' profile 9 warnings
  JVM languages 1 warning
  Java 8 warnings
    Control flow issues 1 warning
      Infinite loop statement 1 warning
        Main 1 warning
          'while' statement cannot complete without throwing an exception
    Declaration redundancy 2 warnings
      Unused declaration 2 warnings
        Entry Points
        Main 2 warnings
    Probable bugs 3 warnings
      Statement with empty body 1 warning
        Main 1 warning
          'while' statement has empty body
      Unused assignment 2 warnings
        Main 2 warnings
          The value 'max(a, b, c)' assigned to 'c' is never used
          The value '"abc"' assigned to 'str' is never used
    Verbose or redundant code constructs 2 warnings
      Manual min/max calculation 1 warning
      Unnecessary 'return' statement 1 warning
```

```java
//TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or
// click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.
public class Main {
    public static void unused(){
        return;
    }
    public static int max(int a, int b, int c){
        if (a > b)
            return a;
        else return b;
    }
    public static void main(String[] args) {
        //TIP Press <shortcut actionId="ShowIntentionActions"/> with you
        // to see how IntelliJ IDEA suggests fixing it.
        int a = 0, b = 1, c = 0;
        c = max(a, b, c);
        String str;
        str = "abc";
        while (true) {}
    }
}
```

接下来将逐一解释所有的warning：

- Contorl flow issues: 控制流错误，具体为while()为死循环
- Declaration redundancy: 声明冗余，具体为未被调用的方法与变量
- Probable bugs: 可能的错误，具体为while()的执行语句为空以及未使用变量/参量，检测软件判断可能为程序员漏写或错写，因此可能导致错误出现
- Verbose or redundant code constructs: 冗余的代码结构，具体为max()方法在Java的Math库中已经给出，因此不必再手动设置一个max()方法；以及在返回值为void函数中，没有必要单独写出return语句。

小结

这两个实验主要是让我们熟悉C语言代码以及Java语言代码检测工具的使用，进一步提示我们不仅要注意error，还要注意warning.在编写代码时，应当尽可能将所有warning消除，养成好的编程习惯。

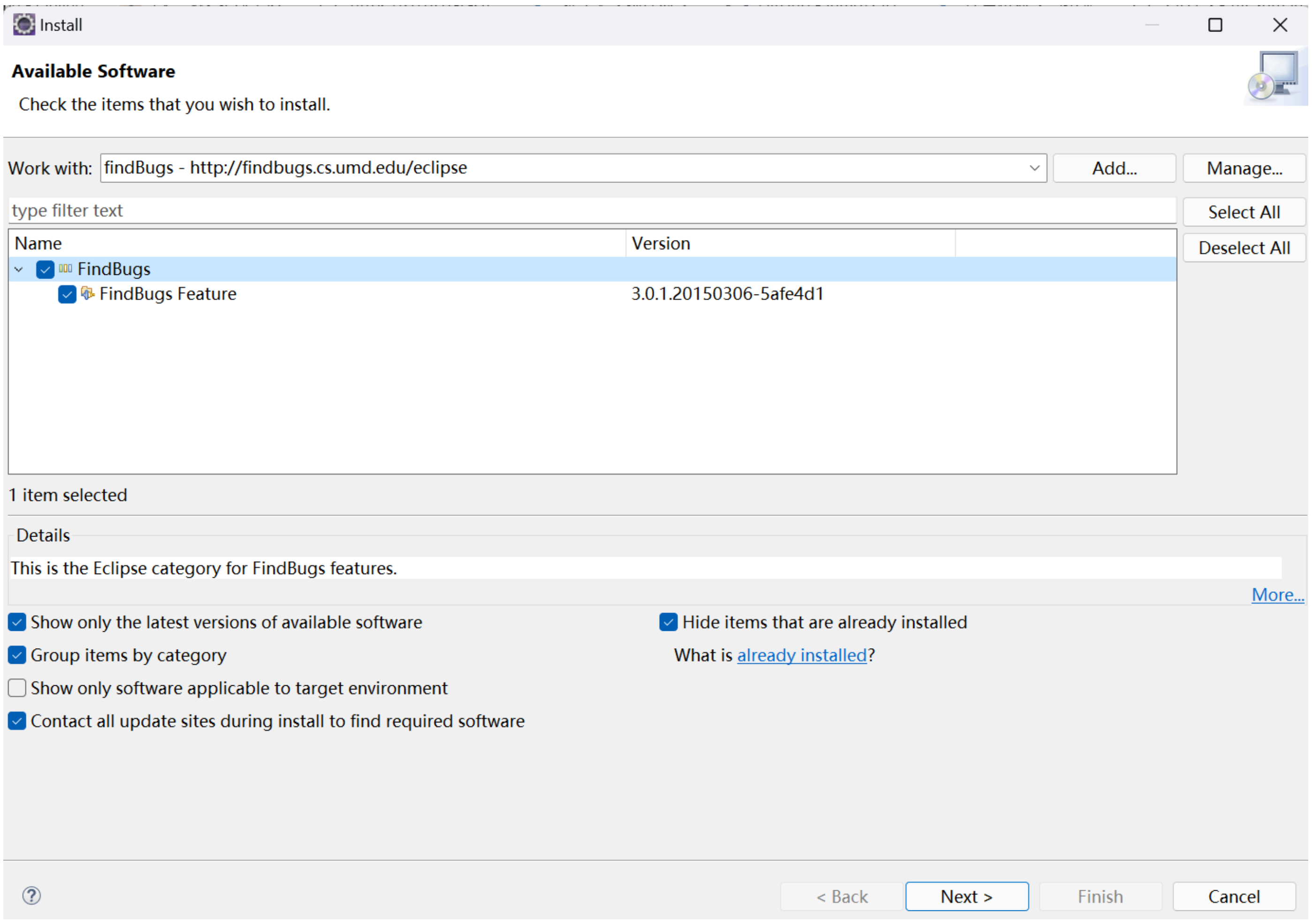
Appendix：failure attempts

