UNIVERSITÉ Concordia UNIVERSITY

SOEN 6011 : Software Engineering Processes

Project Report
Function 4 : $\Gamma(x)$ Gamma Function

Submitted To : Dr. Pankaj Kamthan

Dhruviben Jigeshkumar Modi (40166396)

August 05, 2022

1 Problem 1

1.1 Introduction

F4: $\Gamma(x)$ which is known as Gamma function, is a generalization of the factorial function to non-integer numbers. The gamma function is defined as the improper integral of another function. It is denoted by a capital letter gamma from the Greek alphabet.

Let define f be the Gamma Function from A to B,therefore A is the domain and B is the co-domain of the Gamma Function.

(A)Domain of function: $\{x \in R^{+} : x > 0\}$ includes all complex numbers and the positive integer(except zero and negative integers).

(B)Co-domain of function: $(0, \infty]$

When n in A is a positive integer, then the gamma function is related to the factorial function $\Gamma(n) = (n-1)!$

When n in A for complex numbers with a positive real part, then the $\Gamma(n) = \int_{0}^{\infty} x^{n-1} e^{-x} dx$.

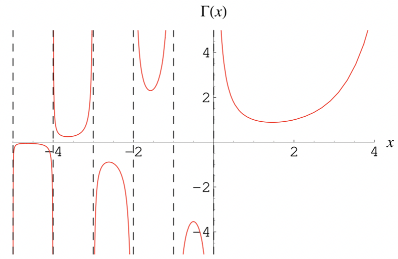


Figure 1.1: Gamma Function

1.2 Properties

[1.] $n \to 0^{+}, \Gamma(n) \to +\infty$

[2.] Extreme property: a$\in$ **R**, $\lim_{n \to \infty} \frac{\Gamma(n+a)}{\Gamma(n)n^{a}} = 1$,

[3.] Density function: $\Gamma\left(n + \frac{1}{2}\right) = \dfrac{(2n)!\sqrt{\pi}}{n!4^{n}}$

[4.] Recursive property: $\Gamma(n+1) = n * \Gamma(n)$

[5.] Euler's Reflection formula: $\Gamma(n) * \Gamma(1-n) = \frac{\pi}{\sin \pi n}$

[6.] Legendre Duplication formula: $\Gamma(n) * \Gamma(n+\frac{1}{2}) = 2^{1-2n} * \sqrt{\pi} * \Gamma(2n)$

[7.] For $\lambda > 0$, $\int_0^\infty x^{n-1} e^{-\lambda x} d = \dfrac{\Gamma(n)}{\lambda^n}$

