UNIVERSITÉ Concordia UNIVERSITY

SOEN 6011 - Software Engineering Processes

Summer 2019

Scientific Calculator- ETERNITY: FUNCTIONS

Project Report

Deliverable 1

Presented to

Instructor: PANKAJ KAMTHAN

By
Prashanthi Ramesh

July 20, 2019

Contents

Chapter 1

Problem 1

1.1 Function Definition

Exponentiation[4] is a mathematical operation, written as x^y, involving two numbers, the base x and the exponent or power y. When y is a positive integer, exponentiation corresponds to repeated multiplication of the base: that is, x^y is the product of multiplying y bases:

$$f(x, y) = x^y \tag{1.1}$$

1.1 is known as the Exponentiation function

$$x^y = x * ... * x \text{ (y times)} \tag{1.2}$$

1.2 is depicts the evaluation of exponentiation function

1.2 Domain

A set of real numbers[5] (-∞ to ∞)

$$\{(x, y) \in \mathbb{R}^2 : (x \geq 0 \wedge y \neq 0) \vee x > 0\} \tag{1.3}$$

1.3 presents the domain of exponentiation function

1.3 Co-domain

A set of all positive real numbers[3] (0 to ∞)

1.4 Characteristics

- **Graph:**
  - In an exponential graph, the "rate of change" increases (or decreases) across the 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.

- **Injectivity:** x^y function is injective which means it is one-to-one function
- **Surjective:** x^y function is not surjective which means it is not onto function
- $x^y = 1$ when y = 0 and x $\neq$ 0
- $x^y = Undefined$ when y = 0 and x = 0
- $x^y = Undefined$ when x = 0 and y $<$ 0

Chapter 2

Problem 2

This section presents the assumptions, requirements for implementing the function f(x,y)=x^y.

2.1 Assumptions

1. The Eternity Function will accept only real numbers as input
2. The Eternity Function will be able to handle entry value of double-precision floating-point
3. The Eternity Function result value will be accurate to fifteen decimal places

2.2 Requirements

Identification	EF_REQ_1
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x as any real number except zero and for y as zero (0), the Eternity Function shall display the result as 1
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_2
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x as any real number except zero and for y as 1, the Eternity Function shall display the result as the same value as x
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_3
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x as any real number and for y as a unreal, non-numeric value, the Eternity Function shall display an error message
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_4
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x a unreal, non-numeric value and for y as a real number, the Eternity Function shall display an error message
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_5
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x a unreal non-numeric value and for y as a unreal non-numeric value, the Eternity Function shall display an error message
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_6
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x as zero (0) and for y as zero (0), the Eternity Function shall display the result an error message
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_7
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x as a real number and for y as real number less than zero (negative value), the Eternity Function shall display the result an error message
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_8
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x as zero (0) and for y as real number, the Eternity Function shall display the result as 1
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_9
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x as a real number less than zero (negative real number) and for y as an odd real number except zero (0), the Eternity Function shall display the result as negative value
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_10
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When the user inputs a value for x as a real number less than zero (negative real number) and for y as an even real number except zero (0), the Eternity Function shall display the result as positive value
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_11
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	Medium
Risk	Low
Description	The Eternity Function shall be available to users for use all the time
Difficulty	Nominal
Type	Non-Functional

Identification	EF_REQ_12
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	Medium
Risk	Low
Description	When the Eternity Function is non-operational, the system shall present the user with notification informing them that the system is unavailable
Raresent the use-tionale	Why
Difficulty	Nominal
Type	Non-Functional

Identification	EF_REQ_13
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	The Eternity Function shall be able to handle entry value of double-precision floating-point.
Difficulty	Nominal
Type	Non-Functional

Identification	EF_REQ_14
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	The Eternity Function shall have a maximum response time of three seconds
Difficulty	Nominal
Type	Non-Functional

Identification	EF_REQ_15
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	The Eternity Function result value shall be accurate to fifteen decimal places
Difficulty	Nominal
Type	Non-Functional

Identification	EF_REQ_15
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	Medium
Risk	Low
Description	The Eternity Function shall save the history of operations
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_16
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	Medium
Risk	Low
Description	The user shall view the history of operations
Difficulty	Nominal
Type	Functional

Identification	EF_REQ_17
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	Low
Risk	Low
Description	The Eternity Function shall be easy to use by members of the public who may have at-least one hand free
Difficulty	Nominal
Type	Usability

Identification	EF_REQ_18
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	Medium
Risk	Medium
Description	The Eternity Function shall be used by members of public without training
Difficulty	Nominal
Type	Usability

