SOEN 6011 : SOFTWARE ENGINEERING PROCESSES

SUMMER 2022

Final Project - Function 7: x^y

Submitted To: Pankaj Kamthan

Submitted By:

Anushka Sharma

40159259

Contents

PROBLEM 1 - F7 : x^y

Repository address : https://github.com/anush97/SOEN6011

Definition of x^y

[1] Exponentiation is a mathematical operation, denoted as x^y, If y is a positive integer and x is any real number, then x^y corresponds to repeated multiplication.

$$x^y = x{*}x{*}x{*}.....{*}x \text{ of y times}$$

The expression can be called as "x raised to the power of y," "x to the power of y," or simply "x to the y." Here, x is the base and y is the exponent or the power.

Domain

[2] All the real numbers from -infinite to +infinite.(-∞ to ∞)

$$(x, y) \in^2 : (x \geq 0 \land y \neq 0) \lor x > 0$$

Co-domain

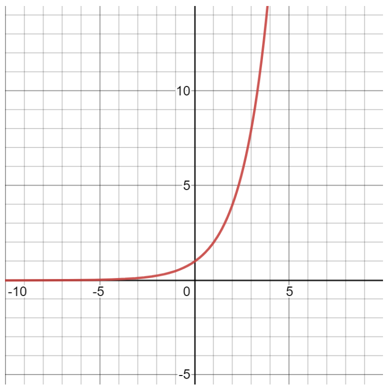
A set of all positive real numbers from zero to infinite (0 to ∞) is known as the Exponentiation function

1. When y is a non-negative integer, the domain is all real numbers: (- ∞,∞)
2. When y is a negative integer, the domain is all real numbers excluding zero ((- ∞, 0) $\cup$ (0,∞))
3. When y is an irrational number and y > 0, the domain is all non-negative real numbers and when y is an irrational number and y < 0, the domain is all positive real numbers
4. The co-domain of the function is [- ∞,∞] and can be indeterminate

Graph Characteristics

[2][3] The **rate of change** increases (or decreases) across the graph in an exponential graph.

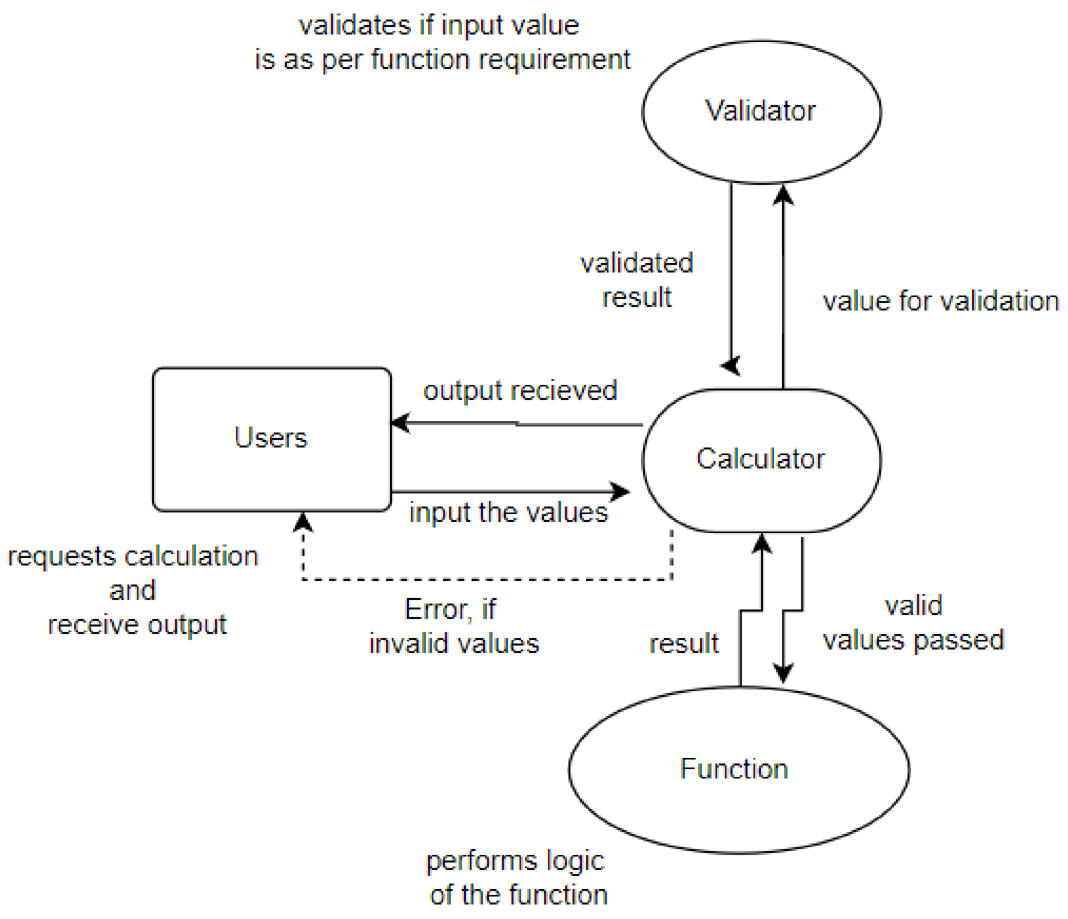
- The exponential graph crosses the y-axis at (0,1).
- The exponential graph increases, when x > 1.
- The exponential graph decreases, when 0 < x < 1.
- The exponential graph is asymptotic to the x-axis - gets very, very close to the x-axis but, in this case, does not touch it or cross it.



