Revealing Code Quality

CS585 Software Verification and Validation
http://cs585.yusun.io
January 26, 2015

Yu Sun, Ph.D.
http://yusun.io
yusun@cpp.edu

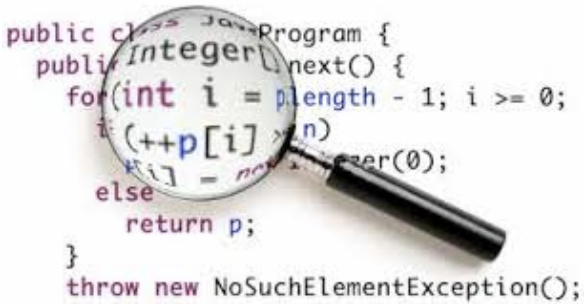
How to verify you are doing everything right?

Code Review: An Effective V&V Approach

- Average defect detection rate:
  - 25% for unit testing
  - 35% for functional testing
  - 45% for integration testing
  - 55% for design review
  - 60% for code review

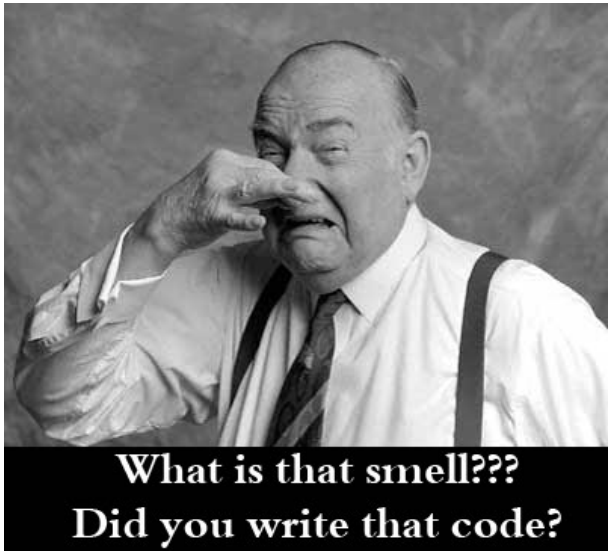
Static Code Analysis

- *Static code analysis* is the process of analyzing code without executing it
- Code review is a sort of static code analysis but is performed with humans or team members. Generally, static code analysis is performed by an automated tool



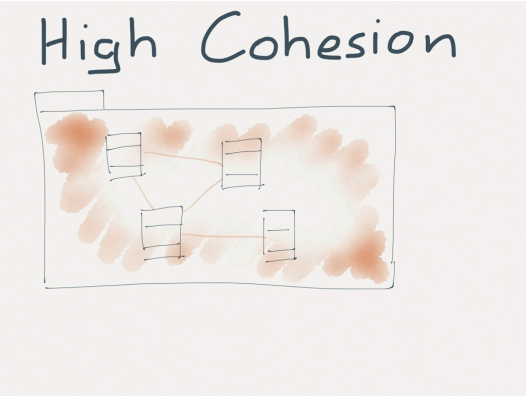
Check Violation of Coding Best Practices

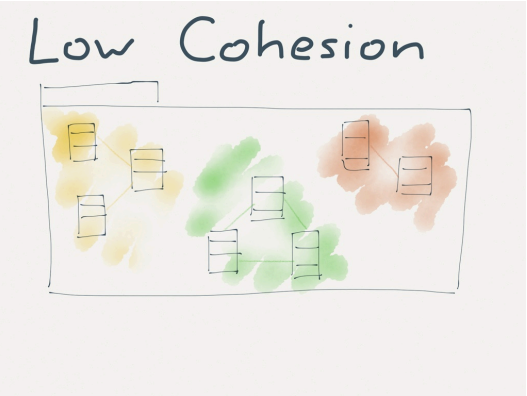
- Long method body
- Long parameter list
- Large classes
- Variable names
- … (all bad code smells)