Step 1: Eclipse安装

本实验似乎并不严格要求在linux平台下进行，而在Lab1.1中已经在Windows系统中安装了Eclipse,因此这里只粘贴Eclipse成功运行的窗口：

Step 2: 选择合适的Java代码检测软件，并编写代码

本实验选择的是findBug，在Install New Software中可以看到对应的软件：



Install

Available Software

Check the items that you wish to install.

Work with: findBugs - http://findbugs.cs.umd.edu/eclipse

Name	Version
☑ FindBugs	
☑ FindBugs Feature	3.0.1.20150306-5afe4d1

1 item selected

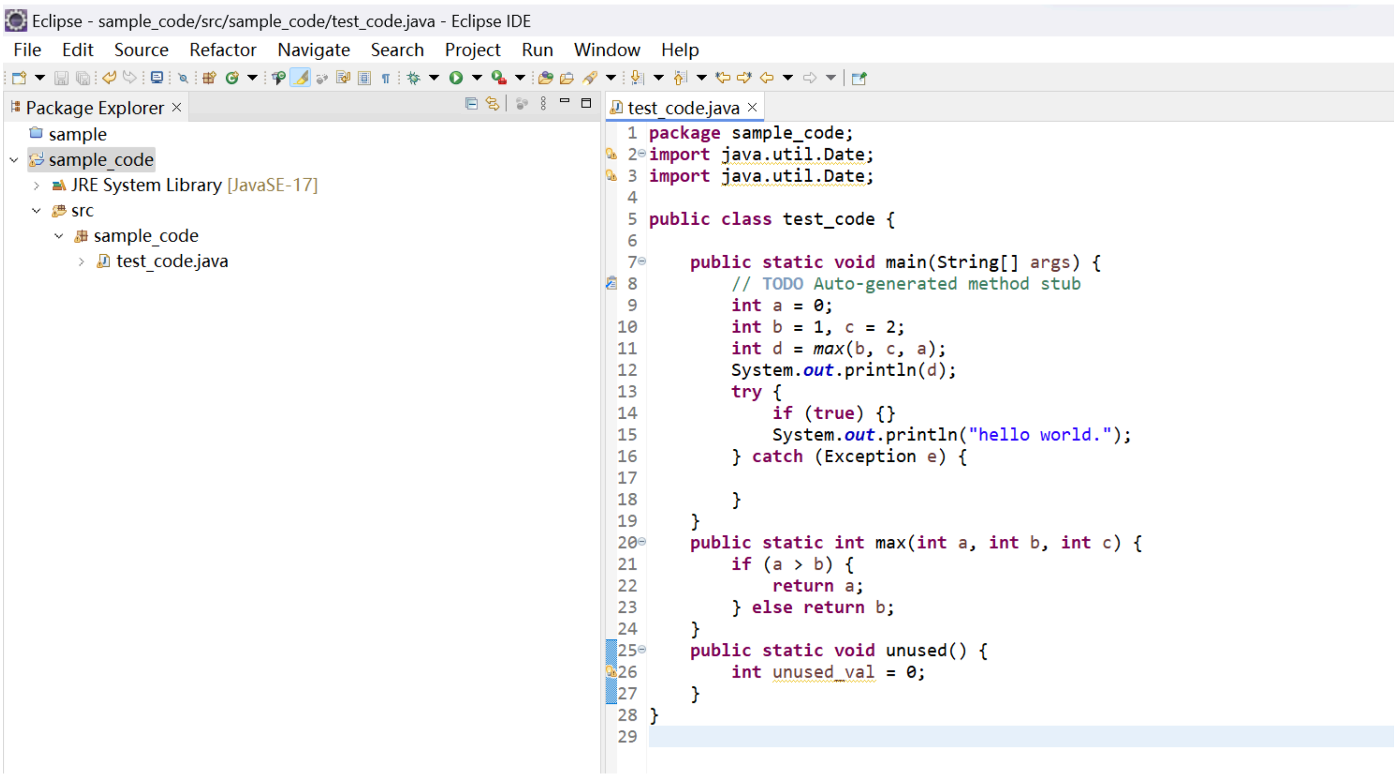
Details

This is the Eclipse category for FindBugs features.

☑ Show only the latest versions of available software
☑ Group items by category
☐ Show only software applicable to target environment
☑ Contact all update sites during install to find required software
☑ Hide items that are already installed

What is already installed?

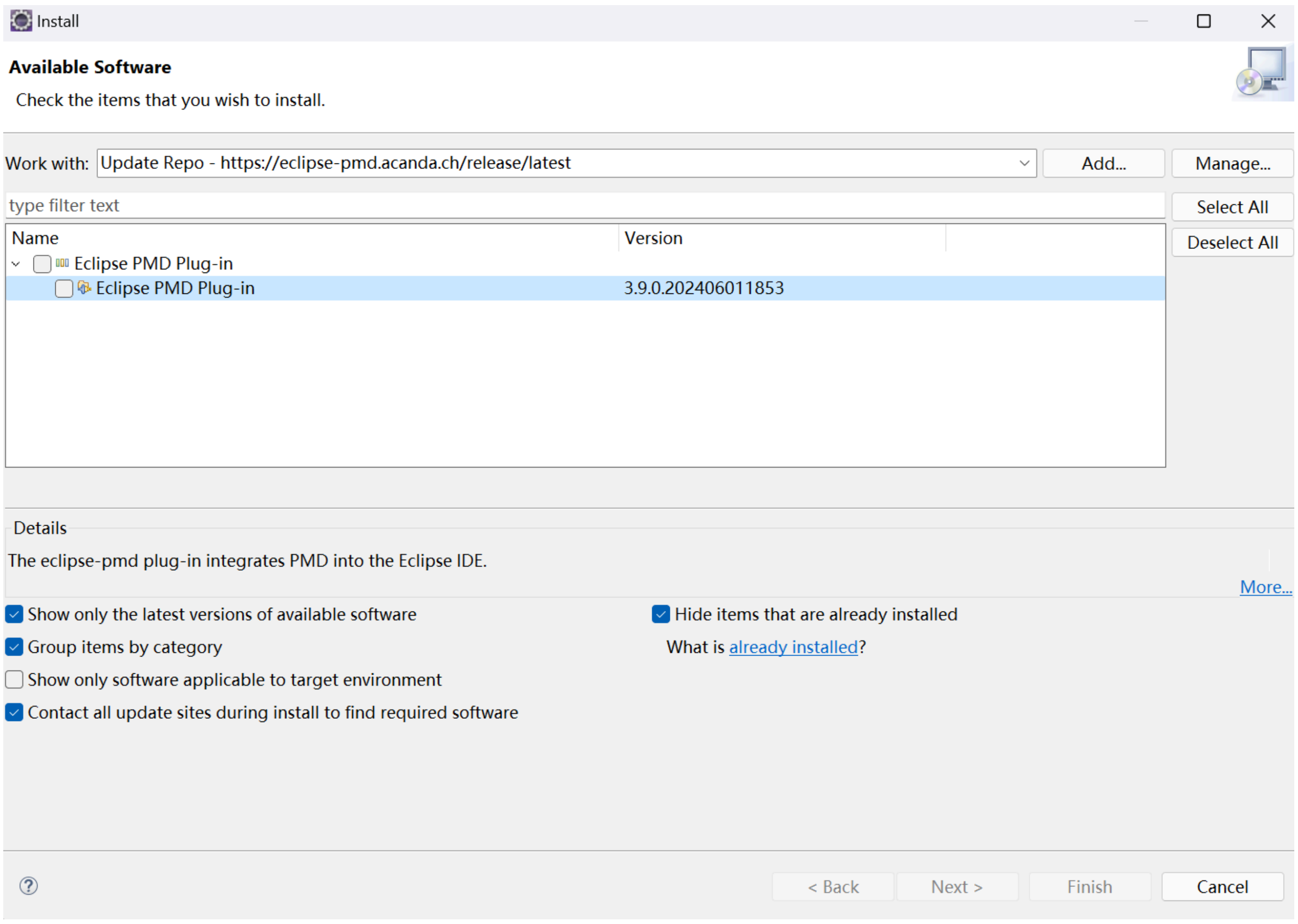
之后使用Lab1.1的方式创建一个简单的Java文件，并编写一份有各种问题的代码：



Eclipse - sample_code/src/sample_code/test_code.java - Eclipse IDE

```java
package sample_code;
import java.util.Date;