Graph crosses the y-axis at (0,1)

b) Context of Use Model

The users of the calculator shall be using it to calculate the result of function on two numbers. These numbers shall be an integer or decimal,so the valid inputs are the digits *0-9* and the decimal points.The calculator should return the result or a message that indicates why it was unable to do so.



PROBLEM 2 - F7 : x^y

Requirements

[4][5]
This section describes the requirements to implement the function 7 x^y.

Assumption : The transcendental function x^y will be accurate and accepts input which comprises of rational and irrational numbers.The exponential power will always be positive.

Requirement Id : F7-R1

Overview	X(0) to the power of Y(0)
Version	1.0
Description	If the user gives an input for X as Zero and input for Y as Zero. The function may return the 1 as output.
Owner	Anushka Sharma
Priority	High
Type	Functional
Difficulty	Medium
Verification Method	F7_TC

Requirement Id : F7-R2

Overview	X(0) to the power of Y(Real Number)
Version	1.0
Description	If the user gives an input for X as zero and input for Y as any Real Number. The function may return zero as output.
Owner	Anushka Sharma
Priority	High
Type	Functional
Difficulty	Medium
Verification Method	F7_TC

Requirement Id : F7-R3

Overview	X(Positive Number) to the power of Y(0)
Version	1.0
Description	If the user gives an input for X of any positive number and input for Y as Zero.The function may return 1 as the output.
Owner	Anushka Sharma
Priority	High
Type	Functional
Difficulty	Medium
Verification Method	F7_TC

Requirement Id : F7-R4

Overview	X(Negative Number) to the power of Y (0)
Version	1.0
Description	If the user gives an input for X of any negative number and input for Y as Zero.The function may return 1 as the output.
Owner	Anushka Sharma
Priority	High
Type	Functional
Difficulty	Medium
Verification Method	F7_TC

Requirement Id : F7-R5

Overview	X(Positive Number) to the power of Y(1)
Version	1.0
Description	If the user gives an input for X as any positive number and input for Y as 1. The function may return X as the output.
Owner	Anushka Sharma
Priority	High
Type	Functional
Difficulty	Medium
Verification Method	F7_TC

Requirement Id : F7-R6

Overview	X(Positive Number) to the power of Y(Positive Number)
Version	1.0
Description	If the user gives an input for X as any positive number and input for Y as positive number. The function may return positive number as the output.
Owner	Anushka Sharma
Priority	High
Type	Functional
Difficulty	Medium
Verification Method	F7_TC

Requirement Id : F7-R7

Overview	X(Negative Number) to the power of Y(Positive Even Number)
Version	1.0
Description	If the user gives an input for X as any Negative number and input for Y as positive Even number. The function may return positive number as the output.
Owner	Anushka Sharma
Priority	High
Type	Functional
Difficulty	Medium
Verification Method	F7_TC

Requirement Id : F7-R8

Overview	X(Negative Number) to the power of Y(Positive Odd Number)
Version	1.0
Description	If the user gives an input for X as any negative number and input for Y as positive odd number. The function may return negative number as the output.
Owner	Anushka Sharma
Priority	High
Type	Functional
Difficulty	Medium
Verification Method	F7_TC

PROBLEM 3 - F7 : x^y

Algorithm : Montgomery's Ladder Technique[]

- Montgomerym's ladder technique addresses defence against side-channel attacks for exponentiation computation.

  The algorithm prevents the recovery of the exponent involved in the computation which could possibly benefit an attacker
- The algorithm performs a fixed sequence of operations (up to log n): a multiplication and squaring takes place for each bit in the exponent, regardless of the bit's specific value.

Advantages	**Disadvantages**
It addresses the concern of MIM(Middle Man attack) observing the sequence of squaring and multiplications can (partially) recover the exponent involved in the computation.	Cache timing attacks are not yet protected and memory access latency might still be observable to an attacker

Algorithm : Taylor series

Taylor series is a representation of a function as an infinite sum of terms that are calculated from the values of the function's derivatives at a single point.

$$x^y = e^{y \ln x} \tag{1}$$

1 evaluation of x^y. Here, e is a mathematical constant approximately equal to 2.71828

$$e^x = 1 + x/1! + x^2/2! + x^3/3! + ... \tag{2}$$

2 express e^x using Taylor Series

$$e^x = 1 + (x/1)(1 + (x/2)(1 + (x/3)(........))) \tag{3}$$

3 The series 2 can be re-written as above

$$log(1 + x) = x - x^2/2 + x^3/3 - ... \tag{4}$$

4 express ln x using Taylor Series

Advantages	**Disadvantages**
Very useful for derivations	Successive terms get very complex and hard to derive
Can be used to get theoretical error bounds	Truncation error tends to grow rapidly away from expansion point
Power series can be inverted to yield the inverse function	Almost always not as efficient as curve fitting or direct approximation

Algorithm 1 Iterative algorithm to calculate x^y

```
function power_function_iterative(x,y)
in: double number x,y
out: double number result
$result \leftarrow 1$

$temp \leftarrow 1$
for $temp \leq y$ do

$result \leftarrow result * x$

$temp \leftarrow temp + 1$
end for

return result
```

The output is stored in result, which is initially set to 1. It is then looped from 1 to y, x number of times, incremented by one on each iteration and on each iteration we multiply result by x. At the end of the loop value of result is equal to x^y.

