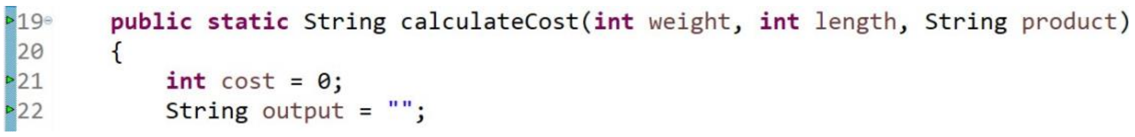
Project
Static Analysis

1) Description of the two Data flow anomalies

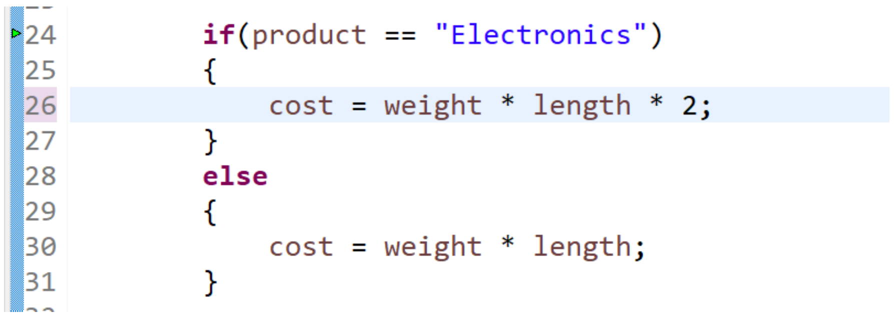
- The tool I have used for static code analysis is **pmd-eclipse-plugin 4.40.0**

- The code contains two Data flow anomalies on lines **21** and **22** respectively.



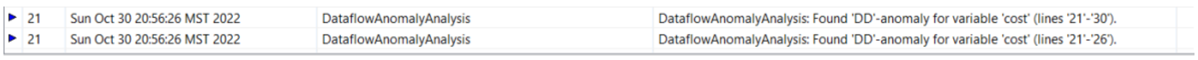
```
19    public static String calculateCost(int weight, int length, String product)
20    {
21        int cost = 0;
22        String output = "";
```

- The **first anomaly** found is for the variable ‘**cost**’ that is afterward used in the ‘if’ (line 26) and ‘else’ conditions respectively (line 30).



```
24        if(product == "Electronics")
25        {
26            cost = weight * length * 2;
27        }
28        else
29        {
30            cost = weight * length;
31        }
```

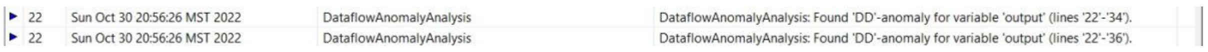
- The tool analysis provides the anomaly and categorizes it as a ‘**DD**’ anomaly.



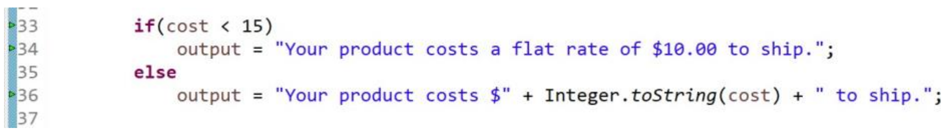
```
▶ 21   Sun Oct 30 20:56:26 MST 2022   DataflowAnomalyAnalysis   DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'cost' (lines '21'-'30').
▶ 21   Sun Oct 30 20:56:26 MST 2022   DataflowAnomalyAnalysis   DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'cost' (lines '21'-'26').
```

- DD means ‘**defining the data objects twice**’. It is categorized as a type of **harmless**, **but suspicious** anomaly [4].