Identification	EF_REQ_19
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	A development programmer who has at least one year of experience supporting this software application shall be able to add a new product feature, including source code modifications and testing
Difficulty	Nominal
Type	Non-functional

Identification	EF_REQ_20
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	Medium
Risk	Medium
Description	It shall be possible for the Eternity Function to be installed by a user who has no special expertise
Difficulty	Nominal
Type	Non-functional

Identification	EF_REQ_21
Version Number	1.0
Owner	Prashanthi Ramesh
Priority	High
Risk	Medium
Description	When a new version of the Eternity Function is released, it shall be possible to upgrade to it from any previous version
Difficulty	Nominal
Type	Non-functional

Chapter 3

Problem 3

3.1 Pseudocode

Algorithm 1: 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.

$$x^y = e^{y \ln x} \tag{3.1}$$

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

$$e^x = 1 + x/1! + x^2/2! + x^3/3! + ... \tag{3.2}$$

3.2 express e^x using Taylor Series

$$e^x = 1 + (x/1)(1 + (x/2)(1 + (x/3)(........))) \tag{3.3}$$

3.3 The series 3.2 can be re-written as above

$$log(1 + x) = x - x^2/2 + x^3/3 - ... \tag{3.4}$$

3.4 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
Object Reference Model parameters embedded as variables	fsdfds
Power series can be inverted to yield the inverse function	Almost always not as efficient as curve fitting or direct approximation

Algorithm 1 Exponentiation[2] by Taylor Series

```
Require: x ≠ 0 AND y > 0
function LOGARITHM(n)                          ▷ algorithmforlog(n)
    sum ← 0
    while n > 1 do
        n ← n/e                                ▷ e is a constant approximately equal to 2.71828
        y ← y + 1
    end while
return y
end function
function EXPONENTIAL(x)                        ▷ algorithmfore^x
    sum ← 1
    n ← 10
    for i ← n − 1, 1 do
        sum ← 1 + x ∗ sum/i
    end for
return sum
end function
logx ←LOGARITHM(x)
result ←EXPONENTIAL(y*logx)
```

Algorithm 2:

- Approximation algorithms[1] are efficient algorithms that find approximate solutions to NP-hard optimization problems with provable guarantees on the distance of the returned solution to the optimal one
- Neural network simulations often spend a large proportion of their time computing exponential functions.

- Since the exponentiation routines of typical math libraries are rather slow, their replacement with a fast approximation can greatly reduce the overall computation time.
- An approximation is perfectly adequate for most neural computation purposes and can save much time.
- Exploiting the IEEE 754 floating-point representation to calculate e^x

Advantages	**Disadvantages**
The calculation only requires 2 shifts, 1 multiplication, 2 addition, and 2 register operations.	The algorithm efficiency gets worse as the values get larger and are not very accurate
The approximation function results are very fast.	Depends on IEEE 754 standard and can have compatibility issues

Algorithm 2 Approximation of the Exponential Function

Require: $x \neq 0$ AND $y > 0$

1: $x \leftarrow >> 32$ ▷ Calculate ln(x)
2: $ln_x \leftarrow x - 1072632447/1512775$
3: $exponent \leftarrow ln_x * y$ ▷ ln(x)*y
4: $result \leftarrow 1512775 * exponent + (1072693248 - 60801)(1072707600)$ ▷ $e^l n(x) * y$
5: $result = result << 32$

Appendix A

GitHub

A.1 Individual GitHub Link

https://github.com/PrashanthiRamesh/SOEN-6011-Project-Calculator/

A.2 Team GitHub Link

https://github.com/niravjdn/SOEN-6011-Project/

Bibliography

[1] Martin Ankerl. *Optimized pow() approximation for Java*. 2017. URL: https://martin.ankerl.com/2007/10/04/optimized-pow-approximation-for-java-and-c-c/ (visited on 10/01/2017).

[2] Szasz Janos. *The algorithmicx package*. 2005. URL: http://tug.ctan.org/macros/latex/contrib/algorithmicx/algorithmicx.pdf (visited on 04/27/2005).

[3] MathBits Teacher. *Exponential Functions*. 2019. URL: https://mathbitsnotebook.com/Algebra2/Exponential/EXExpFunctions.html (visited on 07/06/2019).

[4] Wikipedia. *Exponentiation*. 2019. URL: https://en.wikipedia.org/wiki/Exponentiation (visited on 07/06/2019).

[5] WolframAlpha. *Domain of exponentiation function*. 2019. URL: https://www.wolframalpha.com/input/?i=x%5C%5Ey (visited on 07/06/2019).