Algorithm 2 Recursive Divide and Conquer algorithm to calculate x^y

```
function power_function_recursive(x,y)
in: double number x,y
out: double number result
$power \leftarrow exponent\_helper(x, y)$
$result = power$
return result
```

A helper function called exponent_helper is defined which calculates x^y. In the base case when $y = 0$, we return 1, otherwise when $y = 1$ we return x. When x is even we recurse on $x = x * x$ and $y = y/2$. In case when x is odd we recurse on $x = x * x$ and $y = (y - 1)/2$. In the end, in our main function power_function_recursive we multiply the result of exponent_helper to the value of a and return our result.

Advantages and Disadvantages

Algorithm 1:

Advantages:
1. In terms of space complexity, iterative algorithms don't suffer from stack overflow because all operations are done on the heap.
2. They are easy to comprehend by humans and have better readability.
Disadvantage:
1. The time complexity of the iterative algorithm is $O(n)$, hence it is not very efficient for larger inputs in terms of time.
2. Proper terminating condition for loop is important or else it might lead to infinite looping.

Algorithm 2:

Advantages:
1. The time complexity of the recursive algorithm is $O(logn)$. This recursive algorithm is optimized (tail

```
1. function exponent_helper(x,y)
in:  double number x, y
out:  double number sum
2. if
$x < 0$ then
3.       $x \leftarrow 1.0/x$
4.       $y \leftarrow -y$
5.       return $exponent\_helper(x, y)$
6. else if
$y = 0$ then
7.       return 1.0
8. else if $y = 1$ then
9.       return x
10. else if
$y \mod 2 = 0$ then
11.       $y \leftarrow y * y$
12.       $y \leftarrow y/2$
13.       return $exponent\_helper(x, y)$
14. else
15.       $x \leftarrow x * x$
16.       $y \leftarrow y - 1$
17.       $y \leftarrow y/2$
18.       return $exponent\_helper(x, y)$
19. end if
```

recursive) so that we don't get stack overflow error and it handles large inputs better.
2. Recursion has higher maintainability than looping. Handling the base case properly requires little or no modifications.
Disadvantages:
1. Due to the continuous allocation of memory space leading to a stack overflow, efficiency is significantly affected.
2. It is difficult to comprehend. A certain level of expertise is required for understanding it vividly.

Conclusion

Recursive algorithm which is based on Divide and Conquer Algorithmic Strategy is preferred over iterative algorithm due to above reasonings.

Pseudocode Mind Map

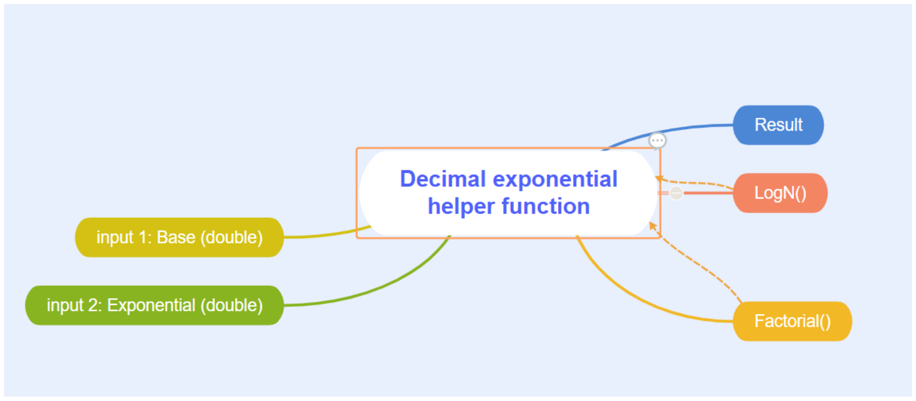

Result
Decimal exponential helper function
LogN()
input 1: Base (double)
input 2: Exponential (double)
Factorial()

Figure 1: **M**ind Map of pseudocode

PROBLEM 4 - F7 : x^y

Source code Review of F7

This section presents an overview of the source code of the Function 7 application (F7-Power Function) and the practices followed during the development.

Standard Programming Style used

Naming Conventions

The code satisfies to the google code style, and the method names and variable names in the code confirm to the standard of camel case nomenclature.

