Concordia University

SOEN 6011 - Software Engineering Process

ETERNITY: FUNCTIONS
arccos(x)

Deliverable 3

Surya Prakash Govindaraju
Student ID : 40085527
Team D

https://github.com/Surya64/SOEN-6011

Contents

1 Source Code Review

For Source Code review of function F2, I have the used the Codacy automate tool for analysis.

1.1 Codacy

Codacy is an automated source code review tool used by many organizations to provide quality software. It automatically identifies issues through static code analysis. The main features are code review automation, code quality analytics, engineering analytics and security code analysis. It involves Checkstyle 8.13 and PMD 5.8.1 features combined.

Checkstyle is a development tool which checks for programming style. It supports Google Java Style Guide and Sun Code Conventions. It is flexible and can configure easily. On the other hand PMD checks for programming flaws in source code like unused variables, empty catch blocks and unnecessary object creation.

In Codacy, the issues are classified into different Categories like Security, Error Prone, Code Style, Compatibility, Unused Code and Performance. Below are the results of the review performed on function F2.

I was able to compile the source code without any errors in the Eclipse. The comments were understandable and easy to go through the flow the source code.

File Name	Issue	Duplication/Clones	Complexity
Calculator	3	0	15
CalculatorTest	2	0	1
CalculatorDriver	1	0	10
CalculatorInterface	0	0	-
LastTangentToOriginator	0	0	-
LastTangentToCareTaker	0	0	-
LastTangent	0	0	1

Table 1: Results from Codacy

File Name	LOC	Source LOC	Commented LOC
Calculator	87	64	10
CalculatorTest	64	38	12
CalculatorDriver	66	60	0
CalculatorInterface	12	7	0
LastTangentToOriginator	6	4	0
LastTangentToCareTaker	5	3	0
LastTangent	16	11	0

Table 2: Results from Codacy

Below are the issues found in java files.

1. Calculator.java
    - Avoid reassigning parameters such as 'x' in line 42.
    - The method tan() has a NPath complexity of 973, which contains lot of nested if/else statements. A threshold of 200 is generally considered the point where measures should be taken to reduce complexity and increase readability.
    - Avoid throwing raw exception types (Line 83). Use subclass exception or error instead.
2. CalculatorTest.java
    - Package name contains upper case characters.
    - assertTrue(!expr) can be replaced by assertFalse(expr) in Line 61.
3. CalculatorDriver.java
    - A switch statement does not contain a break.
4. It has some indentation errors on all the files.
5. Javadoc is missing

2 Testing of Function F3

The function provided for testing is sinh(x) is a hyperbolic function of sine. The User has provided all the required artifacts to perform the testing.

Below are the environments used for testing the function.

1. Java Eclipse environment - jdk 1.8
2. System used is windows7 with 4G RAM
3. JUnit 4.1 library

Test Step:

1. The source code is imported into Eclipse and compiled.
2. The program is run for different inputs and results are verified.
3. Run JUnit test case written.

Results:

Requirement	Description	Result
R1 , R2	Valid input within the range	PASS
R3	Accepts all digits from 1-9 with decimals	PASS
R4	Out of range throws exception	PASS
R5	Non-numeric throws exception	PASS

Table 3: Test Results

From the above, the results are positive and there are no errors found during the testing process. The Test Case covers all the requirements specified. The User can understand the steps to perform the calculation as its a textual interface.

Bibliography

[1] Codacy,
`https://www.codacy.com/`

[2] CheckStyle,
`https://en.wikipedia.org/wiki/Checkstyle`

[3] PMD,
`https://en.wikipedia.org/wiki/PMD_(software)`