- The **second anomaly** is also of type ‘**DD**’ which is for the variable ‘**output**’ that is afterward used in the ‘if’ and ‘else’ conditions below the ‘cost < 15’ statement.



```
22   Sun Oct 30 20:56:26 MST 2022   DataflowAnomalyAnalysis   DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'output' (lines '22'-'34').
22   Sun Oct 30 20:56:26 MST 2022   DataflowAnomalyAnalysis   DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'output' (lines '22'-'36').
```

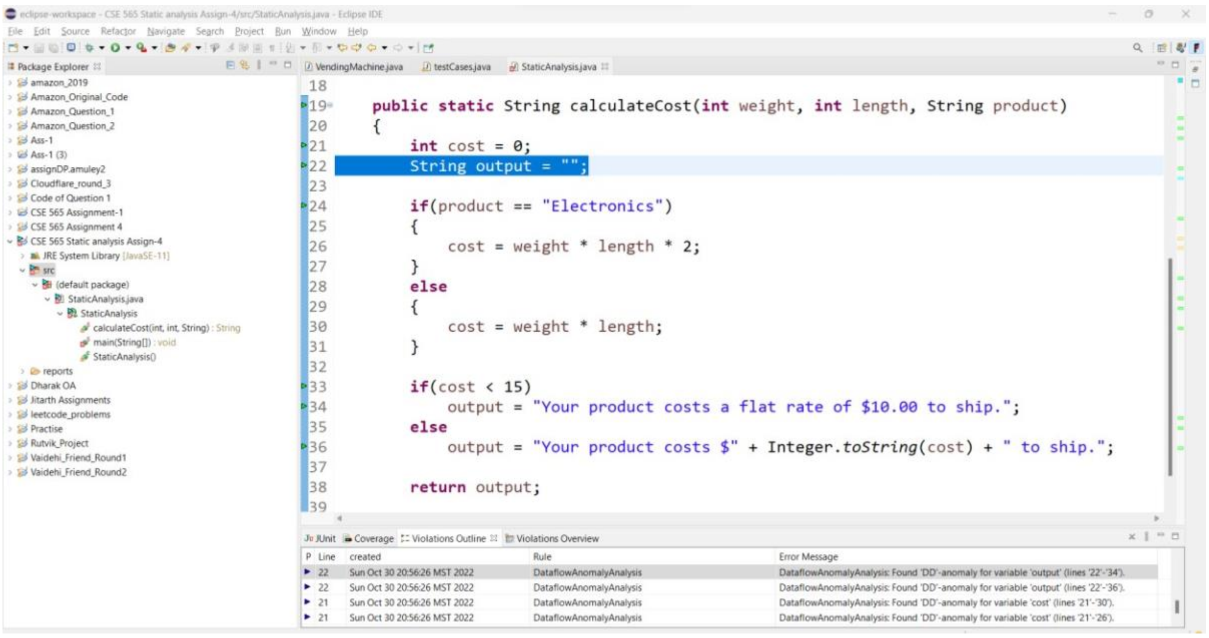


```
33        if(cost < 15)
34            output = "Your product costs a flat rate of $10.00 to ship.";
35        else
36            output = "Your product costs $" + Integer.toString(cost) + " to ship.";
37
```

- DD arises when a recently defined variable is defined again.

2) Screenshot showing the analysis is performed:

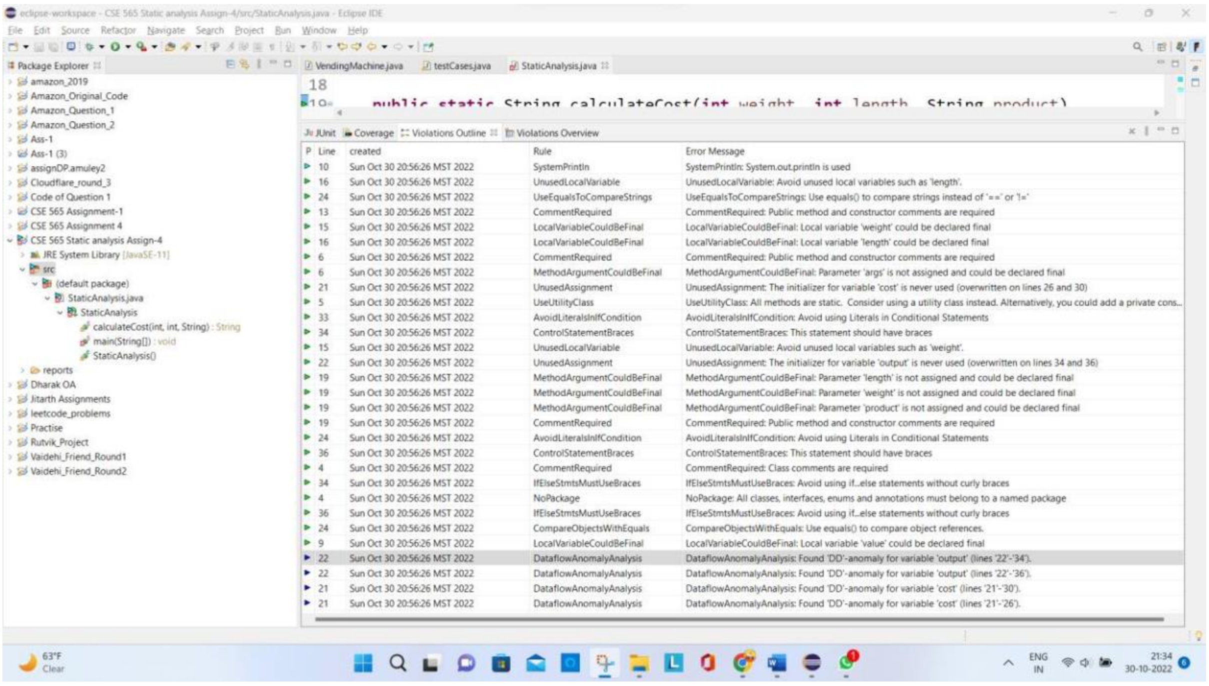
⇨ DataflowAnomaly Analysis:



```java

    public static String calculateCost(int weight, int length, String product)
    {
        int cost = 0;
        String output = "";

        if(product == "Electronics")
        {
            cost = weight * length * 2;
        }
        else
        {
            cost = weight * length;
        }

        if(cost < 15)
            output = "Your product costs a flat rate of $10.00 to ship.";
        else
            output = "Your product costs $" + Integer.toString(cost) + " to ship.";

        return output;