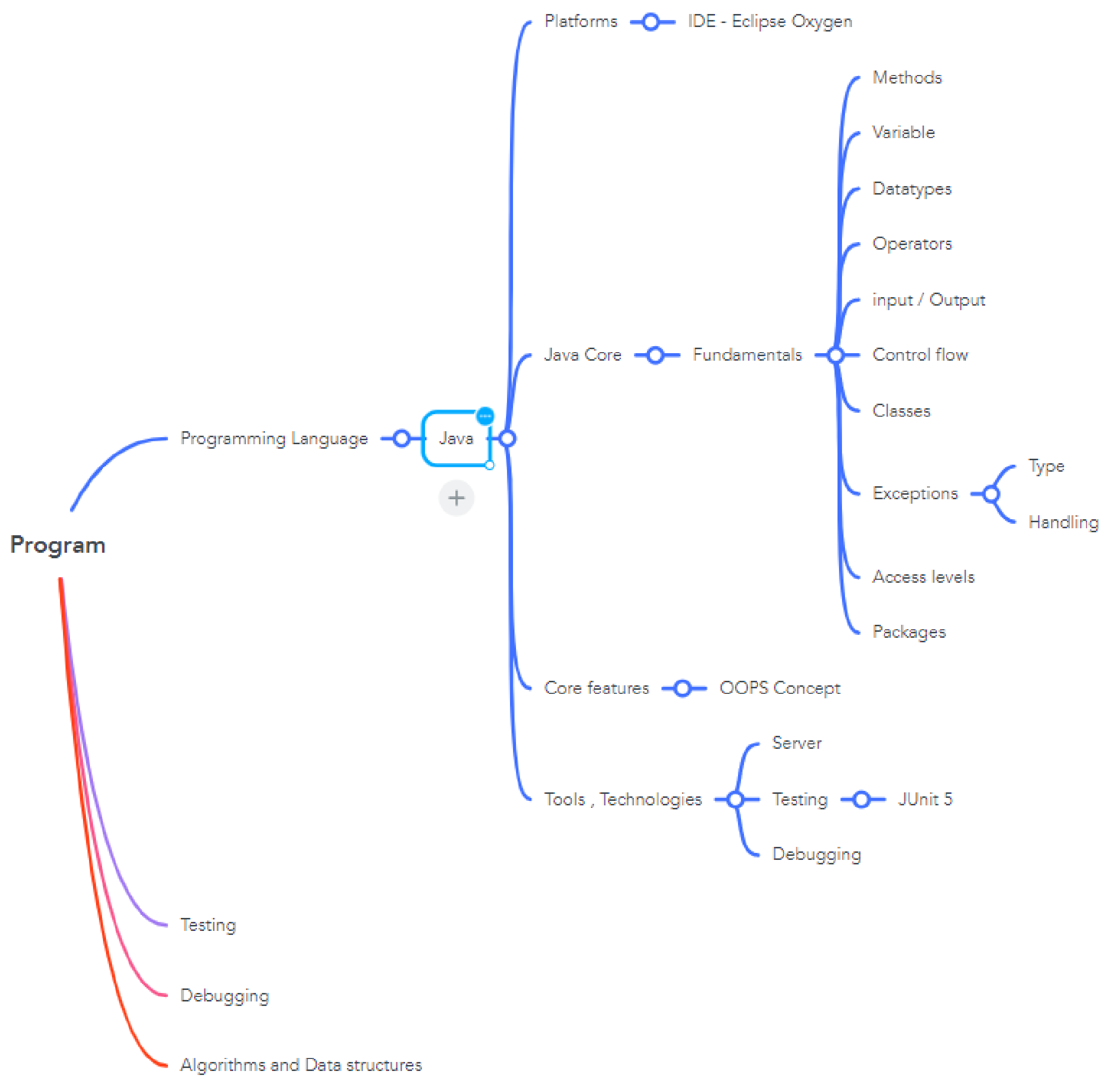


Figure 2: Programming style Mind Map

Working of Source Code

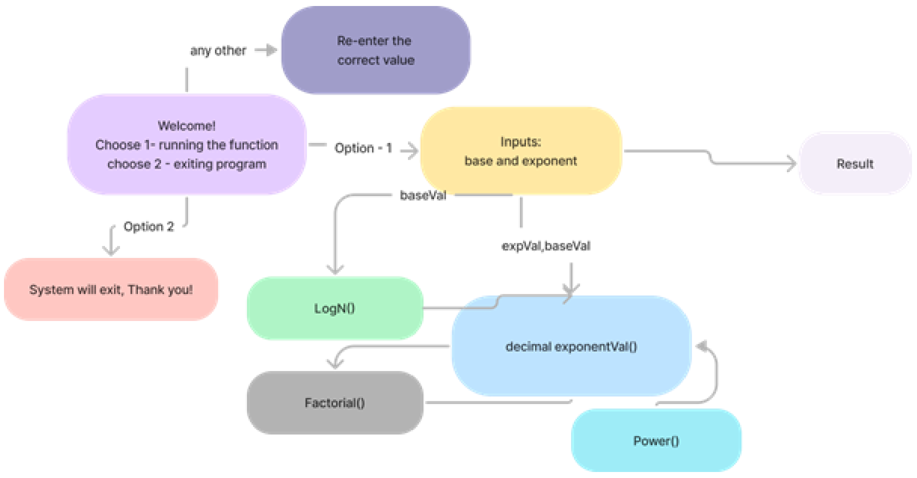
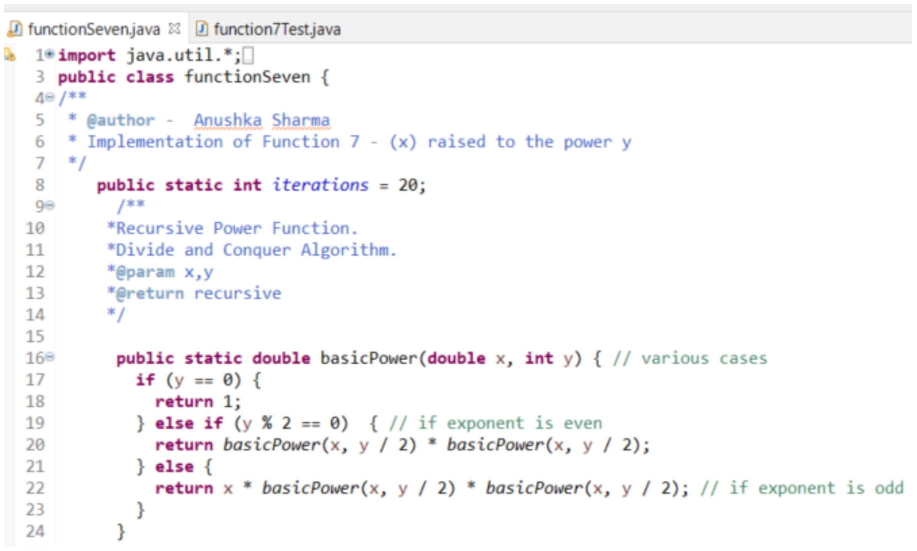

any other
Re-enter the correct value
Welcome! Choose 1- running the function choose 2 - exiting program
Option - 1
Inputs: base and exponent
Result
Option 2
baseVal
expVal,baseVal
System will exit, Thank you!
LogN()
decimal exponentVal()
Factorial()
Power()

Figure 3: Mindmap of code working

JavaDoc

Javadoc is added to facilitate the formation of highly readable documents and understanding of the code. They can define parameters, return type and authors like things for the methods of Classes and Objects.



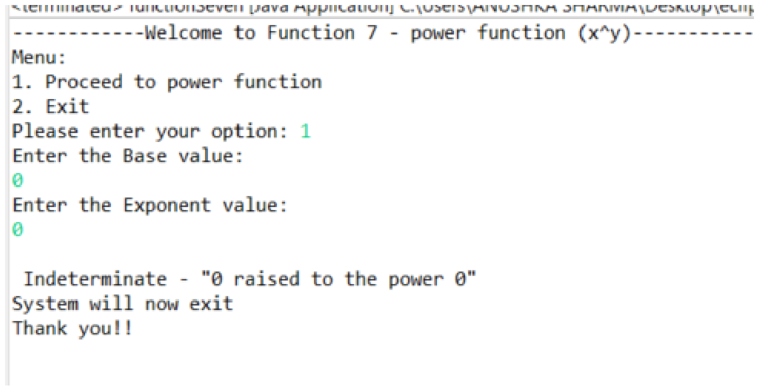
```
import java.util.*;
public class functionSeven {
/**
 * @author -  Anushka Sharma
 * Implementation of Function 7 - (x) raised to the power y
 */
    public static int iterations = 20;
      /**
     *Recursive Power Function.
     *Divide and Conquer Algorithm.
     *@param x,y
     *@return recursive
     */

      public static double basicPower(double x, int y) { // various cases
        if (y == 0) {
          return 1;
        } else if (y % 2 == 0)  { // if exponent is even
          return basicPower(x, y / 2) * basicPower(x, y / 2);
        } else {
          return x * basicPower(x, y / 2) * basicPower(x, y / 2); // if exponent is odd
        }
      }
```

Figure 4: Javadoc in java file

Error Handling

When an exception occurs, Its considered that the exception is thrown and it has to be caught. When an exception is thrown, it is possible to ”catch” the exception and prevent it from crashing the program. This is done with a try..catch block statement in the Function 7 as well. One can use many catch statements to handle various exceptions. My main function consists of a try and catch block that is responsible for checking for an exception of NumberFormatException which is caused when the input in any of the base or exponent value is considered not double value or a value that cannot be converted into double. In simplified form, the syntax for a try..catch statement can be:

```
try {
    statements-1
    ...
    numericInputCheck(inputDataString)
    ...
}
catch ( NumberFormatException exception) {
    statements-2
}
```



```
------------Welcome to Function 7 - power function (x^y)-----------
Menu:
1. Proceed to power function
2. Exit
Please enter your option: 1
Enter the Base value:
0
Enter the Exponent value:
0

 Indeterminate - "0 raised to the power 0"
System will now exit
Thank you!!
```

Figure 5: An Error message in the Function 7 that displays if the input is not accepted.

