Assignment: 3 ID: 1207654691

Part 1: Code Coverage Tool

1. Description of tool and type of coverages it provides:
   Tool used: *CodeCover*
   - It is a free glass box testing tool.
   - CodeCover uses the template engine Velocity.
   - Open language interface, available languages: Java and COBOL.

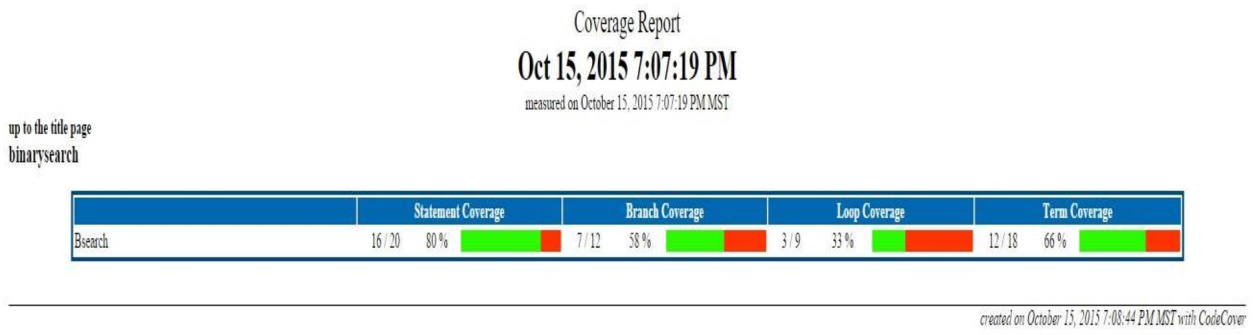
   Coverage provided by the tool:

   - Statement coverage,
   - Branch coverage,
   - Loop coverage,
   - Term coverage,
   - Question mark operator coverage,
   - Synchronized coverage.
2. Source listing of the code:

```
public class Bsearch {
 static int[] data;
 static int size, key;
 static boolean flag;
public static int binarySearch(int key,int[] data){
        int low = 0;
        int size = data.length;
        int high = size - 1;
        if(data == null)
                                        return -2;
                          else if(data.length == 0) {

                 }
                 for (int i = 0; i < data.length-1; i++) {

                          if(data[i] > data[i+1]) {
                          return -2;
                     }
                 }

        while(high >= low) {
           int middle = (low + high) / 2;
           if(data[middle] == key) {
             return middle;
           }
           if(data[middle] < key) {
               low = middle + 1;
           }
           if(data[middle] > key) {
               high = middle - 1;
           }
       }
       return -1;
}
```

3. Test cases:
    i. Key Found: Inputs = {1,2,5,6,7,8} key = 5
    ii. Key Not Found: Input = {1,2,5,6,7,8} Key = 10
    iii. Not Sorted array: Input = {1,5,3,4,6,7} Key = 5

4. Screen Shots:



Coverage Report

Oct 15, 2015 7:07:19 PM

measured on October 15, 2015 7:07:19 PM MST

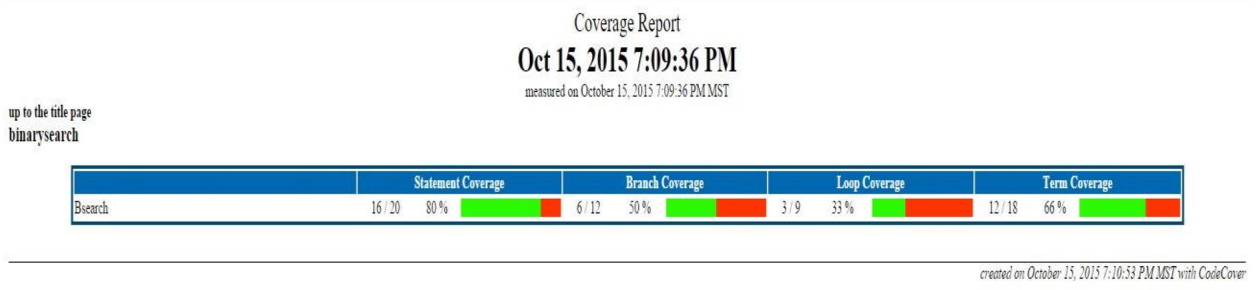
up to the title page

binarysearch

	Statement Coverage	Branch Coverage	Loop Coverage	Term Coverage
Bsearch	16 / 20 80 %	7 / 12 58 %	3 / 9 33 %	12 / 18 66 %