```

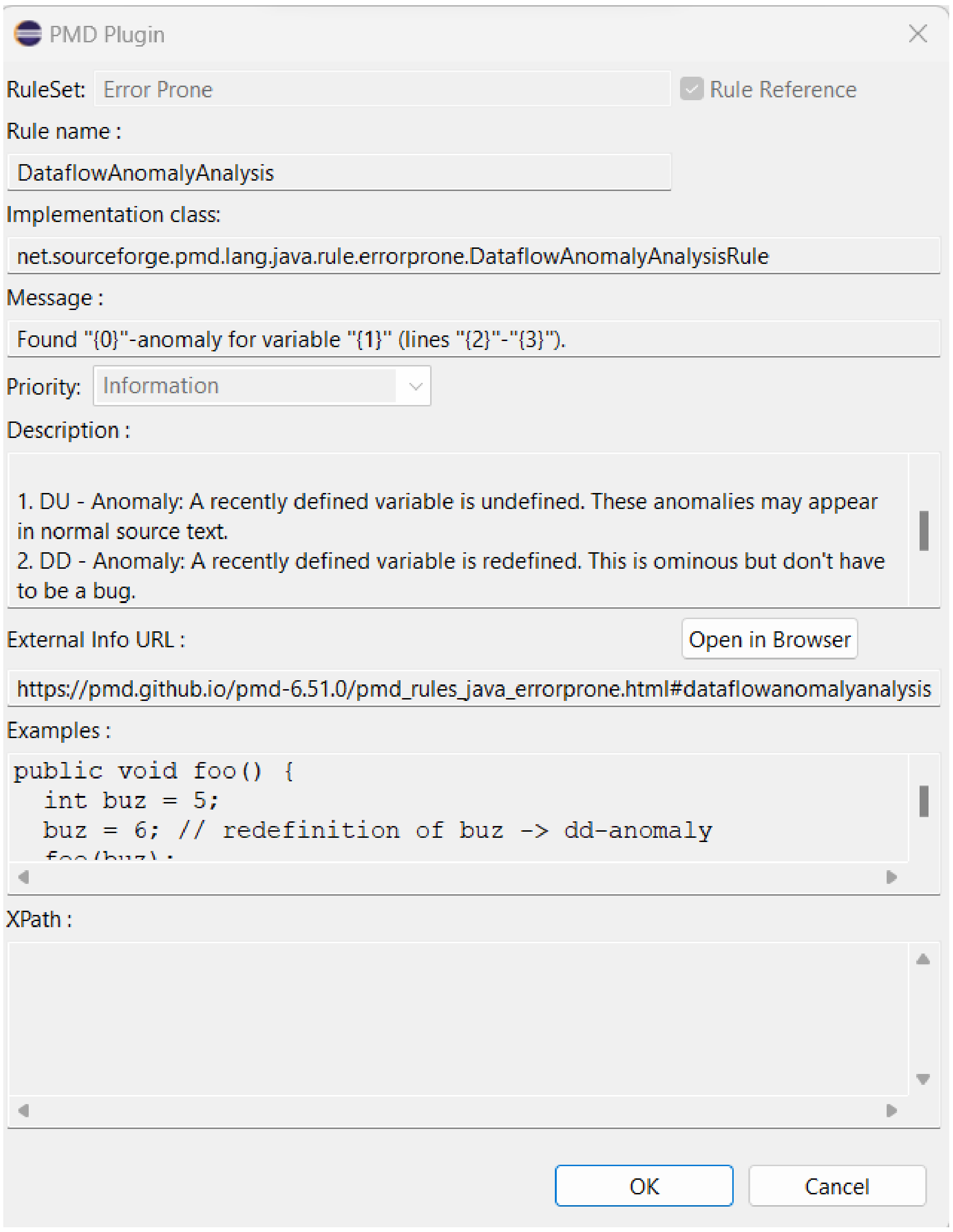
P	Line	created	Rule	Error Message
▶	22	Sun Oct 30 20:56:26 MST 2022	DataflowAnomalyAnalysis	DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'output' (lines '22'-'34').
▶	22	Sun Oct 30 20:56:26 MST 2022	DataflowAnomalyAnalysis	DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'output' (lines '22'-'36').
▶	21	Sun Oct 30 20:56:26 MST 2022	DataflowAnomalyAnalysis	DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'cost' (lines '21'-'30').
▶	21	Sun Oct 30 20:56:26 MST 2022	DataflowAnomalyAnalysis	DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'cost' (lines '21'-'26').

⇨ Complete analysis:



P	Line	created	Rule	Error Message
▶	10	Sun Oct 30 20:56:26 MST 2022	SystemPrintln	SystemPrintln: System.out.println is used
▶	16	Sun Oct 30 20:56:26 MST 2022	UnusedLocalVariable	UnusedLocalVariable: Avoid unused local variables such as 'length'.
▶	24	Sun Oct 30 20:56:26 MST 2022	UseEqualsToCompareStrings	UseEqualsToCompareStrings: Use equals() to compare strings instead of '==' or '!='
▶	13	Sun Oct 30 20:56:26 MST 2022	CommentRequired	CommentRequired: Public method and constructor comments are required
▶	15	Sun Oct 30 20:56:26 MST 2022	LocalVariableCouldBeFinal	LocalVariableCouldBeFinal: Local variable 'weight' could be declared final
▶	16	Sun Oct 30 20:56:26 MST 2022	LocalVariableCouldBeFinal	LocalVariableCouldBeFinal: Local variable 'length' could be declared final
▶	6	Sun Oct 30 20:56:26 MST 2022	CommentRequired	CommentRequired: Public method and constructor comments are required
▶	6	Sun Oct 30 20:56:26 MST 2022	MethodArgumentCouldBeFinal	MethodArgumentCouldBeFinal: Parameter 'args' is not assigned and could be declared final
▶	21	Sun Oct 30 20:56:26 MST 2022	UnusedAssignment	UnusedAssignment: The initializer for variable 'cost' is never used (overwritten on lines 26 and 30)
▶	5	Sun Oct 30 20:56:26 MST 2022	UseUtilityClass	UseUtilityClass: All methods are static. Consider using a utility class instead. Alternatively, you could add a private cons...
▶	33	Sun Oct 30 20:56:26 MST 2022	AvoidLiteralsInIfCondition	AvoidLiteralsInIfCondition: Avoid using Literals in Conditional Statements
▶	34	Sun Oct 30 20:56:26 MST 2022	ControlStatementBraces	ControlStatementBraces: This statement should have braces
▶	15	Sun Oct 30 20:56:26 MST 2022	UnusedLocalVariable	UnusedLocalVariable: Avoid unused local variables such as 'weight'.
▶	22	Sun Oct 30 20:56:26 MST 2022	UnusedAssignment	UnusedAssignment: The initializer for variable 'output' is never used (overwritten on lines 34 and 36)
▶	19	Sun Oct 30 20:56:26 MST 2022	MethodArgumentCouldBeFinal	MethodArgumentCouldBeFinal: Parameter 'length' is not assigned and could be declared final
▶	19	Sun Oct 30 20:56:26 MST 2022	MethodArgumentCouldBeFinal	MethodArgumentCouldBeFinal: Parameter 'weight' is not assigned and could be declared final
▶	19	Sun Oct 30 20:56:26 MST 2022	MethodArgumentCouldBeFinal	MethodArgumentCouldBeFinal: Parameter 'product' is not assigned and could be declared final
▶	19	Sun Oct 30 20:56:26 MST 2022	CommentRequired	CommentRequired: Public method and constructor comments are required
▶	24	Sun Oct 30 20:56:26 MST 2022	AvoidLiteralsInIfCondition	AvoidLiteralsInIfCondition: Avoid using Literals in Conditional Statements
▶	36	Sun Oct 30 20:56:26 MST 2022	ControlStatementBraces	ControlStatementBraces: This statement should have braces
▶	4	Sun Oct 30 20:56:26 MST 2022	CommentRequired	CommentRequired: Class comments are required
▶	34	Sun Oct 30 20:56:26 MST 2022	IfElseStmtsMustUseBraces	IfElseStmtsMustUseBraces: Avoid using if...else statements without curly braces
▶	4	Sun Oct 30 20:56:26 MST 2022	NoPackage	NoPackage: All classes, interfaces, enums and annotations must belong to a named package
▶	36	Sun Oct 30 20:56:26 MST 2022	IfElseStmtsMustUseBraces	IfElseStmtsMustUseBraces: Avoid using if...else statements without curly braces
▶	24	Sun Oct 30 20:56:26 MST 2022	CompareObjectsWithEquals	CompareObjectsWithEquals: Use equals() to compare object references.
▶	9	Sun Oct 30 20:56:26 MST 2022	LocalVariableCouldBeFinal	LocalVariableCouldBeFinal: Local variable 'value' could be declared final
▶	22	Sun Oct 30 20:56:26 MST 2022	DataflowAnomalyAnalysis	DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'output' (lines '22'-'34').
▶	22	Sun Oct 30 20:56:26 MST 2022	DataflowAnomalyAnalysis	DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'output' (lines '22'-'36').
▶	21	Sun Oct 30 20:56:26 MST 2022	DataflowAnomalyAnalysis	DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'cost' (lines '21'-'30').
▶	21	Sun Oct 30 20:56:26 MST 2022	DataflowAnomalyAnalysis	DataflowAnomalyAnalysis: Found 'DD'-anomaly for variable 'cost' (lines '21'-'26').

⇨ Details of Anomaly type provided by tool:



RuleSet: Error Prone ✓ Rule Reference

Rule name : DataflowAnomalyAnalysis

Implementation class: net.sourceforge.pmd.lang.java.rule.errorprone.DataflowAnomalyAnalysisRule

Message : Found "{0}"-anomaly for variable "{1}" (lines "{2}"-"{3}").

Priority: Information

Description :

1. DU - Anomaly: A recently defined variable is undefined. These anomalies may appear in normal source text.
2. DD - Anomaly: A recently defined variable is redefined. This is ominous but don't have to be a bug.

External Info URL : https://pmd.github.io/pmd-6.51.0/pmd_rules_java_errorprone.html#dataflowanomalyanalysis

Examples :

```
public void foo() {
  int buz = 5;
  buz = 6; // redefinition of buz -> dd-anomaly
```

XPath :

3) Your evaluation of the tool's usefulness.