Debugger

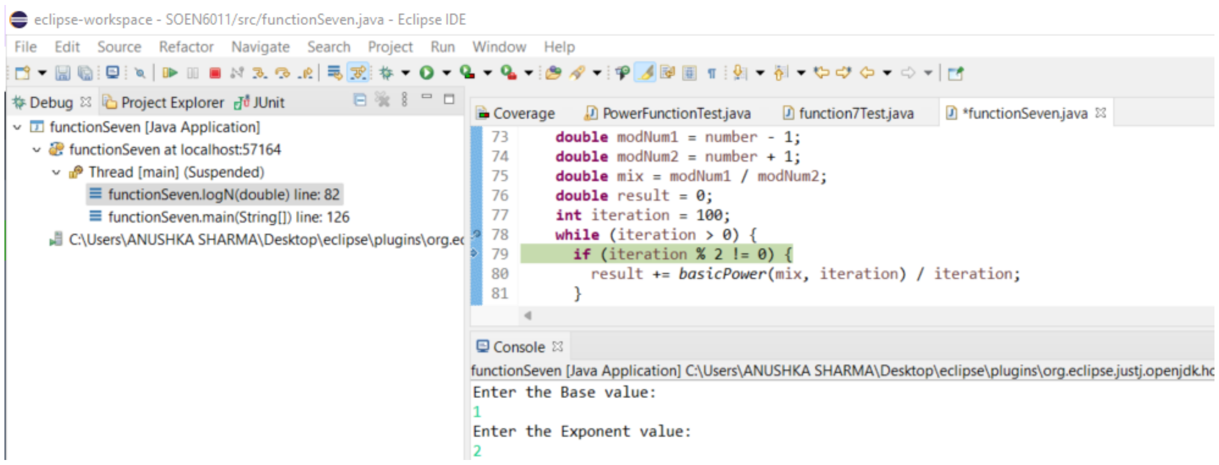
Eclipse has an in-built standard debugger allowing the program to open in a mode known as debug mode.Here, debugging can be performed step by step and we can also put break points so as to run just a section of the code.It offers multiple features like breakpoints , checkpoints and multiple views which enhance the experience of debugging.

Advantages

1. Can add any variable that one want to monitor to watch list and see it working.
2. Eclipse debugger allows to remote debug a process on any other machine .
3. The ability to execute code interactively—setting breakpoints, executing code line by line, and inspecting the value of variables and expressions.
4. The Eclipse Debugging Platform helps developers debug by providing buttons in the toolbar and key binding shortcuts to control program execution.

Disadvantages

1. Debugging with eclipse can be difficult if the execution of a particular function is time bound or if there is usage of thread or sleep statements within the file.
2. For Multi threading, we have to use interfaces that themselves have their short comings .


eclipse-workspace - SOEN6011/src/functionSeven.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Project Explorer
JUnit
Coverage
PowerFunctionTest.java
function7Test.java
*functionSeven.java
Console

```
functionSeven [Java Application]
  functionSeven at localhost:57164
    Thread [main] (Suspended)
      functionSeven.logN(double) line: 82
      functionSeven.main(String[]) line: 126
  C:\Users\ANUSHKA SHARMA\Desktop\eclipse\plugins\org.ec
```

```java
    double modNum1 = number - 1;
    double modNum2 = number + 1;
    double mix = modNum1 / modNum2;
    double result = 0;
    int iteration = 100;
    while (iteration > 0) {
      if (iteration % 2 != 0) {
        result += basicPower(mix, iteration) / iteration;
      }
```

```
functionSeven [Java Application] C:\Users\ANUSHKA SHARMA\Desktop\eclipse\plugins\org.eclipse.justj.openjdk.hc
Enter the Base value:
1
Enter the Exponent value:
2
```

Figure 6: Debugger of Eclipse.java

0.1 Achieved Quality Attributes

0.1.1 Space management

- Limited variables are created
- Functions have been created to avoid duplicates of functionalities
- Used efficient methodology for the execution.

0.1.2 Maintainability

- Common practices has been adapted within a group to avoid ambiguities.
- Useful comments added in the code where it is required.
- Avoided global scoping for common variables and functions.
- Refactored code once every member merged their code to the github branch.

0.1.3 Robustness

- Usage of exception for exceptional test cases.
- Narrowed the variable scope as far as possible in the code.
- Error handling done on the code.
- Usage of mutable variable over creating new variables.

0.1.4 Usable

- Simple console interface provided for user input.
- Error messages are given in wrong input.
- Success messages are given for results.
- Suggestions are given when user encountered any difficulties while using calculator.

0.1.5 Correctness

- Coding standards is followed.
- Proper testing practices has been done on the function assigned.
- JUnit Testing has implemented to check the correctness.

0.1.6 Efficiency

- The main focus on readability is given to the code.4
- The program takes not less than two or three nanoseconds.