created on October 15, 2015 7:08:44 PM MST with CodeCover

[i] Key Found



Coverage Report

Oct 15, 2015 7:09:36 PM

measured on October 15, 2015 7:09:36 PM MST

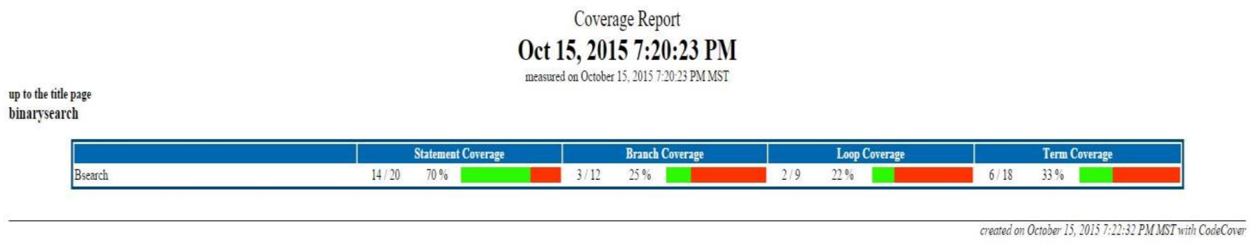
up to the title page

binarysearch

	Statement Coverage	Branch Coverage	Loop Coverage	Term Coverage
Bsearch	16 / 20 80 %	6 / 12 50 %	3 / 9 33 %	12 / 18 66 %

created on October 15, 2015 7:10:53 PM MST with CodeCover

[ii] Key Not Found



Coverage Report

Oct 15, 2015 7:20:23 PM

measured on October 15, 2015 7:20:23 PM MST

up to the title page

binarysearch

	Statement Coverage	Branch Coverage	Loop Coverage	Term Coverage
Bsearch	14 / 20 70 %	3 / 12 25 %	2 / 9 22 %	6 / 18 33 %

created on October 15, 2015 7:22:32 PM MST with CodeCover

[iii] Array Not Sorted

5. Evaluation of tool's usefulness:

- Easy to implement because it needs to install only plugins in eclipse.
- Easy to apply on code.
- It will covers statement, decision/branch, loop, term, question mark operator and synchronized coverage.

Part 2: Static source code analysis:

1. Description of tool and the types of analysis it provides:

   Tool used: *PMD*

- PMD is a source code analyzer. It finds common programming flaws like unused variables, empty catch blocks, unnecessary object creation, and so forth. It supports Java, JavaScript, PLSQL, Apache Velocity, XML, XSL.