1.3 Particular Values

(1) $\Gamma(1) = 0! = 1$
(2) $\Gamma(1/2) = \sqrt{\pi}$

1.4 Context Use Model

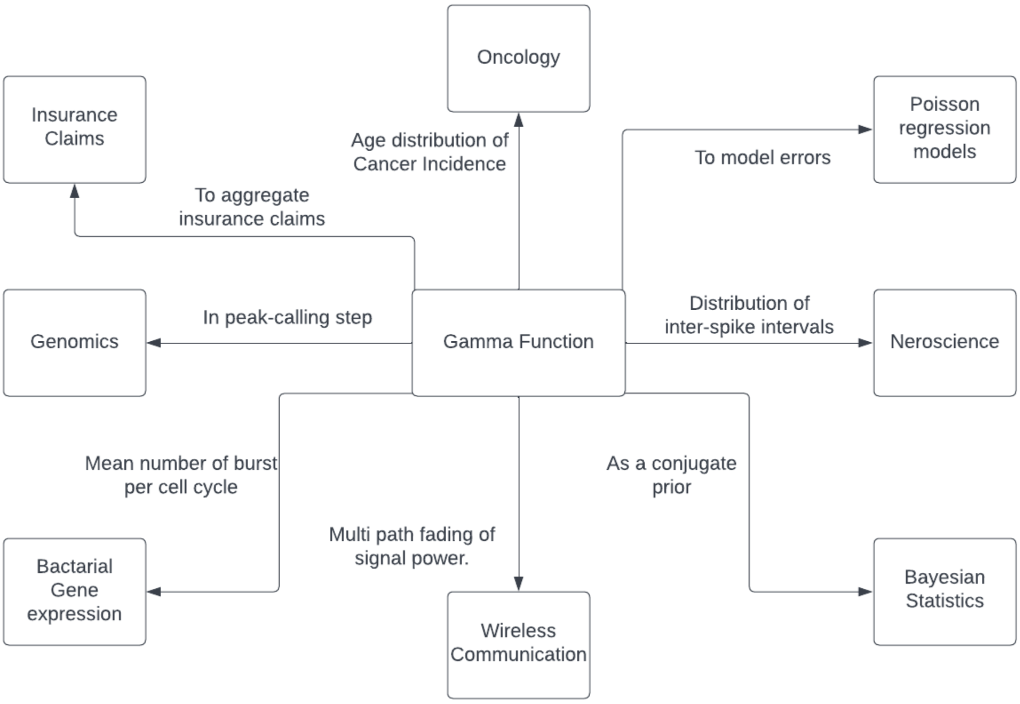

Oncology
Insurance Claims
Poisson regression models
Age distribution of Cancer Incidence
To model errors
To aggregate insurance claims
Genomics
In peak-calling step
Gamma Function
Distribution of inter-spike intervals
Neroscience
Mean number of burst per cell cycle
As a conjugate prior
Multi path fading of signal power.
Bactarial Gene expression
Wireless Communication
Bayesian Statistics

Figure 1.2: Context use model

2 Problem 2

2.1 Explicit Assumption

1. n is positive real number $n \in R^+$
2. For n $\in Z^+$, it is easier to compute $\Gamma(n)$

2.2 Requirements and corresponding properties

F4 - $\Gamma(x)$

(1) First Requirement : The Gamma Function $\Gamma(x)$ requires x as its variable input in order to proceed.

- ID : FR1
- Version number: 1.0
- Priority: High
- Rationale: x
- Difficulty: Easy.
- Type: Functional requirement

(2) Second Requirement : If the user provides a legitimate input in domain of R^+, the Gamma function's output must be exact and correct.

- ID : FR2
- Version number: 1.0
- Priority: High
- Rationale: $x > 0$, The fundamental goal of the calculating system is to provide an accurate answer quickly, hence this criteria is essential.
- Difficulty: Difficult, selection of correct algorithm.
- Type: Functional requirement

(3) Third Requirement : When the user entered the parameter x, Gamma calculator will check the value of a parameter. If the entered value is zero or negative it displays the error message. If the parameter is in form of String then exception is thrown with an error message.

- ID : FR3
- Version number: 1.0
- Priority: High
- Rationale: For the gamma function, 0 and all the negative integers are not defined. For example $\Gamma(0) = \int_0^{\infty} x^{-1}e^{-x}dx$. It is not integrable. For small value of x, it appears to decay slowly, but for large values, it decays quite quickly. Non-Integral form is:

  $\lim_{a\to 0}\int_a^1 x^{-1}e^{-x}dx \geq \frac{1}{e}\lim_{a\to 0}\int_a^1 \frac{dx}{x} = \lim_{a\to 0} -log_a = \infty$

  Thus $\Gamma(0)$ is undefined, and hence it is also undefined for all the negative integers.
- Difficulty: Easy
- Type: Functional requirement

(4) Fourth Requirement : To avoid a stack overflow, the gamma function can be calculated if the inputs are positive integers using the tail recursion function.

- ID : FR4
- Version number: 1.0
- Priority: High
- Rationale: $\{ \forall n \in R^+ \mid \Gamma(n) = (n-1)! \}$
- Difficulty: Moderate
- Type: Functional Requirements

(5) Fifth Requirement : The output of the gamma function can be computed for the fractional value of the input parameter using Stirling's approximation for performing definite integration.

- ID : FR5
- Version number: 1.0
- Priority: High
- Rationale: $\{ \forall n \in R^+ \mid$ y $\Gamma n = \sqrt{2\cdot\pi\cdot n}\cdot(\frac{n}{e})^n \}$
- Difficulty: Difficult
- Type: Functional Requirements

(6) Sixth Requirement : The system must be maintainable and managable to achieve usability.

- ID : NFR1
- Version number: 1.0
- Priority: High
- Rationale: To make a system well-organized it is essential that each modules must be implemented separately, otherwise it is difficult to manage such complex system.
- Difficulty: Moderate
- Type: Non-Functional Requirement

(7) Seventh Requirement : The system portability.

- ID : NFR2
- Version number: 1.0
- Priority: High
- Rationale: Java's architectural neutral features give the system the flexibility to function on many system architectures.
- Difficulty: Moderate
- Type: Non-Functional Requirement

(8) Eighth Requirement : The system Usability.

