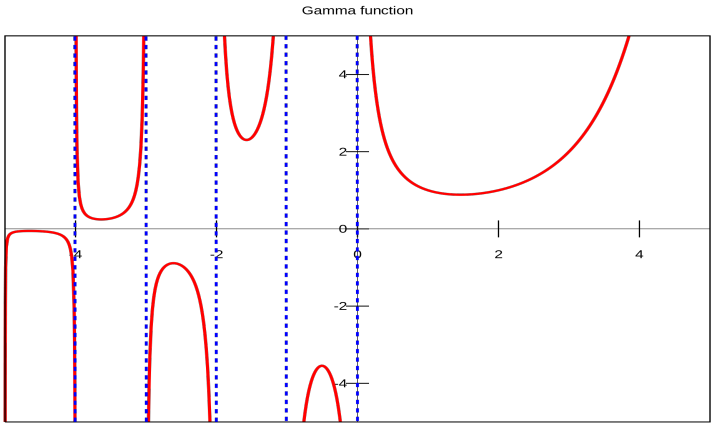
40071296 - DELIVERABLE3

A PREPRINT

Shivani Jindal
40071296
SOEN 6011
Software Engineering Processes
Department of Software Engineering and Computer Science
Concordia University
Montreal
https://github.com/ShivaniJindal/SOEN-6011-40071296

July 31, 2019

$$\Gamma(z) = \int_0^{\infty} e^{-t} t^{z-1} dt$$



CODE REVIEW

Code review is a widely-used technique for improving software quality by human inspection. Code review can detect many kinds of problems in code, but as a starter, this reading talked about these general principles of good code:

- Comments where needed
- Avoid magic numbers
- One purpose for each variable
- Use good names
- No global variables
- Easy to understand
- Ready to change
- Return results, don't print them
- Use white-space for readability

Results for source code reviewed of F6, coded by Girish Kumar Kadapa.

1. Formatting:-

Q1.1: Are the spaces and line breaks are at right place?
Indentation is good, no extra spaces and line breaks are there.It is readable. But due to nested if-else it becomes at some places hard to understand.
Q1.2: Is he using tabs or spaces?
Spaces has been used.
Q1.3: How are the curly braces laid out?
It is according to java standards. They are starting just one space next to condition and ending just next to the start of another condition. However, closing of braces is not properly spaced so in case of nested if-else it becomes harder to find ending of the braces when there is no start of another condition.

2. Style:-

Q2.1: Are the variables/parameters declared as final?
None of the parameter is declared final which is not required as we are computing mathematical function in which values of each sub-function will get updated depending upon inputs, it cannot be fixed whereas there are some constants such as denominator, sign which should be declared as final in order to avoid mathematical error.
Q2.2: Are method variables defined close to the code where they're

used or at the start of the method?
Variables which are required to calculate main function, F6 are declared at class level but are used only at function level. Other all variables are declared at method level only where they are needed.

3. Naming:-

Q3.1: Do the field/constant/variable/parameters/class names conform to standards?
Yes, names of variables/classes/methods all are adherent to Java coding rules, standards and guidelines.

4. Design:-

Q4.1: Does the code follow SOLID principles ?
S:- Single Responsibility Principle
Yes, code follows this principle as for all sub-functions different methods has been made which are their own sufficient to make calculations.
O:- Open/Closed Principle
Code does not follow this principle.
L:- Liskov Substitution Principle
Yes, code follows this principle as output of one function can sub substitute as input for another function.
I:- Interface segregation principle
No, here that principle is not required as it is the code of only one mathematical function for which all the sub-functions coded are required to evaluate and there is nothing like option or additional

functionality.
D:- Dependency inversion principle
Yes, code do satisfy this principle.
Q4.2: Do I understand what the code does?
Yes, it is fully understandable as per programmer perspective as well as per user as interface is very user-friendly.
Q4.3: Does the code function as I expect it to?
Yes, it computes accurate values for given mathematical function.
Q4.4: Does this code fulfill regulatory requirements?
Yes, it fulfills all the listed requirements in Deliverable 1.

5. Readability and Maintainability :-

Q5.1: Do the names (of fields, variables, parameters, methods and classes) actually reflect the thing they represent?
Class name is not reflect the function for which code is.
Function in which all sub-functions are called to calculate the given function named as function 6 which is very vague as the coder who is not part of this course will never know what function 6 stands for.
Other two method names are suitable to their role and all variables too.
Q5.2: Can I understand what the code does by reading it?
Yes, it is very easy to understand as code has complementary inline comments which are making things easy to understand and read.
Q5.3: Are the exception error messages understandable?
Yes they are but general exception is thrown not specific according to the code.

Results for test code reviewed of F7, coded by Jasmine Kalra.

Q 1: Can I understand what the tests do?
Tests for Main file are clear written in Main.test file. However, the ones written for Calculate.java are misleading as the inline comments are referring to Main.java but as per the file names, it states they are belong to Calculate.java file. In addition to this, the names of test cases in CalculateTest.java file are very confusing, multiple test cases for one function does not give clear categorization of why those have been made and what is their purpose.

Q 2: Do the tests cover a good subset of cases?
Yes, they do cover most of them as per the visibility and coverage, but not all.

Q 3: Do they cover exceptional cases?
Yes, for every possible mathematical real number code has been tested against for mathematical function.

Q 4: Are there cases that haven't been considered?
Yes, there are as most of the test cases are focused on negative integers, there is no test case which is testing combination of positive and negative real numbers.

Q 5: Are confusing sections of code either documented, commented, or covered by understandable tests (according to team

preference)?
No, there is very poor inline documentation of code in terms of comments. Purpose of test cases and what they are testing is very hard to understand.

Q 6: Does tests covered all stated requirements?
Yes, it do covers.

Q 7: Is the test testing only one code unit at a time?
Yes, each test is only for specific functionality of source code.

Q 8: Are the test cases dependent on each other?
No, they are independent.

Q 9: Do the test cases names conform to standards?
Names are written according to standards but names themselves are not as per standards as they are confusing, same name used mostly for all test cases just with the difference of count.

Q 10: Do the most appropriate assertion methods are used ?
Yes.

References

[1] Lokesh Gupta, JUnit Best Practices Guide https://howtodoinjava.com/best-practices/unit-testing-best-practices-junit-reference-guide/

[2] Reading 4: Code Review http://web.mit.edu/6.005/www/fa15/classes/04-code-review/

[3] What to look for in a Code Review *Posted on July 23, 2015 by Trisha Gee* https://blog.jetbrains.com/upsource/2015/07/23/what-to-look-for-in-a-code-review/