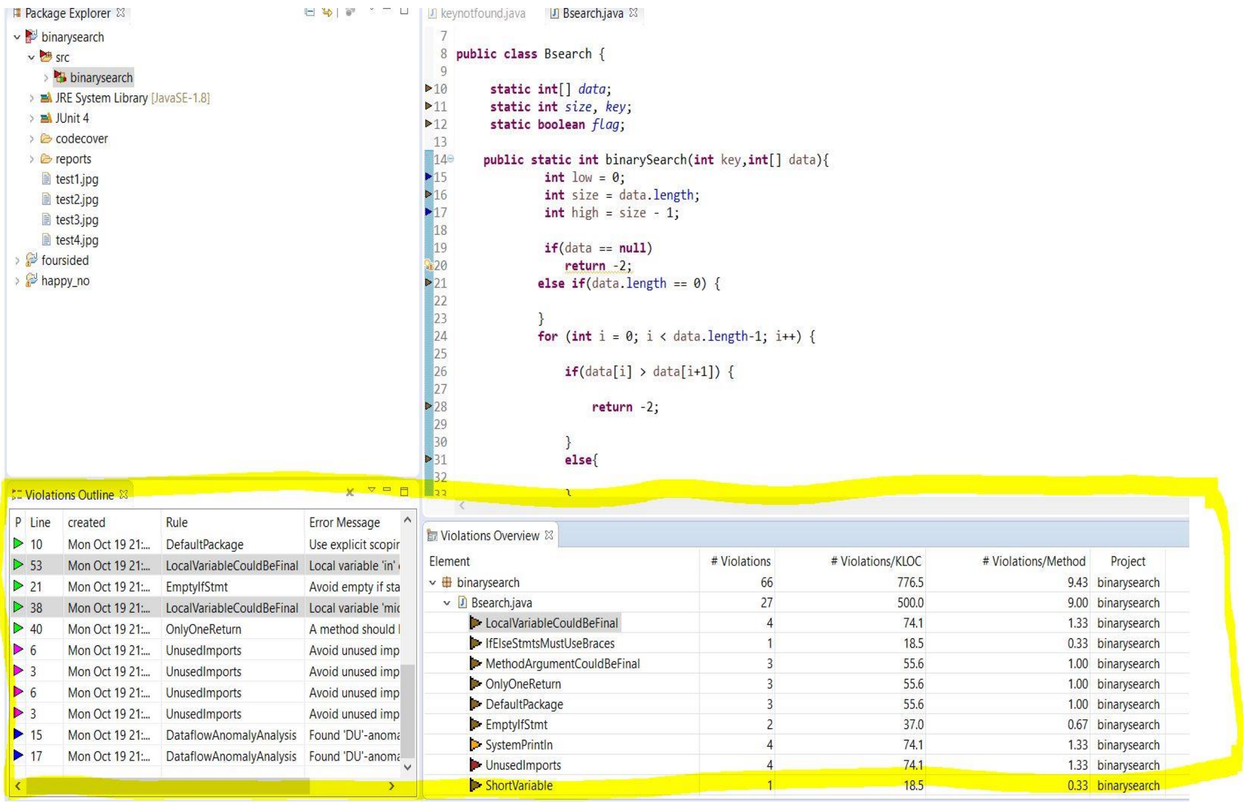
Types of analysis:

- Only one return
- Default package
- Short variable
- Data flow anomaly analysis
- Unused imports.

2. Source listing of your code:

```
public static int binarySearch(int key,int[] data){
        int low = 0;                          → Data Flow Anomaly
        int temp = 0;                         → Data Flow Anomaly
        int temp2 = 0;                        → Data Flow Anomaly
        int size = data.length;
        int high = size - 1;                  → Data Flow Anomaly
        if(data == null)
                            return -2;
                    else if(data.length == 0) {
                }
                for (int i = 0; i < data.length-1; i++) {
                    if(data[i] > data[i+1]) {
                    return -2;
                  }
                  else{
                  }
                }
        while(high >= low) {
           int middle = (low + high) / 2;
           if(data[middle] == key) {
             return middle;
           }
           if(data[middle] < key) {
              low = middle + 1;
           }
           if(data[middle] > key) {
              high = middle - 1;
           }
        }
return -1;
    }
```

3. Screen Shots:



Package Explorer

```
binarysearch
  src
    binarysearch
  JRE System Library [JavaSE-1.8]
  JUnit 4
  codecover
  reports
  test1.jpg
  test2.jpg
  test3.jpg
  test4.jpg
foursided
happy_no
```

keynotfound.java | Bsearch.java

```java
public class Bsearch {

    static int[] data;
    static int size, key;
    static boolean flag;

    public static int binarySearch(int key,int[] data){
            int low = 0;
            int size = data.length;
            int high = size - 1;

            if(data == null)
               return -2;
           else if(data.length == 0) {

           }
           for (int i = 0; i < data.length-1; i++) {

               if(data[i] > data[i+1]) {

                   return -2;

               }
               else{
```