- ID : NFR3
- Version number: 1.0
- Priority: High
- Rationale: Any user should be able to easily comprehend the system. For a novice user, error messages must be helpful.
- Difficulty: Easy
- Type: Non-Functional Requirement

3 Problem 3

3.1 Gamma function with Lanczos

Algorithm 1 Lanczos approximation for Gamma Function

```
Require: value: $x > 0$                                        ▷ $x \in Z^{+}$
Ensure: g is a constant that chosen randomly with the condition that $Re(z) > \frac{1}{2}$.
procedure LANCZOSGAMMA(x)
    if x then $< \frac{1}{2}$ then
        $\Gamma(x) = \frac{\pi}{\sin \pi x \Gamma(1-x)}$                ▷ Reflection formula
    end if
    g ← a small random integer
    P ← a convergence of nine to ten terms         ▷ double floating-point precision.
    for $i \leftarrow 1, P.length$ do
        $P_g(x) = P_0 + \frac{P_i}{x+i}$
    end for
    $\Gamma(x+1) = \sqrt{2\pi}\left(x+g+\frac{1}{2}\right)^{x+\frac{1}{2}} e^{-\left(x+g+\frac{1}{2}\right)} P_g(x)$
    return $\Gamma(x)$
end procedure

$result \leftarrow \Gamma(x)$
```

Lanczos approximation is a method for computing the gamma function numerically, It is a practical alternative to the Stirling's approximation for calculating the gamma function with fixed precision.. The Lanczos approximation extends the factorial technique, which will be used to calculate input smaller than 0.5, by using the reflection formula. It will apply a different formula for any other values. In order to compute the gamma function with standard single or double floating-point precision and select a fixed constant g, 9 to 10 terms of the series in an aggregate are required. These terms will all be used to determine the coefficients then incorporate it into the given formula $\Gamma(x+1) = \sqrt{2\pi}\left(x+g+\frac{1}{2}\right)^{x+\frac{1}{2}} e^{-\left(x+g+\frac{1}{2}\right)} P_g(x)$.

3.2 Gamma function with Stirling

Stirling's estimate gives an approximate value for the factorial function n! or the gamma function $\Gamma(n)$ when $n > 1$. For all positive integers, $n! = \Gamma(n+1)$ is applied. The approximation can most simply be derived for n an integer by approximating the sum over the terms of the factorial with an integral, so that $\ln n! = n \ln(n) - n + \Theta \ln(n)$. However the gamma function, unlike the factorial, is more broadly defined for all complex numbers other than non-positive integers; nevertheless, Stirling's formula may still be applied. If $Re(x) > 0$ then, specifying the constant in $\mathscr{O}(\ln(n))$ error term gives $\frac{1}{2}\ln(2\pi n)$, yielding the more precise formula $n! \sim \sqrt{2\pi n}(\frac{n}{e})^n$. Stirling's approximation can be extended to the double inequality as follow $\sqrt{2\pi n}(\frac{n}{e})^n e^{\frac{1}{12n+1}} < n! < \sqrt{2\pi n}(\frac{n}{e})^n e^{\frac{1}{12n}}$

