SOEN6011- Software Engineering Processes

Deliverable 2

F3: sinh(x)

Github Link: https://github.com/SinglaAnkur/SOEN_6011

Submitted by:
Ankur Singla
40090208

Contents

1 Problem 4 **3**
1.1 Debugger: 3
1.1.1 Advantages 3
1.1.2 Disadvantages 3
1.2 Tool used to check code quality 4
1.2.1 Advantages 4
1.2.2 Disadvantages 4
1.3 Efforts made to achieve quality attributes 5
1.3.1 Correctness: 5
1.3.2 Efficient: 5
1.3.3 Maintainable: 5
1.3.4 Robust: 6
1.3.5 Usable: 6

2 References **7**

1 Problem 4

1.1 Debugger:

Debugger makes it easy to pause the execution of a program at a certain point and inspect the code.[1] IntelliJ IDEA provides inbuilt Java Debugger. Below are the advantages and disadvantages .

1.1.1 Advantages

1.1.1.1 IntelliJ IDEA is a good tool for application development. Inbuilt debugger saves the hassle of installing a separate software for debugging.

1.1.1.2 Adding a breakpoint[2] is just a click of mouse. Breakpoint could be added to a statement or a method.

1.1.1.3 IntelliJ Debugger works comparatively faster than other(Eclipse) debuggers.

1.1.1.4 If a statement contains multiple method calls, then its "Smart Step Into"[3] feature allows to step into a particular method in the call.

1.1.1.5 It allows to interact with the result of evaluating expression at any given instantanious time.

1.1.2 Disadvantages

1.1.2.1 It encounters large RAM usage. Sometimes while debugging CPU usage spikes high.

1.2 Tool used to check code quality

Checkstyle is a tool used to analyze source code in software development. It ensures that proper coding rules/standards are being followed. Below are the advantages and disadvantages of using Checkstyle..

1.2.1 Advantages

1.2.1.1 Checkstyle automates the process of source code check and enforces good programming practices.

1.2.1.2 It reports possible different ways of coding which is not only useful for testing but for learning as well.

1.2.1.3 It helps to improve the quality, readability and reusability of the code[4].

1.2.1.4 It could be configured to support any coding standard.

1.2.1.5 Provides support for JavaDocs.

1.2.1.6 Easy to setup[5].

1.2.2 Disadvantages

1.2.2.1 This tool cannot state if the code is correct or complete.

1.2.2.2 Strict imposition of the coding conventions leads to difficulty in sharing code with outside parties unless they follow same set of standards.

1.3 Efforts made to achieve quality attributes

1.3.1 Correctness:

You can never say if a code is 100% correct. However, through software testing you can increase your **confidence** in the code. Below is the efforts made to achieve this:

1.3.1.1 Unit test cases have been written manually to ensure actual result matches expected result for each method.

1.3.1.2 It helps to identify errors, gaps or any missing requirements[6].

1.3.1.3 IntelliJ comes with preinstalled JaCoCo library which provides report on code coverage at class and method level[7].

1.3.2 Efficient:

1.3.2.1 Use of efficient algorithm to calculate the value of e^x in order to reduce the complexity.

1.3.2.2 Unwanted loops and variables were removed to reduce time complexity and memory consumption.

1.3.2.3 For x>15 in $sinh(x)$, value of $sinh(x) \approx \frac{e^x}{2}$
For x<-15 in $sinh(x)$, value of $sinh(x) \approx \frac{-e^{-x}}{2}$
So it was enough to calculate just one of the exponential values in each case.

1.3.3 Maintainable:

1.3.3.1 IntelliJ has inbuilt automatic code refactoring whcih makes it easy to maintain the code with just a click of mouse.

1.3.3.2 Classes and variables have been created with descriptive naming. As per **Martin Fowler**[8]:
"Good programmers write code that humans can understand."

1.3.3.3 Proper use of comments only when required.
Developing a maintainable software helps in reducing **Technical Debt**.

1.3.4 Robust:

1.3.4.1 The software has been built to handle any kind of unusual input or condition.

1.3.4.2 Numerous test cases have been written and tested based on undesired user inputs to provide better error management.

1.3.5 Usable:

1.3.5.1 Different classes have been divided into different modules. Classes have been decoupled where ever possible.

1.3.5.2 Various classes have been extended to reuse the functionality.

2 References

[1] TutorialsPoint Team, 'Intellij Idea - Debugging'. [Online]. Available: https://www.tutorialspoint.com/intellij_idea/intellij_idea_debugging.htm [Accessed: 23-July-2019]
[2] Wikipedia, 'Breakpoint'. [Online]. Available: https://en.wikipedia.org/wiki/Breakpoint[Accessed: 23-July-2019]
[3] IntelliJ IDEA Team, 'Debugging'. [Online]. Available: https://www.jetbrains.com/help/idea/debugging-code.html [Accessed: 23-July-2019]
[4] Wikipedia, 'Checkstyle'. [Online]. Available: https://en.wikipedia.org/wiki/Checkstyle [Accessed: 25-July-2019]
[5] J. Kaushalya. 2016. [Online]. Available: https://medium.com/@jayanga/how-to-configure-checkstyle-and-findbugs-plugins-to-intellij-idea-for-wso2-products-c5f4bbe9673a [Accessed: 25-July-2019]
[6] Guru99 Team, 'What is Software Testing?'. [Online]. Available: https://www.guru99.com/software-testing-introduction-importance.html[Accessed: 28-July-2019]
[7] Baeldung Team, 'Intro to JaCoCo', 2018. [Online]. Available: https://www.baeldung.com/jacoco[Accessed: 28-July-2019]
[8] M. Fowler, 'Refactoring: Improving the Design of Existing Code', 1999 . [Online]. Available: https://www.csie.ntu.edu.tw/~r95004/Refactoring_improving_the_design_of_existing_code.pdf[Accessed: 29-July-2019]