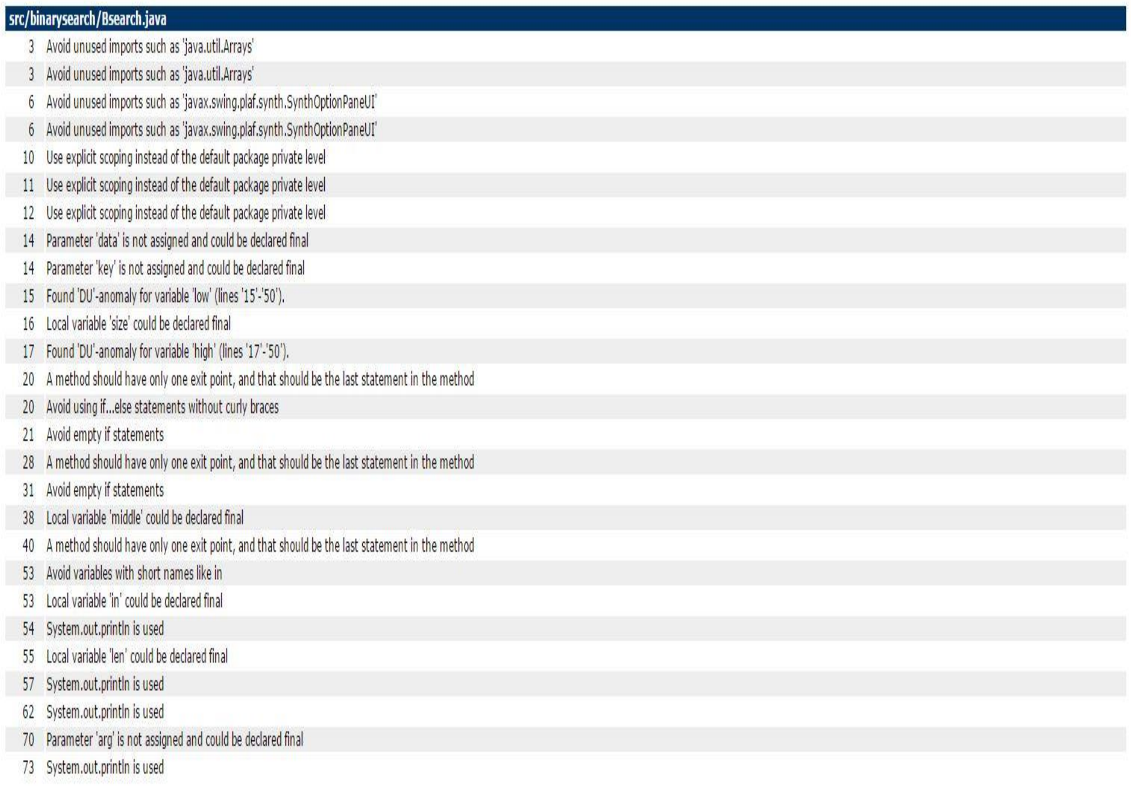
Violations Outline

P	Line	created	Rule	Error Message
▶	10	Mon Oct 19 21:...	DefaultPackage	Use explicit scopir
▶	53	Mon Oct 19 21:...	LocalVariableCouldBeFinal	Local variable 'in'
▶	21	Mon Oct 19 21:...	EmptyIfStmt	Avoid empty if sta
▶	38	Mon Oct 19 21:...	LocalVariableCouldBeFinal	Local variable 'mic
▶	40	Mon Oct 19 21:...	OnlyOneReturn	A method should
▶	6	Mon Oct 19 21:...	UnusedImports	Avoid unused imp
▶	3	Mon Oct 19 21:...	UnusedImports	Avoid unused imp
▶	6	Mon Oct 19 21:...	UnusedImports	Avoid unused imp
▶	3	Mon Oct 19 21:...	UnusedImports	Avoid unused imp
▶	15	Mon Oct 19 21:...	DataflowAnomalyAnalysis	Found 'DU'-anoma
▶	17	Mon Oct 19 21:...	DataflowAnomalyAnalysis	Found 'DU'-anoma