Algorithm 2 Stirling's approximation for Gamma function

```
Require: value: $x > 0$                                      ▷ $x \in Z^+$
Ensure: x is large in absolute value
procedure STIRLINGGAMMA(x)
        $\Gamma(x) = $ SQUAREROOT($n$)              ▷ formula is an asymptotic expansion
    return $\Gamma(x)$
    procedure SQUAREROOT(n)
        error ← 0.00001
        errorPrecision ← 1
        temp ← n
        while $errorPrecision > error$ do
            $n \leftarrow \frac{n+\frac{temp}{n}}{2}$
            $errorPrecision \leftarrow n - \frac{temp}{n}$
        end while
        return $n$
    end procedure
    procedure POWER($base, exponent$)
        Convert the exponent into String
        $exponentArray \leftarrow Split\ the\ exponent\ into\ integer\ and\ fractional\ part$
        if $exponentArray[1] > 0$ then
                return FRACTIONPOWER($base, exponent$)     ▷ it returns fractionPower
        end if
        if $exponent < 0$ then
                $base \leftarrow 1/base$
                $exponent \leftarrow exponent * (-1)$
            return POWER(base, exponent)                   ▷ it returns the power
        end if
        else if $exponent \leftarrow 0$ then
              return 1                                     ▷ it returns 1
        else if $exponent \mod 2 \leftarrow 0$ then
              $base \leftarrow base * base$
              $exponent \leftarrow exponent/2$
        return EXPONENT(base , exponent)                   ▷ it returns power
        else
              $exponent \leftarrow (exponent - 1)/2$
        return  $base$ * POWER(base * base , exponent)     ▷ it returns the power
    end procedure
end procedure
```

```
procedure stirlingGamma(x)
    procedure fractionPower($base, exponent$)
        if $exponent \leftarrow 0$ then
                return 1.0                                        ▷ it returns the value 1
        end if
        if $base < 0$ then
                $ln \leftarrow$ logarithm$(base * (-1))$
        end if
        else
            $ln \leftarrow$ logarithm$(base)$
        if $base \leftarrow 0$ and $exponent > 0$ then
                return answer                                     ▷ it returns the answer
        end if
        for $i \leftarrow 1, 125$ do
                $u \leftarrow$ power$(exponent * ln, i)$
                $l \leftarrow$ factorial$(i)$
                answer = answer + (u/l)
        end for
        if $base < 0$ and $exponent \mod 2 \neq 0$ then
                return $answer * (-1)$                            ▷ it returns the answer * (-1)
        end if
        else
            return answer;                                        ▷ it returns the answer
    end procedure
    procedure logarithm(n)
        $base \leftarrow (n-1)/(n+1)$
        for $i \leftarrow 1, 125$ do
                $e \leftarrow 2 * i - 1$
                $r \leftarrow r + (1/e) *$ power$(base, exponent)$
        end for
        return 2 * $r$                                            ▷ it returns a
    end procedure
    procedure factorial($n$)
            return factorialtailRecursion(n,1)                    ▷ it returns recursivefactorial function
    end procedure
    procedure factorialtailRecusrion($n, a$)
        if $n \leftarrow 0$ then
                return a;                                         ▷ it returns a
        end if
        return factorialTailRecursive(n-1 , n*a)  ▷ it returns factorialTailRecursive
    end procedure
end procedure

$result \leftarrow \Gamma(x)$
```

3.3 Advantages And Disadvantages

3.3.1 The Lanczos approximation method

- Performance : Evidently, the Lanczos approximation method increases efficiency compared to the original gamma function defined simply as $\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, dt$ which need the integral calculating.

- Area of Application: Compared to the regular factorial function, which can only handle positive integers, the Lanczos approximation approach for the gamma function provides a wider range of applications. The method can be extended over the complete complex plane without the negative integer using the reflection formula, but it is only valid for arguments in the right complex half-plane when using the formula deduction.

- Accurateness : The outcome deviates slightly since the Lanczos approach is still only an approximation.

- Disadvantages : It can be challenging to calculate the Lanczos coefficients, and how well the calculations turn out relies on the two arbitrary parameters, g and n. Another disadvantage is that for fixed real parts of the free parameter, utilising complex coefficients increases calculation time while decreasing accuracy of the Lanczos approximation.

3.3.2 The Stirling's approximation method

- Performance : Since it takes time to resolve convergent improper integration, the Stirling's approximation method is more efficient than the original gamma function.

- Area of Application: For any complex numbers other than non-positive integers, the Stirling's approximation for gamma function is more generally defined than the factorial, but Stirling's approximation can still be used.

- Accurateness : The benefit of the Stirling approximation approach is that even when the value of the parameter x is relatively small, the result is still quite accurate and results are almost identical when the value of the parameter x is large.

- Disadvantage : The drawback of this approach cannot comprehend real values that are less than or equal to 0.

3.3.3 Conclusion

The Stirling's approximation method is quite straightforward when compared to other gamma function implementation techniques.