0.2 Quality Check of Source Code

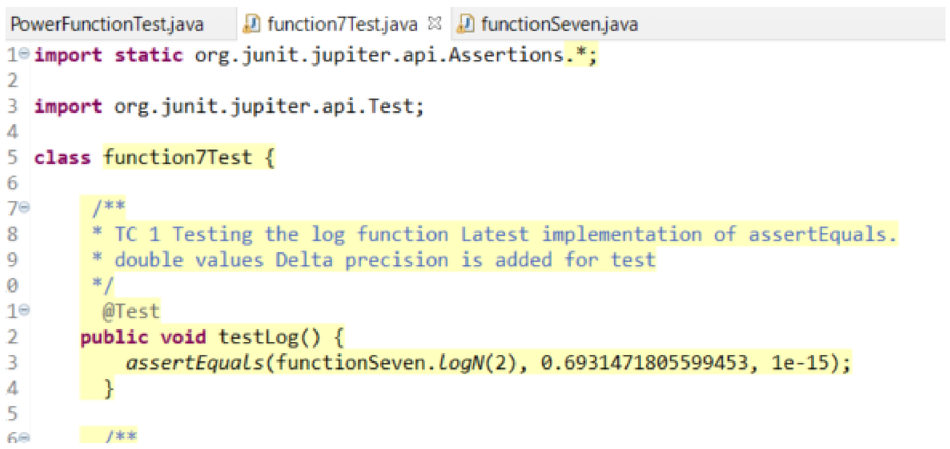
It's a programming tool that automates the process of inspecting Java code as per the guidelines and standards, saving the time and effort of doing so manually. It's works well for projects that want to enforce a coding standard.

Advantages

- The checkstyle is portable between different IDEs.
- It can be easily integrated as a build tool into your Software Configuration Management i.e. projects.
- Checkstyle is a stand-alone framework, integrating it with your other tools is considerably easier as does not disturb the flow of the code.

Disadvantages

- Checkstyle analyses one file at a time statistically.



```
import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.Test;

class function7Test {

    /**
     * TC 1 Testing the log function Latest implementation of assertEquals.
     * double values Delta precision is added for test
     */
     @Test
    public void testLog() {
        assertEquals(functionSeven.logN(2), 0.6931471805599453, 1e-15);
     }

     /**
```

Figure 7: Checkstyle of functionseven.java

PROBLEM 5 - F7 : x^y

Unit Test Case Description

The unit test cases implemented using **JUnit5** for Function 7 (F7-Power Function) which are traceable to requirements in section 2.

Test Case : F7_TC_1

Test Case ID	F7_TC_1
Requirement ID	F7-R1
Action	The user chooses ”1 ” to proceed with the function, then input base value followed by exponential value and presses Enter.
Input(s)	base = 0.0, exponent = 0.0
Expected Output	Indeterminate - ”0 raised to the power 0”
Actual Output	Indeterminate - ”0 raised to the power 0”
Test Result	Success

Test Case : F7_TC_2

Test Case ID	F7_TC_2
Requirement ID	F7-R2
Action	The user chooses ”1 ” to proceed with the function, then input base value followed by exponential value and presses Enter.
Input(s)	base = 0.0, exponent = 6.0
Expected Output	0.0
Actual Output	0.0
Test Result	Success

Test Case : F7_TC_3

Test Case ID	F7_TC_3
Requirement ID	F7-R3
Action	The user chooses "1 " to proceed with the function, then input base value followed by exponential value and presses Enter.
Input(s)	base = 37.0, exponent = 0.0
Expected Output	1.0
Actual Output	1.0
Test Result	Success

Test Case : F7_TC_4

Test Case ID	F7_TC_4
Requirement ID	F7-R4
Action	The user chooses "1 " to proceed with the function, then input base value followed by exponential value and presses Enter.
Input(s)	base = -9.0, exponent = 0.0
Expected Output	1.0
Actual Output	1.0
Test Result	Success

Test Case : F7_TC_5

Test Case ID	F7_TC_5
Requirement ID	F7-R5
Action	The user chooses to proceed with the function , then input base value followed by exponen
Input(s)	base = 7.0, exponent = 1.0
Expected Output	7.0
Actual Output	7.0
Test Result	Success

Test Case : F7_TC_6

Test Case ID	F7_TC_6
Requirement ID	F7-R6
Action	The user chooses ”1 ” to proceed with the function, then input base value followed by exponential value and presses Enter.
Input(s)	base = 5, exponent = 9
Expected Output	1953125.0
Actual Output	1953125.0
Test Result	Success

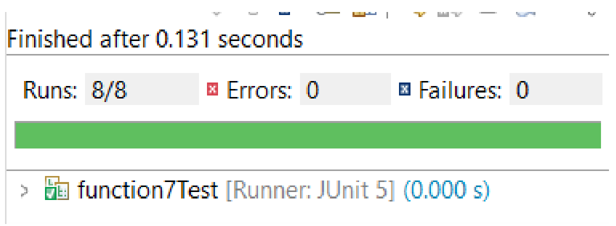
Test Case : F7_TC_7

Test Case ID	F7_TC_7
Requirement ID	F7-R6
Action	The user chooses ”1 ” to proceed with the function, then input base value followed by exponential value and presses Enter.
Input(s)	base = -3, exponent = 4.4
Expected Output	125.916
Actual Output	125.916
Test Result	Success

