

# Deliverable-3

Sakshi Nagpal- <https://github.com/nagpalsakshi110/SOEN6011>

July 2019

**Code Repository:** <https://github.com/Elhamnf/SOEN6011-1>

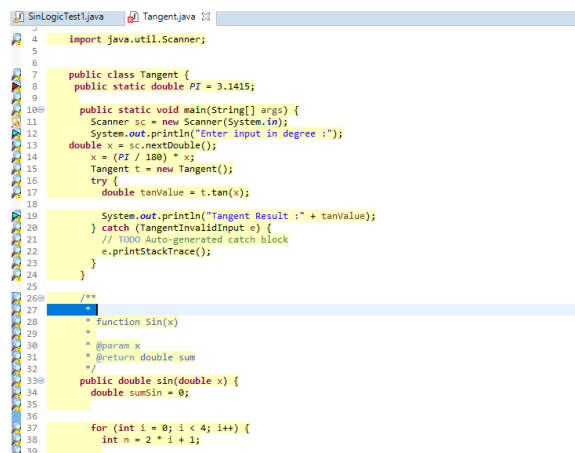
## 1 Problem-5

In this document we will be reviewing the **Trigonometric Tangent function**. The function's source code will be reviewed on the basis of set published standard practices and guidelines and we have also used Eclipse's static code analysis plugins like Checkstyle and PMD. The constraints that are used for reviewing are mentioned below.

- **Naming Convention:** As per this constraint we will check the programming style followed by the programmer. This constraint will ensure the block level scopes, the naming of variables and methods.
- We will also be reviewing the code by **Programming Principles** which will ensure whether the programmer is using proper programming principles like making instances of classes where required, making methods performing only single function.
- **Readability** a major constraint that ensures that the code is written as if it is understandable by a person with no knowledge of the code for which we will check the **Code layout**. To check the code layout we have used checkStyle. In which we will check the code on the basis of the principles of coding like white space, ensuring the clog braces are at same level as the opening brace.
- We will also check the **Commenting** done in the code ensuring this constraint will make the code **reusable and maintainable**. If the additional information is provided useful and readable enough that a new user is able to understand the code.
- Last but not the least we will check the **Code Quality** and for this we will check the structure of the code and the complexity of the code.

**Code Review:** The reviewing process was carried out both manually and using the IDE plugins mentioned above.

- The image below shows that the program **doesn't follow the set standards for programming**.
- The code also **doesn't follow the rules for indentation** the spacing should be of 2 characters for the methods but in the code the spacing is of 4 characters.
- The **naming conventions are also not followed** in naming some variables like for declaration of pi the declaration is done in upper case.
- The code written is **very much readable**. The comments give a clarity of the functions and the methods.



```

4 import java.util.Scanner;
5
6
7 public class Tangent {
8     public static double PI = 3.1415;
9
10    public static void main(String[] args) {
11        Scanner sc = new Scanner(System.in);
12        System.out.println("Enter input in degree :");
13        double x = sc.nextDouble();
14        x = (PI / 180) * x;
15        Tangent t = new Tangent();
16        try {
17            double tanValue = t.tan(x);
18
19            System.out.println("Tangent Result : " + tanValue);
20        } catch (TangentInvalidInput e) {
21            // 1000 Auto-generated catch block
22            e.printStackTrace();
23        }
24    }
25
26    /*
27     * function Sin(x)
28     *
29     * @param x
30     * @return double sum
31     */
32    public double sin(double x) {
33        double sumSin = 0;
34
35        for (int i = 0; i < 4; i++) {
36            int n = 2 * i + 1;
37
38        }
39    }

```

Figure 1: Source code review

- The code **follows the coding principles** like having less than 72 characters in a line.
- It was **easy to understand** the code. The commenting and the javadocs made things very clear about the methods and classes.
- There are certain violations of **Naming conventions**.

**In conclusion** there are some violations of Java Programming guidelines like naming convention, Indentation and code layout. The code overall is readable, well-structured understandable and the programmer took care of the complexity of the code and used variables and data structures where required. So, for the new programmer who doesn't have knowledge about he function it was easier to Scan through the code and understand the basic functioning of tangent function.

## 2 Problem-7

Code Repository: <https://github.com/jonofre21/SOEN6011>

In this section we are reviewing the test cases of the Trigonometric function sine. The programmer has used the JUnit for designing the test cases. The test cases are clear and easily understandable. The code was cloned from the repository mentioned above and then run on my system. The following images shows that all the test cases were success full.

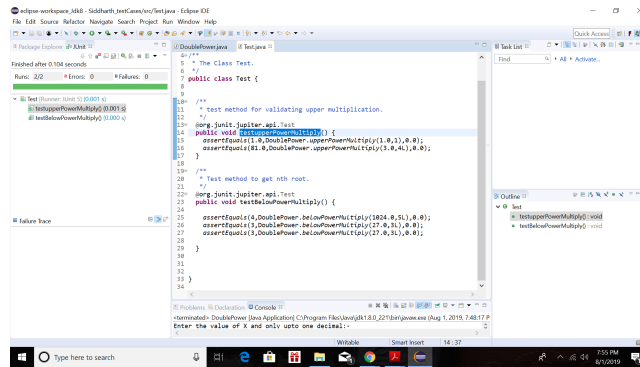


Figure 2: Test case Review

There could have been more test cases for ensuring the proper functionality of sine function. There were in total 5 test cases that checked all the methods implemented in the code but 3 that check the function of sine. Overall, the test cases were well structured and followed the General Standards for programming .

## 3 References:

1. Thongtanunam, P, Tantithamthavorn, C., Kula, R. G., Yoshida, N., Iida, H., Matsumoto, K. (2015). Who should review my code? A file location-based code-reviewer recommendation approach for Modern Code Review. 2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Re engineering (SANER). doi:10.1109/saner.2015.7081824
2. <https://www.cis.gvsu.edu/java-coding-style-guide/>