3.4 MindMap

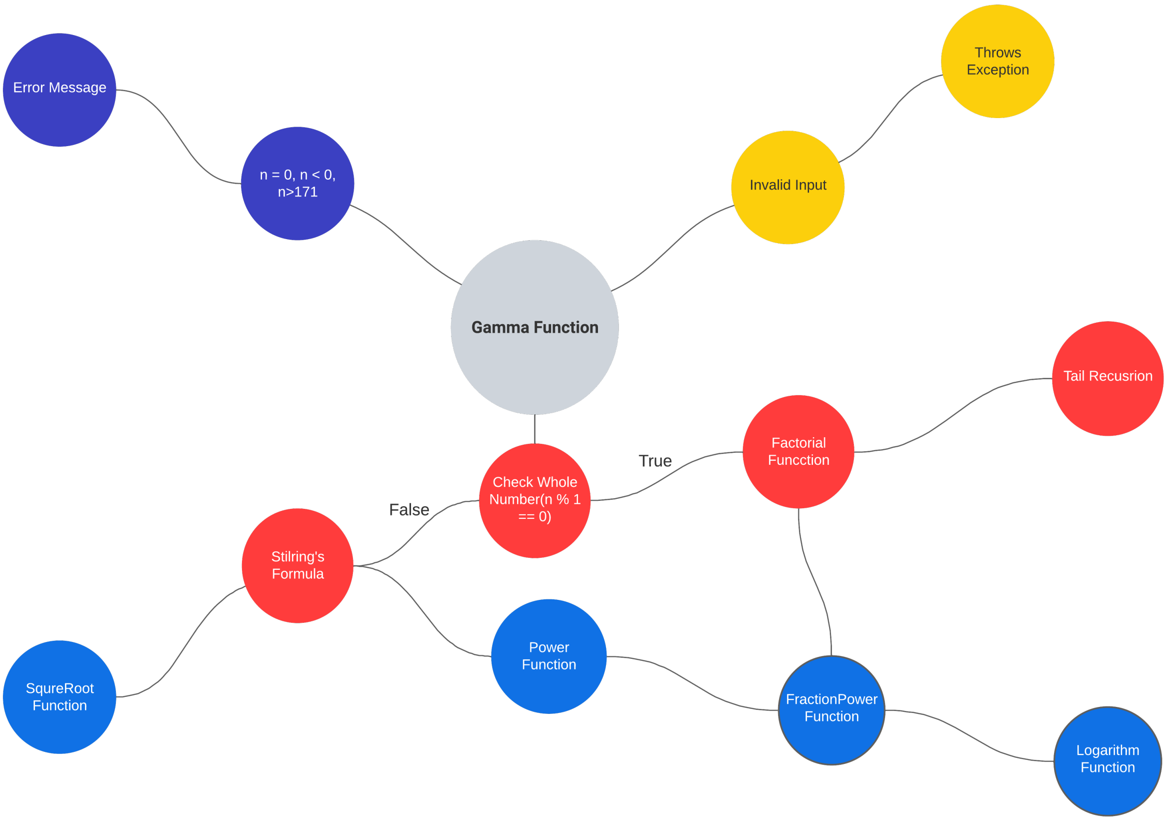


Figure 3.1: MindMap to implement Gamma Function.

4 Problem 4

4.1 Error Handling and Messaging

Maintaining the application's regular flow is the major benefit of exception handling. Exception is an unwanted or unexpected event, which occurs during the execution of a program

In Gamma Function calculator, if user enters any invalid input ($n = 0, n < 0, n > 171$), then the program will notify user with an error message containing ”Infinity” as Gamma value of such numbers are positive infinity

Additionally, when a user accidentally enters a String, then in such case program throws an InputmisMatchException with a message that ”Please enter correct input”. In Gamma Function, exceptions are handled using try..catch method.

4.2 Quality Attributes

- Usability : To calculate the corresponding value of $\Gamma(n)$, the user only needs to enter the value of n, if the input is correct then the system returns the calculated value otherwise for the invalid input, the system also returns the exception that the input is not valid, which can help the user to easily perform the gamma function.
- Robustness : System can take positive real numbers, if the input is not valid then the system simply throws an exception and prints the error handling message containing ”Invalid input”, but the system will not crash.
- Efficient : The system is effective, which is made possible by a straightforward and uncomplicated method; As a result, the user will receive a prompt response within a second of entering correct input, whereas the system prints ”Infinity” promptly when user enters 0, less than 0 and greater than 171 as input.
- Maintainability : The separation of the function modules makes the system maintainable. The project's code follows standard programming practises, and each function and variable has a name that makes it clear what it does. Javadoc is written for each function hence the workload for maintenance is minimal.
- Correctness : The system's main requirement is correctness and precision. By using Stirling's formula for valid input, the application process will provide a reasonably accurate output.