Violations Overview

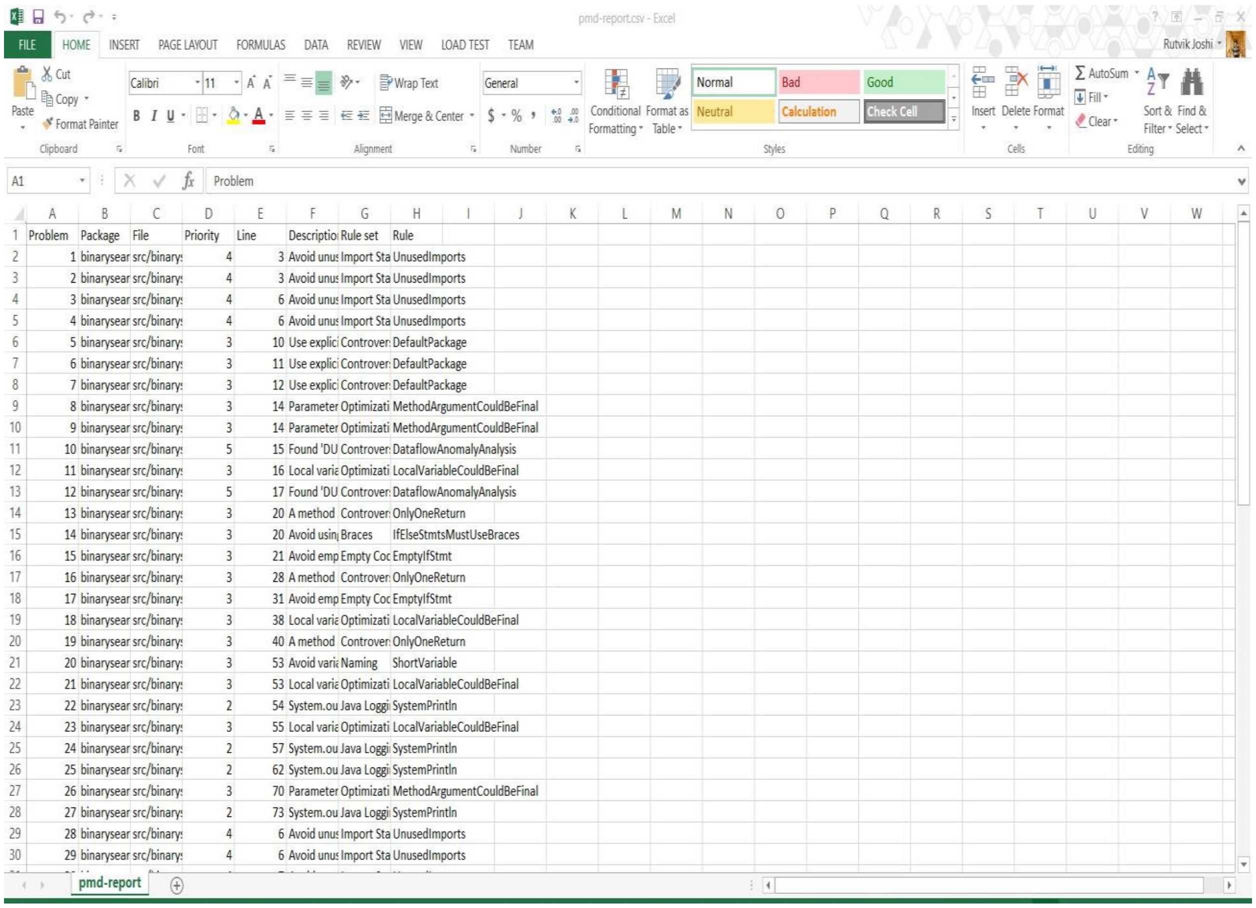
Element	# Violations	# Violations/KLOC	# Violations/Method	Project
binarysearch	66	776.5	9.43	binarysearch
Bsearch.java	27	500.0	9.00	binarysearch
LocalVariableCouldBeFinal	4	74.1	1.33	binarysearch
IfElseStmtsMustUseBraces	1	18.5	0.33	binarysearch
MethodArgumentCouldBeFinal	3	55.6	1.00	binarysearch
OnlyOneReturn	3	55.6	1.00	binarysearch
DefaultPackage	3	55.6	1.00	binarysearch
EmptyIfStmt	2	37.0	0.67	binarysearch
SystemPrintln	4	74.1	1.33	binarysearch
UnusedImports	4	74.1	1.33	binarysearch
ShortVariable	1	18.5	0.33	binarysearch