Check Code Cohesion

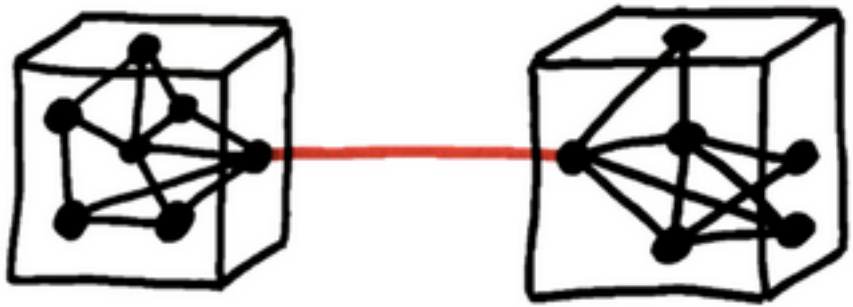
- Cohesion represents responsibility of a single module (class)
  - If a module or class possesses too many responsibilities, the class or module is less cohesive
  - Performing multiple dissimilar tasks introduces complexity and maintainability issues. High cohesion means performing only a particular type of task





Check Code Coupling

- Coupling measures the dependency on other modules or code
  - Low dependency enforces high cohesion. If module C depends on two other modules, A and B, any change in the APIs of A or B will force C to change



Check Cyclomatic Complexity

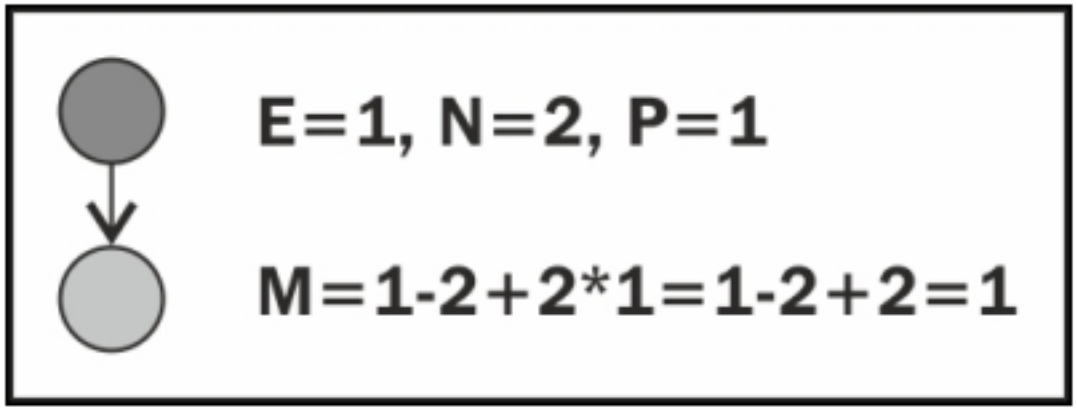
- *Cyclomatic complexity* measures the complexity of a program - the number of linearly independent paths in a program
- Complexity is represented as:

$$M = E - N + 2P$$

  - M is complexity, E is the number of edges of the graph, N is the number of nodes of the graph, and P is the number of connected components