4.3 Debugger

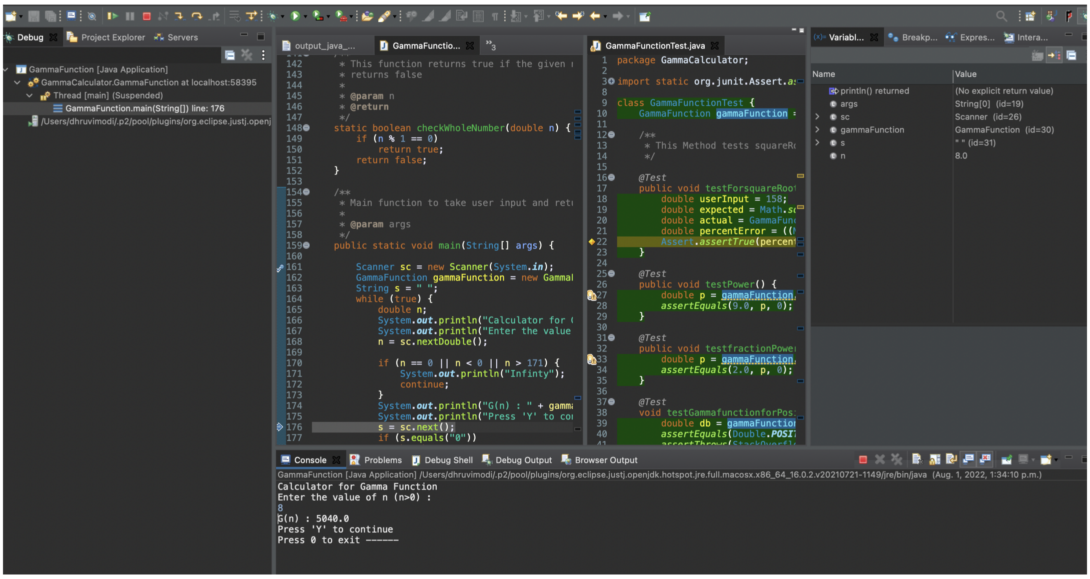
Finding and resolving errors in a program's source code is the process of debugging. Debugging tools are offered by contemporary IDEs like Eclipse, making it simpler for developers to interactively go through their code and analyse it to find and fix any flaws. To assist a developer in debugging successfully and efficiently, the Eclipse Java IDE offers a variety of debugging tools and perspectives grouped in the Debug Perspective.

4.3.1 Advantages

- It is efficient, does not require coding knowledge, and provides the greatest context for identifying issues.
- Error happens in a specific function, but code does not use that function until after the programme has been running for a while. Breakpoints specify where to "pause" a program's execution for the debugger. This makes it possible for you to access the desired area of the software fast.
- One of a debugger's most potent capabilities is single-stepping, which enables a reverse engineer to carry out a single instruction at a time before handing back control to the debugger.

4.3.2 Disadvantages

- It requires human engagement, has a fair learning curve, is not always accessible, and cannot persist output.
- Another drawback of debugger is that the programmer cannot select which method to enter when there are several method calls because it lacks the wise step into function.



```
GammaFunction [Java Application]
  GammaCalculator.GammaFunction at localhost:58395
    Thread [main] (Suspended)
      GammaFunction.main(String[]) line: 176
  /Users/dhruvimodi/.p2/pool/plugins/org.eclipse.justj.openj
```

```
     * This function returns true if the given 
     * returns false
     *
     * @param n
     * @return
     */
    static boolean checkWholeNumber(double n) {
        if (n % 1 == 0)
            return true;
        return false;
    }

    /**
     * Main function to take user input and ret
     *
     * @param args
     */
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        GammaFunction gammaFunction = new Gamma
        String s = " ";
        while (true) {
            double n;
            System.out.println("Calculator for (
            System.out.println("Enter the value
            n = sc.nextDouble();

            if (n == 0 || n < 0 || n > 171) {
                System.out.println("Infinty");
                continue;
            }
            System.out.println("G(n) : " + gamm
            System.out.println("Press 'Y' to co
            s = sc.next();
            if (s.equals("0"))
```

```
package GammaCalculator;

import static org.junit.Assert.a

class GammaFunctionTest {
    GammaFunction gammaFunction =

    /**
     * This Method tests squareR
     */

    @Test
    public void testForsquareRoo
        double userInput = 158;
        double expected = Math.s
        double actual = GammaFun
        double percentError = ((
        Assert.assertTrue(percen
    }

    @Test
    public void testPower() {
        double p = gammaFunction.
        assertEquals(9.0, p, 0);
    }

    @Test
    public void testfractionPowe
        double p = gammaFunction.
        assertEquals(2.0, p, 0);
    }

    @Test
    void testGammafunctionforPos
        double db = gammaFunctio
        assertEquals(Double.POSI
```