[i] Eclipse violation overview



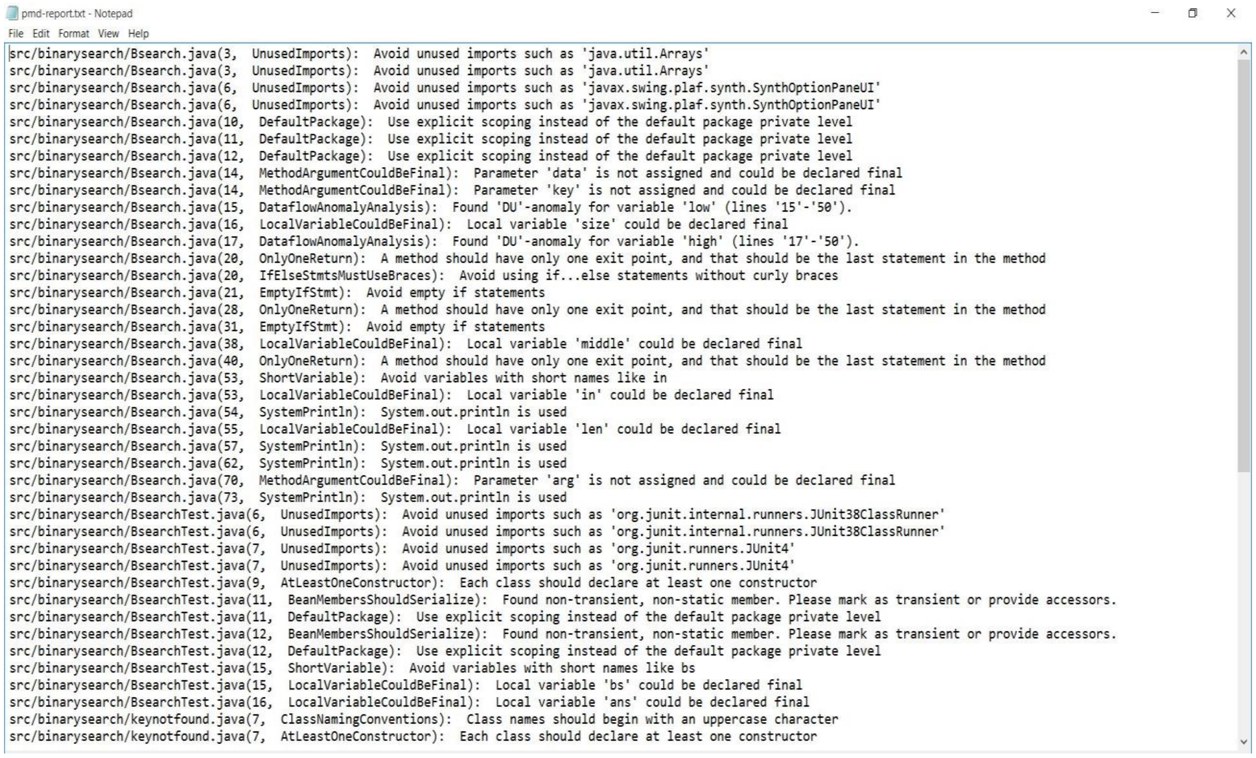
src/binarysearch/Bsearch.java	
3	Avoid unused imports such as 'java.util.Arrays'
3	Avoid unused imports such as 'java.util.Arrays'
6	Avoid unused imports such as 'javax.swing.plaf.synth.SynthOptionPaneUI'
6	Avoid unused imports such as 'javax.swing.plaf.synth.SynthOptionPaneUI'
10	Use explicit scoping instead of the default package private level
11	Use explicit scoping instead of the default package private level
12	Use explicit scoping instead of the default package private level
14	Parameter 'data' is not assigned and could be declared final
14	Parameter 'key' is not assigned and could be declared final
15	Found 'DU'-anomaly for variable 'low' (lines '15'-'50').
16	Local variable 'size' could be declared final
17	Found 'DU'-anomaly for variable 'high' (lines '17'-'50').
20	A method should have only one exit point, and that should be the last statement in the method
20	Avoid using if...else statements without curly braces
21	Avoid empty if statements
28	A method should have only one exit point, and that should be the last statement in the method
31	Avoid empty if statements
38	Local variable 'middle' could be declared final
40	A method should have only one exit point, and that should be the last statement in the method
53	Avoid variables with short names like in
53	Local variable 'in' could be declared final
54	System.out.println is used
55	Local variable 'len' could be declared final
57	System.out.println is used
62	System.out.println is used
70	Parameter 'arg' is not assigned and could be declared final
73	System.out.println is used