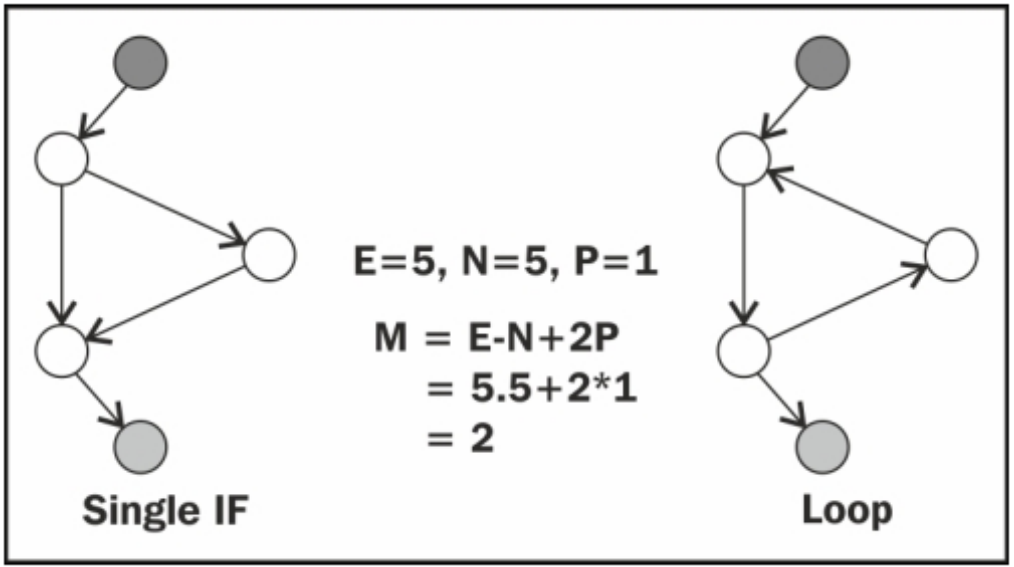
Check Cyclomatic Complexity

- A method that has no conditional statements has a cyclomatic complexity of 1



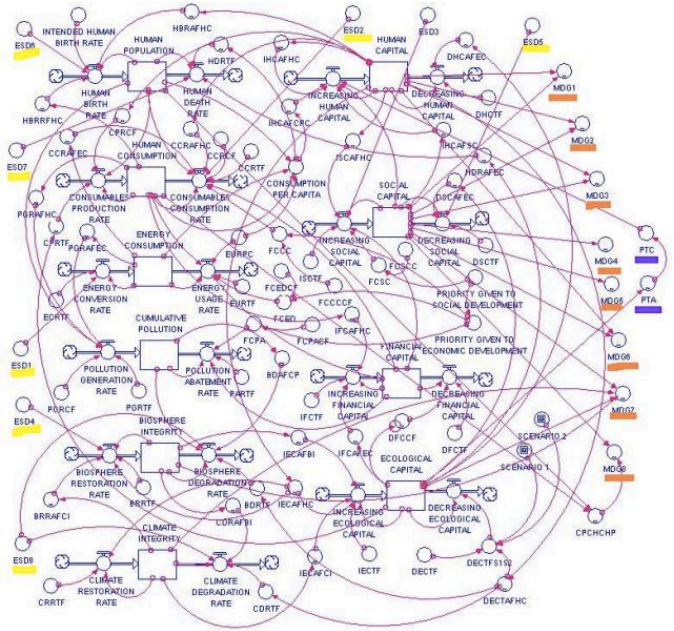
Check Cyclomatic Complexity

- A method with a single condition (an IF) or a single loop (a FOR) has a complexity of 2



Check Cyclomatic Complexity

- Any method with a complexity greater than 10 has a serious problem



Apply Tools to Automate the Checking

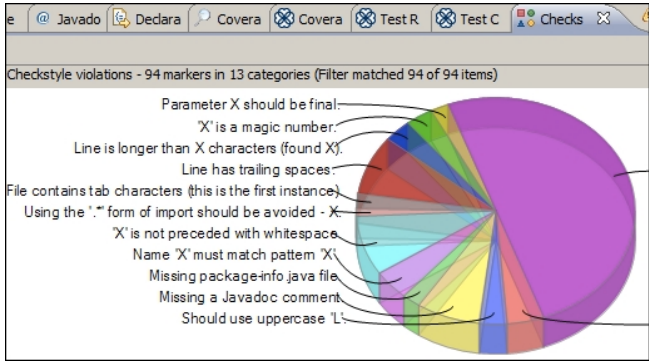
- Tools and automations are the key of software engineering

Working with the Checkstyle Plugin

- Missing Javadoc comments
- The use of magic numbers
- Naming conventions of variables and methods
- Method's argument length and line lengths
- The use of imports
- The spaces between some characters
- The good practices of class construction
- Duplicated code

Overview of Checkstyle violations - 94 markers in 13 categories (Filter matched 94 of 94 items)

Checkstyle violation type	Marker count
File contains tab characters (this is the first instance).	2
Parameter X should be final.	2
Should use uppercase 'L'.	3
Line is longer than X characters (found X).	3
'X' is a magic number.	3
Method 'X' is not designed for extension - needs to be abstract, final or empty.	4
Name 'X' must match pattern 'X'.	4
Missing a Javadoc comment.	7
'X' is not preceded with whitespace.	8
Line has trailing spaces.	8
'X' is not followed by whitespace.	47