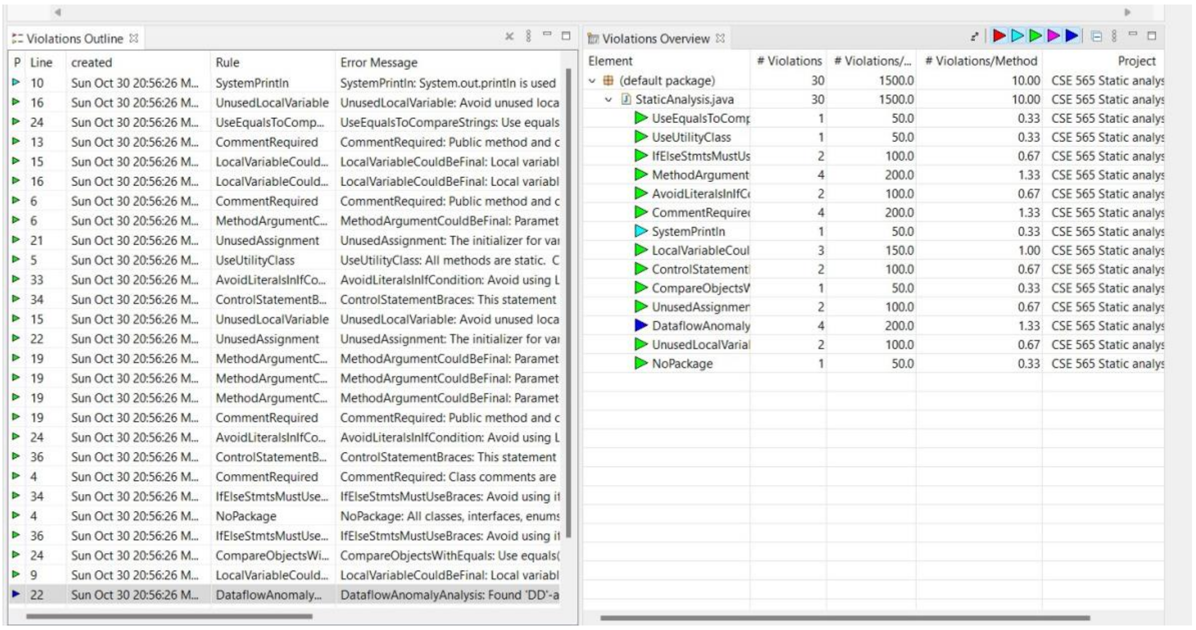
- pmd-eclipse plugin is a very useful plugin when it comes to code analysis. The tool is available on the **eclipse marketplace**.

- After clicking the install button, and restarting the IDE, we can directly right-click on the **project -> PMD -> Check code** to analyze the code.

- The tool provides the Violations overview, violations outline, and detailed descriptions.

- It categorizes violations in **5** types: Warning, Important, Urgent, critical, and blocker (from less severe to more severe)

- Under detailed outline, it shows the type of violation, line, the time when it was founded (in execution), Rule it violates, and Error message.