[ii] HTML report



Problem	Package	File	Priority	Line	Descriptio	Rule set	Rule
1	binarysear	src/binary	4	3	Avoid unus	Import Sta	UnusedImports
2	binarysear	src/binary	4	3	Avoid unus	Import Sta	UnusedImports
3	binarysear	src/binary	4	6	Avoid unus	Import Sta	UnusedImports
4	binarysear	src/binary	4	6	Avoid unus	Import Sta	UnusedImports
5	binarysear	src/binary	3	10	Use explici	Controver	DefaultPackage
6	binarysear	src/binary	3	11	Use explici	Controver	DefaultPackage
7	binarysear	src/binary	3	12	Use explici	Controver	DefaultPackage
8	binarysear	src/binary	3	14	Parameter	Optimizati	MethodArgumentCouldBeFinal
9	binarysear	src/binary	3	14	Parameter	Optimizati	MethodArgumentCouldBeFinal
10	binarysear	src/binary	5	15	Found 'DU	Controver	DataflowAnomalyAnalysis
11	binarysear	src/binary	3	16	Local varia	Optimizati	LocalVariableCouldBeFinal
12	binarysear	src/binary	5	17	Found 'DU	Controver	DataflowAnomalyAnalysis
13	binarysear	src/binary	3	20	A method	Controver	OnlyOneReturn
14	binarysear	src/binary	3	20	Avoid usin	Braces	IfElseStmtsMustUseBraces
15	binarysear	src/binary	3	21	Avoid emp	Empty Coc	EmptyIfStmt
16	binarysear	src/binary	3	28	A method	Controver	OnlyOneReturn
17	binarysear	src/binary	3	31	Avoid emp	Empty Coc	EmptyIfStmt
18	binarysear	src/binary	3	38	Local varia	Optimizati	LocalVariableCouldBeFinal
19	binarysear	src/binary	3	40	A method	Controver	OnlyOneReturn
20	binarysear	src/binary	3	53	Avoid varia	Naming	ShortVariable
21	binarysear	src/binary	3	53	Local varia	Optimizati	LocalVariableCouldBeFinal
22	binarysear	src/binary	2	54	System.ou	Java Loggi	SystemPrintln
23	binarysear	src/binary	3	55	Local varia	Optimizati	LocalVariableCouldBeFinal
24	binarysear	src/binary	2	57	System.ou	Java Loggi	SystemPrintln
25	binarysear	src/binary	2	62	System.ou	Java Loggi	SystemPrintln
26	binarysear	src/binary	3	70	Parameter	Optimizati	MethodArgumentCouldBeFinal
27	binarysear	src/binary	2	73	System.ou	Java Loggi	SystemPrintln
28	binarysear	src/binary	4	6	Avoid unus	Import Sta	UnusedImports
29	binarysear	src/binary	4	6	Avoid unus	Import Sta	UnusedImports