Name	Value
println() returned	(No explicit return value)
args	String[0] (id=19)
sc	Scanner (id=26)
gammaFunction	GammaFunction (id=30)
s	" " (id=31)
n	8.0

```
GammaFunction [Java Application] /Users/dhruvimodi/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64_16.0.2.v20210721-1149/jre/bin/java  (Aug. 1, 2022, 1:34:10 p.m.)
Calculator for Gamma Function
Enter the value of n (n>0) :
8
G(n) : 5040.0
Press 'Y' to continue
Press 0 to exit ------
```

Figure 4.1: Screenshot of debugging of Gamma Function.

4.4 Tool-Checkstyle

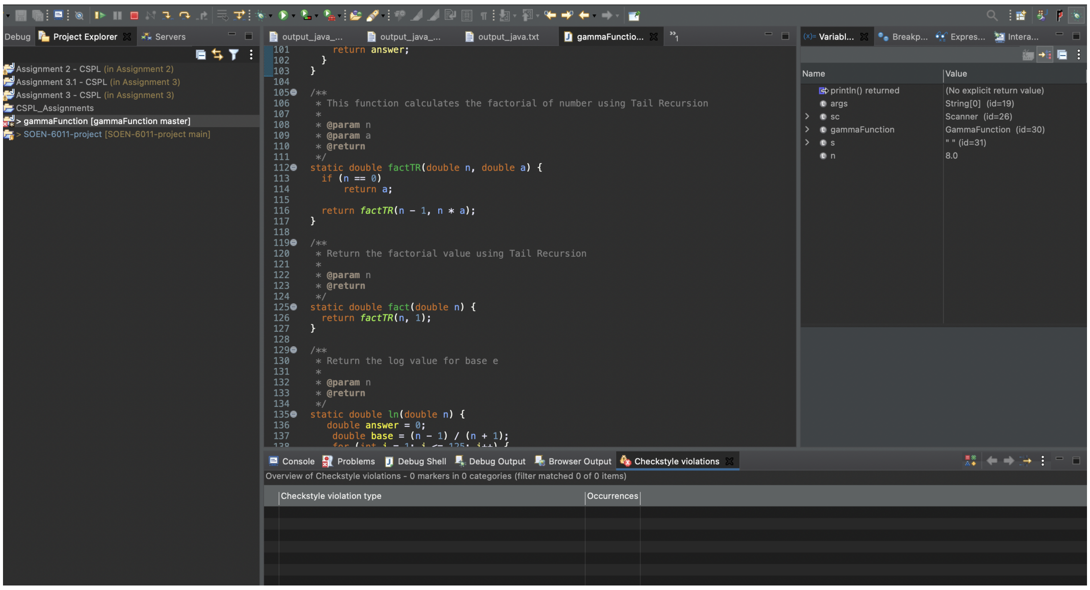
Java developers can use the Open Source development tool Checkstyle to make sure their code complies with a set of coding standards. The static source code analyzer Checkstyle is integrated into the Eclipse IDE by the Eclipse Checkstyle Plugin (eclipse-cs).

4.4.1 Advantages

- Due to the fact that checkstyle was initially intended to be an independent framework, it is much simpler to combine it with other technologies.
- Code specification verification can be automated with CheckStyle, the key components of the checking include Javadoc comments, name conventions, titles, line lengths, the use of imports and scope modifiers, intervals between characters, etc.
- Team projects benefit from improved code readability, higher project quality, and simpler project maintenance.
- It has the expertise its own rules for developers. Although checkstyle includes more styles than Eclipse does and allows programmers to create their own unique rules,

4.4.2 Disadvantages

- Javac must be able to compile the code in order to find legitimate violations. If not, parsing errors are challenging to comprehend.
- It is difficult to ascertain the type of an expression and its whole inheritance structure.
- The binary Byte Order Mark (BOM) of a file cannot be handled effectively by Checkstyle. In addition, each file is processed separately, making it impossible to view the contents of other files while doing checks.



```
      return answer;
    }
  }

  /**
   * This function calculates the factorial of number using Tail Recursion
   *
   * @param n
   * @param a
   * @return
   */
  static double factTR(double n, double a) {
    if (n == 0)
        return a;

    return factTR(n - 1, n * a);
  }

  /**
   * Return the factorial value using Tail Recursion
   *
   * @param n
   * @return
   */
  static double fact(double n) {
    return factTR(n, 1);
  }

  /**
   * Return the log value for base e
   *
   * @param n
   * @return
   */
  static double ln(double n) {
     double answer = 0;
      double base = (n - 1) / (n + 1);
      for (int i = 1; i <= 125; i++) {
```