Violations Outline

P	Line	created	Rule	Error Message
▶	10	Sun Oct 30 20:56:26 M...	SystemPrintln	SystemPrintln: System.out.println is used
▶	16	Sun Oct 30 20:56:26 M...	UnusedLocalVariable	UnusedLocalVariable: Avoid unused loca
▶	24	Sun Oct 30 20:56:26 M...	UseEqualsToComp...	UseEqualsToCompareStrings: Use equals
▶	13	Sun Oct 30 20:56:26 M...	CommentRequired	CommentRequired: Public method and c
▶	15	Sun Oct 30 20:56:26 M...	LocalVariableCould...	LocalVariableCouldBeFinal: Local variabl
▶	16	Sun Oct 30 20:56:26 M...	LocalVariableCould...	LocalVariableCouldBeFinal: Local variabl
▶	6	Sun Oct 30 20:56:26 M...	CommentRequired	CommentRequired: Public method and c
▶	6	Sun Oct 30 20:56:26 M...	MethodArgumentC...	MethodArgumentCouldBeFinal: Paramet
▶	21	Sun Oct 30 20:56:26 M...	UnusedAssignment	UnusedAssignment: The initializer for vai
▶	5	Sun Oct 30 20:56:26 M...	UseUtilityClass	UseUtilityClass: All methods are static. C
▶	33	Sun Oct 30 20:56:26 M...	AvoidLiteralsInIfCo...	AvoidLiteralsInIfCondition: Avoid using L
▶	34	Sun Oct 30 20:56:26 M...	ControlStatementB...	ControlStatementBraces: This statement
▶	15	Sun Oct 30 20:56:26 M...	UnusedLocalVariable	UnusedLocalVariable: Avoid unused loca
▶	22	Sun Oct 30 20:56:26 M...	UnusedAssignment	UnusedAssignment: The initializer for vai
▶	19	Sun Oct 30 20:56:26 M...	MethodArgumentC...	MethodArgumentCouldBeFinal: Paramet
▶	19	Sun Oct 30 20:56:26 M...	MethodArgumentC...	MethodArgumentCouldBeFinal: Paramet
▶	19	Sun Oct 30 20:56:26 M...	MethodArgumentC...	MethodArgumentCouldBeFinal: Paramet
▶	19	Sun Oct 30 20:56:26 M...	CommentRequired	CommentRequired: Public method and c
▶	24	Sun Oct 30 20:56:26 M...	AvoidLiteralsInIfCo...	AvoidLiteralsInIfCondition: Avoid using L
▶	36	Sun Oct 30 20:56:26 M...	ControlStatementB...	ControlStatementBraces: This statement
▶	4	Sun Oct 30 20:56:26 M...	CommentRequired	CommentRequired: Class comments are
▶	34	Sun Oct 30 20:56:26 M...	IfElseStmtsMustUse...	IfElseStmtsMustUseBraces: Avoid using if
▶	4	Sun Oct 30 20:56:26 M...	NoPackage	NoPackage: All classes, interfaces, enums
▶	36	Sun Oct 30 20:56:26 M...	IfElseStmtsMustUse...	IfElseStmtsMustUseBraces: Avoid using if
▶	24	Sun Oct 30 20:56:26 M...	CompareObjectsWi...	CompareObjectsWithEquals: Use equals(
▶	9	Sun Oct 30 20:56:26 M...	LocalVariableCould...	LocalVariableCouldBeFinal: Local variabl
▶	22	Sun Oct 30 20:56:26 M...	DataflowAnomaly...	DataflowAnomalyAnalysis: Found 'DD'-a

Violations Overview

Element	# Violations	# Violations/...	# Violations/Method	Project
(default package)	30	1500.0	10.00	CSE 565 Static analys
StaticAnalysis.java	30	1500.0	10.00	CSE 565 Static analys
UseEqualsToComp	1	50.0	0.33	CSE 565 Static analys
UseUtilityClass	1	50.0	0.33	CSE 565 Static analys
IfElseStmtsMustUs	2	100.0	0.67	CSE 565 Static analys
MethodArgument	4	200.0	1.33	CSE 565 Static analys
AvoidLiteralsInIfC	2	100.0	0.67	CSE 565 Static analys
CommentRequire	4	200.0	1.33	CSE 565 Static analys
SystemPrintln	1	50.0	0.33	CSE 565 Static analys
LocalVariableCoul	3	150.0	1.00	CSE 565 Static analys
ControlStatement	2	100.0	0.67	CSE 565 Static analys
CompareObjectsV	1	50.0	0.33	CSE 565 Static analys
UnusedAssignmer	2	100.0	0.67	CSE 565 Static analys
DataflowAnomaly	4	200.0	1.33	CSE 565 Static analys
UnusedLocalVarial	2	100.0	0.67	CSE 565 Static analys
NoPackage	1	50.0	0.33	CSE 565 Static analys

- To conclude, the tool is free, takes no configuration effort, and it provides a detailed analysis.

References:

[1] https://stackoverflow.com/questions/48266017/wh
at- kind-of-test-coverage-criteria-eclipse-uses

[2] https://www.tutorialspoint.com/software_testing_diction
ary/anomaly.htm