Checkstyle Maven Plugin

```
<reporting>
    <plugins>
       <plugin>
         <groupId>org.apache.maven.plugins</groupId>
         <artifactId>maven-checkstyle-plugin</artifactId>
         <version>2.13</version>
         <reportSets>
           <reportSet>
             <reports>
               <report>checkstyle</report>
             </reports>
           </reportSet>
         </reportSets>
       </plugin>
    </plugins>
</reporting>
```

mvn checkstyle:checktyle

Checkstyle Quiz

- Should array initialization contain a trailing comma?

```
int[] a = new int[]
{
    1,
    2,
    3,
};
```

```
int[] a = new int[]
{
    1,
    2,
    3
};
```

Checkstyle Quiz

- Any problem with the following statement?

```
if (myStringObject.equals("My_Sweet_String")) {
        ...
}
```

```
if ("My_Sweet_String".equals(myStringObject)) {
        ...
}
```

Checkstyle Quiz

- Any problem with the following statement?

```
Boolean isSkipped = true;
```

Checkstyle Quiz

- Any problem with the following statement?

```
If (value == 54) {
    return value;
}
```

Checks that there are no "magic numbers", where a magic number is a numeric literal that is not defined as a constant. By default, -1, 0, 1, and 2 are not considered to be magic numbers.

Checkstyle Quiz

- Any problem with the following statement?

```
if (valid())
    return false;
else
    return true;
```

Checks for over-complicated boolean return statements

```
return !valid();
```

Checkstyle Quiz

- What's the ideal number of return statements in a method?

```
if (valid())
    return false;
else
    return true;
```

Restricts the number of return statements (2 by default). Too many return points can mean that code is attempting to do too much or may be difficult to understand.

Checkstyle Quiz

- What's the order of class declaration?

```
Instance variables
Static variables
Constructors
Methods
(public, private, protected, default)
```

1. Class (static) variables. First the public class variables, then protected, then package level (no access modifier), and then private.
2. Instance variables. First the public class variables, then protected, then package level (no access modifier), and then private.
3. Constructors
4. Methods

Checkstyle Quiz

- Any problem with the following statement?

```
public class MyTest {
    private boolean isTested = false;
    private int target = 0;

...
}
```

```
public class MyTest {
    private boolean isTested;
    private int target;

...
}
```

Checkstyle Quiz

- Any problem with the following statement?

```
public class MyTest {
    private MyTest() {
        ...
    }
    ...
}
```

a class which has only private constructors is declared as final

Checkstyle Quiz

- Any problem with the following statement?

```
public class StringUtils {  // A String utility class
    public static int count(char c, String s) {
        // ...
    }
}
```

```
public class StringUtils // not final to allow subclassing
{
    protected StringUtils() {
        // prevents calls from subclass
        throw new UnsupportedOperationException();
    }

    public static int count(char c, String s) {
        // ...
    }
}
```