[iii] CSV report



```
src/binarysearch/Bsearch.java(3,  UnusedImports):  Avoid unused imports such as 'java.util.Arrays'
src/binarysearch/Bsearch.java(3,  UnusedImports):  Avoid unused imports such as 'java.util.Arrays'
src/binarysearch/Bsearch.java(6,  UnusedImports):  Avoid unused imports such as 'javax.swing.plaf.synth.SynthOptionPaneUI'
src/binarysearch/Bsearch.java(6,  UnusedImports):  Avoid unused imports such as 'javax.swing.plaf.synth.SynthOptionPaneUI'
src/binarysearch/Bsearch.java(10,  DefaultPackage):  Use explicit scoping instead of the default package private level
src/binarysearch/Bsearch.java(11,  DefaultPackage):  Use explicit scoping instead of the default package private level
src/binarysearch/Bsearch.java(12,  DefaultPackage):  Use explicit scoping instead of the default package private level
src/binarysearch/Bsearch.java(14,  MethodArgumentCouldBeFinal):  Parameter 'data' is not assigned and could be declared final
src/binarysearch/Bsearch.java(14,  MethodArgumentCouldBeFinal):  Parameter 'key' is not assigned and could be declared final
src/binarysearch/Bsearch.java(15,  DataflowAnomalyAnalysis):  Found 'DU'-anomaly for variable 'low' (lines '15'-'50').
src/binarysearch/Bsearch.java(16,  LocalVariableCouldBeFinal):  Local variable 'size' could be declared final
src/binarysearch/Bsearch.java(17,  DataflowAnomalyAnalysis):  Found 'DU'-anomaly for variable 'high' (lines '17'-'50').
src/binarysearch/Bsearch.java(20,  OnlyOneReturn):  A method should have only one exit point, and that should be the last statement in the method
src/binarysearch/Bsearch.java(20,  IfElseStmtsMustUseBraces):  Avoid using if...else statements without curly braces
src/binarysearch/Bsearch.java(21,  EmptyIfStmt):  Avoid empty if statements
src/binarysearch/Bsearch.java(28,  OnlyOneReturn):  A method should have only one exit point, and that should be the last statement in the method
src/binarysearch/Bsearch.java(31,  EmptyIfStmt):  Avoid empty if statements
src/binarysearch/Bsearch.java(38,  LocalVariableCouldBeFinal):  Local variable 'middle' could be declared final
src/binarysearch/Bsearch.java(40,  OnlyOneReturn):  A method should have only one exit point, and that should be the last statement in the method
src/binarysearch/Bsearch.java(53,  ShortVariable):  Avoid variables with short names like in
src/binarysearch/Bsearch.java(53,  LocalVariableCouldBeFinal):  Local variable 'in' could be declared final
src/binarysearch/Bsearch.java(54,  SystemPrintln):  System.out.println is used
src/binarysearch/Bsearch.java(55,  LocalVariableCouldBeFinal):  Local variable 'len' could be declared final
src/binarysearch/Bsearch.java(57,  SystemPrintln):  System.out.println is used
src/binarysearch/Bsearch.java(62,  SystemPrintln):  System.out.println is used
src/binarysearch/Bsearch.java(70,  MethodArgumentCouldBeFinal):  Parameter 'arg' is not assigned and could be declared final
src/binarysearch/Bsearch.java(73,  SystemPrintln):  System.out.println is used
src/binarysearch/BsearchTest.java(6,  UnusedImports):  Avoid unused imports such as 'org.junit.internal.runners.JUnit38ClassRunner'
src/binarysearch/BsearchTest.java(6,  UnusedImports):  Avoid unused imports such as 'org.junit.internal.runners.JUnit38ClassRunner'
src/binarysearch/BsearchTest.java(7,  UnusedImports):  Avoid unused imports such as 'org.junit.runners.JUnit4'
src/binarysearch/BsearchTest.java(7,  UnusedImports):  Avoid unused imports such as 'org.junit.runners.JUnit4'
src/binarysearch/BsearchTest.java(9,  AtLeastOneConstructor):  Each class should declare at least one constructor
src/binarysearch/BsearchTest.java(11,  BeanMembersShouldSerialize):  Found non-transient, non-static member. Please mark as transient or provide accessors.
src/binarysearch/BsearchTest.java(11,  DefaultPackage):  Use explicit scoping instead of the default package private level
src/binarysearch/BsearchTest.java(12,  BeanMembersShouldSerialize):  Found non-transient, non-static member. Please mark as transient or provide accessors.
src/binarysearch/BsearchTest.java(12,  DefaultPackage):  Use explicit scoping instead of the default package private level
src/binarysearch/BsearchTest.java(15,  ShortVariable):  Avoid variables with short names like bs
src/binarysearch/BsearchTest.java(15,  LocalVariableCouldBeFinal):  Local variable 'bs' could be declared final
src/binarysearch/BsearchTest.java(16,  LocalVariableCouldBeFinal):  Local variable 'ans' could be declared final
src/binarysearch/keynotfound.java(7,  ClassNamingConventions):  Class names should begin with an uppercase character
src/binarysearch/keynotfound.java(7,  AtLeastOneConstructor):  Each class should declare at least one constructor
```

[iv]Report in text format

4. Evolution of tool’s usefulness:
   - Easy to implement, we need to install plugins for eclipse to use it.
   - It generates different type of reports such as CSV, HTML, Text, XML, XLS etc.
   - It is easy to apply on code by checking its preferences in properties of project.
   - It differentiates different flows in different colors, so easy to identify different flows.
   - It shows flows in code line wise, so easy to identify in code.

References:

[i] Code Cover, http://codecover.org/

[ii] PMD, https://pmd.github.io/