Test Case : F7_TC_8

Test Case ID	F7_TC_8
Requirement ID	F7-R6
Action	The user chooses ”1 ” to proceed with the function, then input base value followed by exponential value and presses Enter.
Input(s)	base = -9, exponent = 3
Expected Output	-729
Actual Output	-729
Test Result	Success

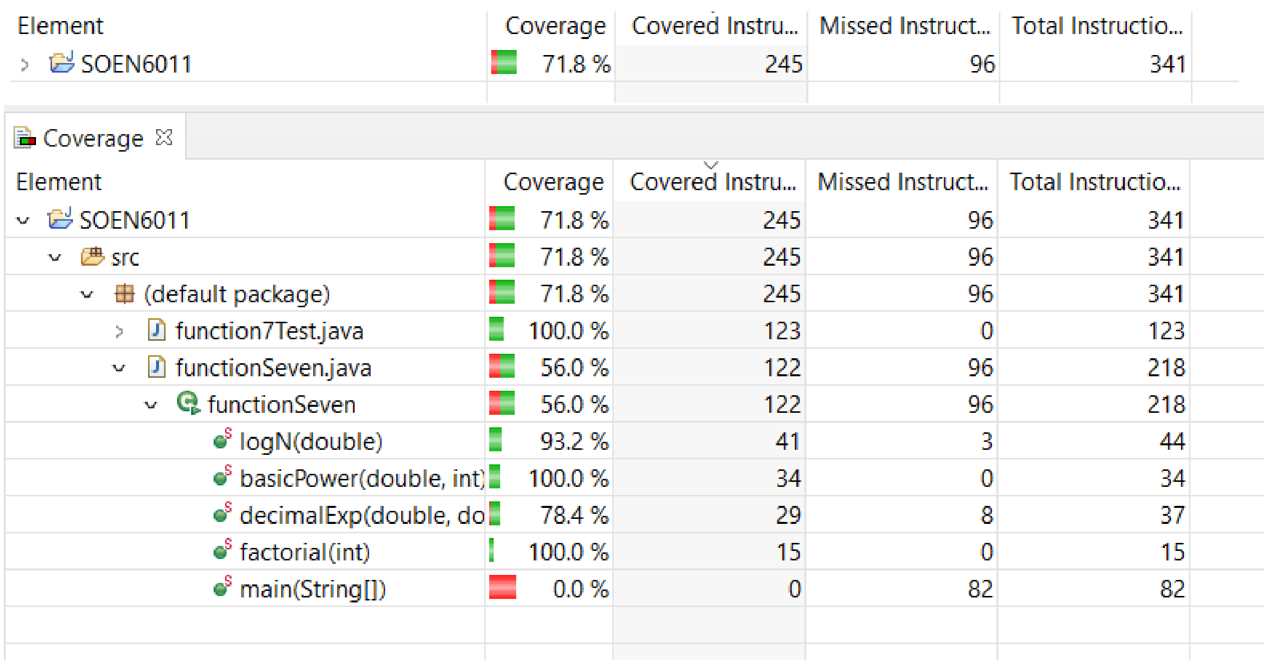
Hence all the requirements were satisfied and all the test cases passed. Unit Test case report are as follows:


Finished after 0.131 seconds

Runs: 8/8 Errors: 0 Failures: 0

> function7Test [Runner: JUnit 5] (0.000 s)

Figure 8: Test case result of function F7 : x^y using Junit4



Element	Coverage	Covered Instru...	Missed Instruct...	Total Instructio...
> SOEN6011	71.8 %	245	96	341

Coverage

Element	Coverage	Covered Instru...	Missed Instruct...	Total Instructio...
SOEN6011	71.8 %	245	96	341
src	71.8 %	245	96	341
(default package)	71.8 %	245	96	341
> function7Test.java	100.0 %	123	0	123
functionSeven.java	56.0 %	122	96	218
functionSeven	56.0 %	122	96	218
logN(double)	93.2 %	41	3	44
basicPower(double, int)	100.0 %	34	0	34
decimalExp(double, do	78.4 %	29	8	37
factorial(int)	100.0 %	15	0	15
main(String[])	0.0 %	0	82	82

Figure 9: Coverage of JUnit testing of function F7 : x^y

References

[1] Nykamp DQ: Basic rules for exponentiation
`https://mathbitsnotebook.com/Algebra2/Exponential/EXExpFunctions.html`

[2] MathBits Teacher: Exponential Functions,
`https://mathbitsnotebook.com/Algebra2/Exponential/EXExpFunctions.html`

[3] WolframAlpha: Domain of exponentiation function,
`https://www.wolframalpha.com/input/?i=x%5Ey`

[4] ReqView : Nykamp DQ: Requirements Specification Templates
`https://www.reqview.com/doc/iso-iec-ieee-29148-templates`

[5] 29148-2018-ISO/IEC/IEEE International Standard-Systems and software engineering-Life cycle processes-Requirements engineering,
`https://standards.ieee.org/standard/29148-2018.html`

[6] CheckStyle. Eclipse Checkstyle Plugin. 2019.
`https://checkstyle.org/eclipse-cs`

[7] Mike Spivey. ”The fuzz Manual” Manual and software copyright . J. M. Spivey 1988, 1992, 2000