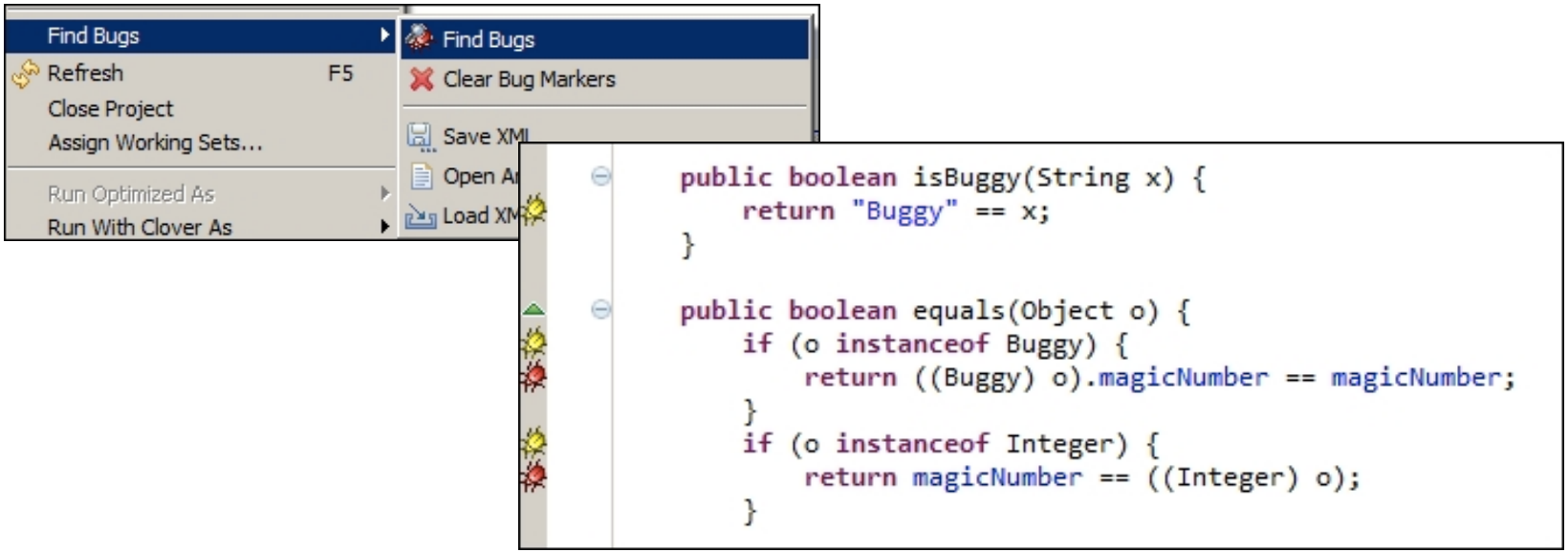
Checkstyle Quiz

- Where to declare nested (inner) class/interface?

nested (inner) classes/interfaces are declared at the bottom of the class after all method and field declarations

Exploring the FindBugs Plugin

- A program which uses static analysis to look for bugs in Java code



Exploring the FindBugs Plugin

- Correctness bug
  - an apparent coding mistake that results in code that was probably not what the developer intended, e.g, a method ignores the return value of a self-assigned field
- Bad practice
  - violations of recommended best practices and essential coding practice
- Dodgy errors
  - code that is confusing, anomalous, or written in a way that leads to errors

FindBugs Maven Plugin

```
<reporting>
    <plugins>
       <plugin>
         <groupId>org.codehaus.mojo</groupId>
         <artifactId>findbugs-maven-plugin</artifactId>
         <version>3.0.1-SNAPSHOT</version>
       </plugin>
    </plugins>
</reporting>
```

mvn findbugs:findbugs

FindBugs Quiz

- How to clear a collection?

```
myList.removeAll(myList);
```

Don't use removeAll to clear a collection

If you want to remove all elements from a collection c, use c.clear, not c.removeAll(c). Calling c.removeAll(c) to clear a collection is less clear, susceptible to errors from typos, less efficient and for some collections, might throw a ConcurrentModificationException.

FindBugs Quiz

- How to quit your application?

```
System.exit(...)
```

Invoking System.exit shuts down the entire Java virtual machine. This should only been done when it is appropriate. Such calls make it hard or impossible for your code to be invoked by other code. Consider throwing a RuntimeException instead.

FindBugs Quiz

- What is the problem with the following statement?

```
If (myString == ”message” ) {
    ...
}
```

This code compares java.lang.String objects for reference equality using the == or != operators. Unless both strings are either constants in a source file, or have been interned using the String.intern() method, the same string value may be represented by two different String objects. Consider using the equals(Object) method instead.