Name	Value
println() returned	(No explicit return value)
args	String[0] (id=19)
> sc	Scanner (id=26)
> gammaFunction	GammaFunction (id=30)
> s	" " (id=31)
n	8.0

Overview of Checkstyle violations - 0 markers in 0 categories (filter matched 0 of 0 items)

Checkstyle violation type	Occurrences

Figure 4.2: Screenshot of Checkstyle of Gamma Function.

5 Problem 5

5.1 Unit test cases

5.1.1 Test Case 1

- **ID** : TC1
- **Requirement ID** : FR1, FR2, FR4
- **Name of Test Case** : testGammafunctionforPositiveInteger()
- **Test Priority** : High
- **Pre-condition** : Start the programme, then type a value into the console.
- **Test Data** : 142, 234
- **Expected Outcome** : 1.8981437590761713E243, ∞
- **Actual Outcome** : 1.8981437590761713E243, ∞
- **Test Result** : Success

5.1.2 Test Case 2

- **ID** : TC2
- **Requirement ID** : FR2, FR5
- **Name of Test Case** : testGammafunctionforDecimalNumber()
- **Test Priority** : High
- **Pre-condition** : Start the programme, then type a value into the console.
- **Test Data** : 21.14
- **Expected Outcome** : 3.699869886541974E18
- **Actual Outcome** : 3.699869886541974E18
- **Test Result** : Success

5.1.3 Test Case 3

- **ID** : TC3
- **Requirement ID** : FR3
- **Name of Test Case** : testGammafunctionforPositiveInteger()
- **Test Priority** : High
- **Pre-condition** : Start the programme, then type a value into the console.
- **Test Data** : 0, -12
- **Expected Outcome** : Exception, Exception
- **Actual Outcome** : Exception, Exception
- **Test Result** : Success

5.1.4 Test Case 4

- **ID** : TC4
- **Requirement ID** : FR3
- **Name of Test Case** : testPositiveLimitForGammaFunction()
- **Test Priority** : Medium
- **Pre-condition** : Start the programme, then type a value into the console.
- **Test Data** : 175
- **Expected Outcome** : $+\infty$
- **Actual Outcome** : Exception, Exception
- **Test Result** : Success

5.1.5 Test Case 5

- **ID** : TC5
- **Requirement ID** : FR1
- **Name of Test Case** : testMainmethod()
- **Test Priority** : High
- **Pre-condition** : Start the programme, then type a value into the console.
- **Test Data** : 9 y ghjk 0 -4 200 67 y 150 0
- **Expected Outcome** : 40320.0, Exception, Exception, Error message, Error message, ∞, 5.443449390774432E92, Exception, 3.8089226376305687E260, Error message

- **Actual Outcome** : 40320.0, Exception, Exception, Error message, Error message, ∞, 5.443449390774432E92, Exception, 3.8089226376305687E260, Error message
- **Test Result** : Success

5.1.6 Conclusion

The Junit testing with the Junit 5 standard is used to test the Gamma Function programme. The running result and the expected outcome are compared using a series of ”assertion” methods to determine whether all test cases either succeed or fail.

Bibliography

Gamma function
https://en.wikipedia.org/wiki/Gamma function

[Reference Resource] Function
https://www.physics.uoguelph.ca/chapter-2-gamma-function

29148-2018 — ISO/IEC/IEEE International Standard – Systems and software engineering – Life cycle processes – Requirements engineering. (2018, November 30). Retrieved from https://standards.ieee.org/standard/29148-2018.html

Stirling's Formula.
https://mathworld.wolfram.com/StirlingsApproximation.html

Stirling's approximation.
https://en.wikipedia.org/wiki/Stirling's-approximation-An-alternative-derivation

Lanczos approximation.
https://en.wikipedia.org/wiki/Lanczos-approximation

Exception Handling in JAVA
https://www.geeksforgeeks.org/exceptions-in-java/

Debugging
https://www.techtarget.com/searchsoftwarequality/definition/debugging

Google CheckStyle for JAVA Programming
https://checkstyle.sourceforge.io/google_style.html