import java.util.Date;

public class test_code {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int a = 0;
        int b = 1, c = 2;
        int d = max(b, c, a);
        System.out.println(d);
        try {
            if (true) {}
            System.out.println("hello world.");
        } catch (Exception e) {

        }
    }
    public static int max(int a, int b, int c) {
        if (a > b) {
            return a;
        } else return b;
    }
    public static void unused() {
        int unused_val = 0;
    }
}
```

安装eclipse-pmd：


Install

Available Software

Check the items that you wish to install.

Work with: Update Repo - https://eclipse-pmd.acanda.ch/release/latest

Name	Version
☐ Eclipse PMD Plug-in	
☐ Eclipse PMD Plug-in	3.9.0.202406011853

Details

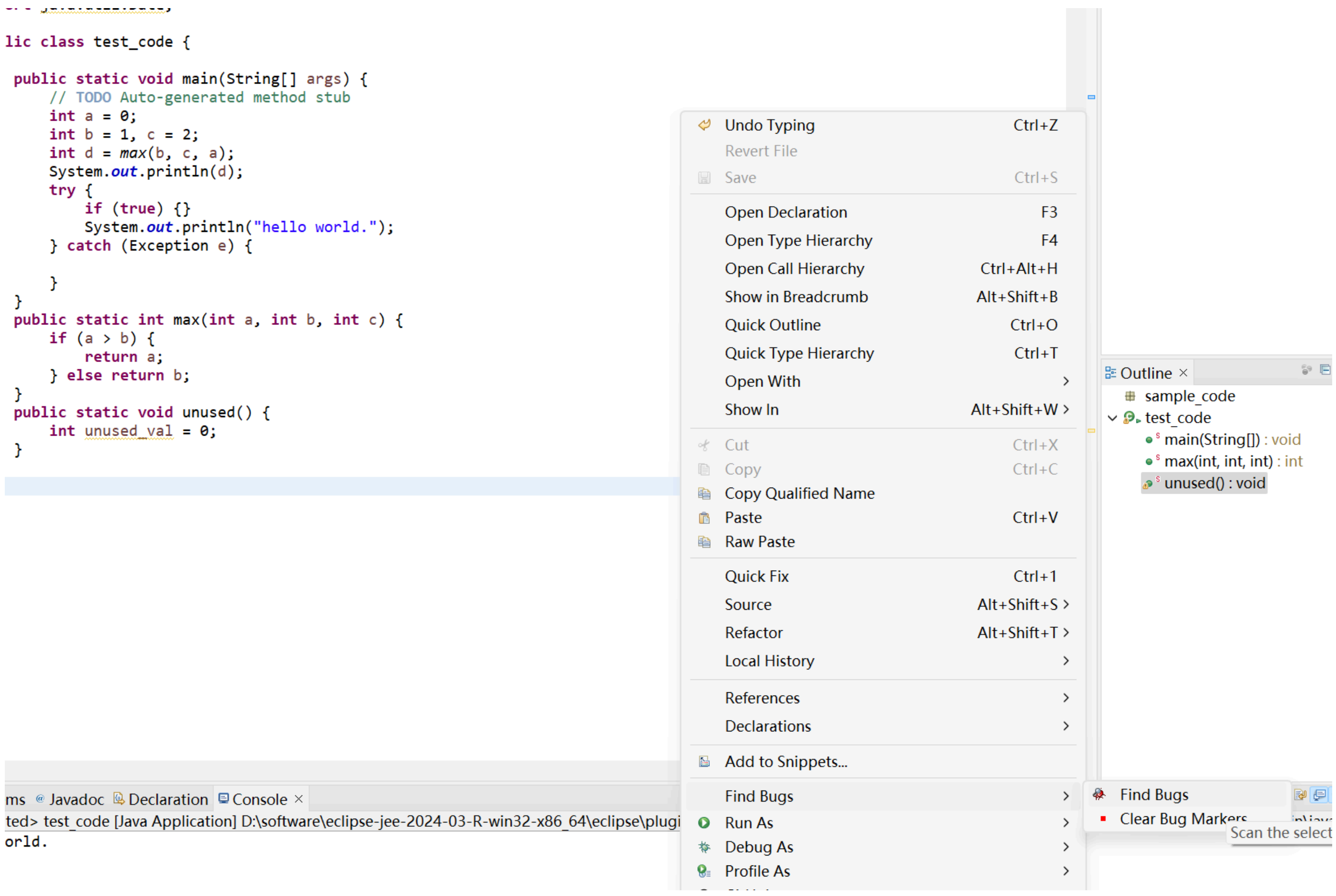
The eclipse-pmd plug-in integrates PMD into the Eclipse IDE.

- [✓] Show only the latest versions of available software
- [✓] Group items by category
- [ ] Show only software applicable to target environment
- [✓] Contact all update sites during install to find required software
- [✓] Hide items that are already installed

What is already installed?

Step 3: 对代码进行检测

右键后找到对应安装的代码检测软件：



```
lic class test_code {

 public static void main(String[] args) {
     // TODO Auto-generated method stub
     int a = 0;
     int b = 1, c = 2;
     int d = max(b, c, a);
     System.out.println(d);
     try {
         if (true) {}
         System.out.println("hello world.");
     } catch (Exception e) {

     }
 }
 public static int max(int a, int b, int c) {
     if (a > b) {
         return a;
     } else return b;
 }
 public static void unused() {
     int unused_val = 0;
 }
```

```
ms  Javadoc  Declaration  Console ×
ted> test_code [Java Application] D:\software\eclipse-jee-2024-03-R-win32-x86_64\eclipse\plugi
orld.
```

运行后发现BugExplorer中无任何信息，且日志中无任何信息。在设置preference检测力度、property的相关设置之后仍然无效，查找众多博客后依旧无法生效，因此更换为其他平台。