FindBugs Quiz

- equals() and hashCode()

HE: Class defines equals() but not hashCode()
HE: Class defines equals() and uses Object.hashCode()
HE: Class defines hashCode() but not equals()
HE: Class defines hashCode() and uses Object.equals()

```
If you want to override, override both!
Equal objects should have the same hash code.
```

FindBugs Quiz

- What's the issue with the following method?

```
private int num;
private String data;

public boolean equals(Object obj) {
          if (this == obj)
                    return true;
          if ((obj.getClass() != this.getClass())
                    return false;
          // object must be Test at this point
          Test test = (Test) obj;
          return num == test.num &&
              (data == test.data || (data != null && data.equals(test.data)));
}
```

NP: equals() method does not check for null argument

FindBugs Quiz

- What's the issue with the following method?

```
Class ErrorStateException {
    private String errorMessage;

    public void setErrorMessage(String errorMessage) {
        this.errorMessage = errorMessage;
    }

    public StringgetErrorMessage() {
        return errorMessage;
    }
}
```

Nm: Class is not derived from an Exception, even though it is named as such

FindBugs Quiz

- What's the issue with the following method?

```
public void deleteTmpFile(File myTempFile) {
    try {
        myTempFile.delete();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

RV: Method ignores exceptional return value

FindBugs Quiz

- What's the issue with the following method?

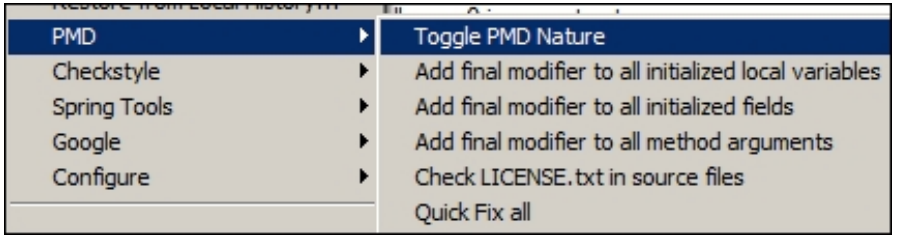
```
public void copyContent() {
          File file = new File("C:/robots.txt");
          FileInputStream fis = null;

          try {
                    fis = new FileInputStream(file);

                    int content;
                    while ((content = fis.read()) != -1) {
                              // convert to char and display it
                              System.out.print((char) content);
                    }
          } catch (IOException e) {
                    e.printStackTrace();
          }
}
```

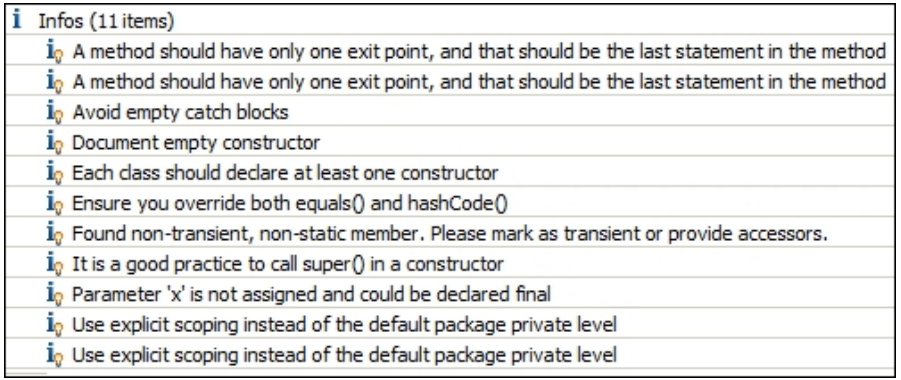
FindBugs Quiz

```
public void copyContent() {
          File file = new File("C:/robots.txt");
          FileInputStream fis = null;
          try {
                    fis = new FileInputStream(file);
                    int content;
                    while ((content = fis.read()) != -1) {
                              // convert to char and display it
                              System.out.print((char) content);
                    }
          } catch (IOException e) {
                    e.printStackTrace();
          } finally {
                    try {
                              if (fis != null)
                                        fis.close();
                    } catch (IOException ex) {
                              ex.printStackTrace();
                    }
          }
}
```

Working with the PMD Plugin

- PMD can find
  - duplicate code
  - dead code
  - empty if/while statements
  - empty try/catch blocks
  - complicated expressions
  - cyclomatic complexity
  - …





Monitoring Code Quality with SonarQube

- SonarQube is a web-based open source continuous quality assessment dashboard for:
  - Bugs and potential bugs
  - Breach in coding standards
  - Duplications
  - Lack of unit tests
  - Bad distribution of complexities
  - Spaghetti design
  - Not enough or too many comments