# Islamic University of Technology

**Board Bazar, Gazipur, Dhaka, 1704, Bangladesh**

---

**SWE 4304 - Software Project Lab**

Final Project Report

---

# MathVoyage JAVA library

Team Voyager

January 19, 2024

# Team Voyager

Adib Sakhawat
ID : 210042106
BSc. in Software Engineering
Department of Computer Science and Engineering

Takia Farhin
ID : 210042117
BSc. in Software Engineering
Department of Computer Science and Engineering

Tahsin Islam
ID : 210042137
BSc. in Software Engineering
Department of Computer Science and Engineering

**Supervised by:**

Syed Rifat Raiyan
Lecturer
Department of Computer Science and Engineering
Islamic University of Technology

**Abstract**

Math libraries are essential tools for developers, simplifying complex calculations. Introducing ‘mathVoyage,’ a Java toolkit offering versatile math functions. In this paper, we are describing the implementation of coordinate geometry, trigonometry, matrix operations, combinatorics, vector, number system modules, and more. It streamlines mathematical tasks using several well-known algorithms, saving developers time and ensuring precision in Java applications.

# Contents

**1 Project Introduction and Motivation** **3**
- 1.1 Introduction . . . . . 3
- 1.2 Motivation . . . . . 5

**2 Implemented Features** **7**
- 2.1 Coordinate Geometry Module . . . . . 7
- 2.2 Trigonometry Module . . . . . 9
- 2.3 Matrix Module . . . . . 9
- 2.4 Vector Module . . . . . 10
- 2.5 Number Theory Module . . . . . 11
- 2.6 Basic Numerical Module . . . . . 12
- 2.7 Combinatorics . . . . . 12
- 2.8 Bit-wise Operation . . . . . 12

**3 Tools and Technologies** **14**
- 3.1 Programming Language . . . . . 14
- 3.2 Development Environment . . . . . 14
- 3.3 Version Control . . . . . 14
- 3.4 Documentation Tools . . . . . 15
- 3.5 Collaboration Tools . . . . . 15
- 3.6 Progress Tracking Tools . . . . . 15
- 3.7 Performance Optimization . . . . . 15

**4 Application Domain** **21**
- 4.1 Purpose of VMath . . . . . 21
- 4.2 Target Audience . . . . . 21
- 4.3 Key Features . . . . . 21
  - 4.3.1 Modularity and Extensibility . . . . . 21
  - 4.3.2 Ease of Use . . . . . 22
- 4.4 Real-World Applications . . . . . 22

**5 Contributions** **23**
5.1 Numeric Discussion . . . . . . . . 23
5.2 Individual Contribution . . . . . . . . 24
5.2.1 Adib Sakhawat - 210042106 . . . . . . . . 24
5.2.2 Tahsin Islam - 210042137 . . . . . . . . 24
5.2.3 Takia Farhin - 210042117 . . . . . . . . 25

**6 Useful Links** **26**

# Chapter 1

# Project Introduction and Motivation

## 1.1 Introduction

In software development, having strong math tools is crucial as they serve as the foundation for many applications and solutions. That's why we introduced mathVoyage, a Java library encompassing a wide range of math tools for developers.

**Project Overview**

In today's software ecosystem, precision and efficiency are paramount. Whether crafting scientific simulations, engineering solutions, or applications requiring mathematical rigor, mathVoyage aims to be the trusted companion. This project addresses intricate demands of mathematics and geometry, providing a versatile collection of functions to simplify complex calculations and enhance the precision and versatility of Java applications.

**Module Organization**

The mathVoyage project is organized into modules, each catering to specific mathematical domains:

- Coordinate Geometry
- Trigonometry
- Matrix Algebra
- Vector Operations
- Bit-wise Operations

- Combinatorics
- Number Conversions

**Coordinate Geometry Module**

This module equips developers with the ability to effortlessly handle points, lines, and shapes in 2D space. It includes functions for calculating distances, classifying triangles, and identifying various quadrilaterals.

**Trigonometry Module**

The library provides a rich set of functions for converting between degrees and radians, computing sine, cosine, tangent, cotangent, secant, cosecant, and their inverse counterparts. These functions are crucial for precise mathematical calculations involving angles.

**Matrix Algebra Module**

Facilitating matrix creation, manipulation, and operations, including determinant calculations, addition and subtraction, multiplication (with other matrices and scalars), transposition, inversion, and powering matrices to integer exponents.

**Vector Operations Module**

Essential vector arithmetic functions include addition, subtraction, scalar multiplication, and cross-product calculations.

**Bit-wise Operations Module**

Functions like AND, OR, NOT, XOR, Left Shift, and Right Shift are provided for powerful binary data manipulation.

**Combinatorics Module**

Useful for permutation and combination calculations, determining the number of subsets, and returning subsets.

**Number Conversions Module**

Practical number conversion functions, including decimal to binary, octal, hexadecimal, and n-base conversions, as well as reverse conversions.

**Conclusion**

In summary, the mathVoyage project stands as a testament to our commitment to providing developers with a powerful toolbox of mathematical functionalities. It simplifies work, enhances precision, and boosts the efficiency of Java applications. We invite you to explore this project and embark on your mathematical voyage with confidence.

## 1.2 Motivation

In the world of software development, mathematics served as the universal language that underpinned the creation of applications, simulations, and solutions across diverse domains. From engineering and physics simulations to financial modeling, data analysis, and game development, mathematical operations were an integral part of the software development process.

However, despite the commonality of mathematics in software, developers often faced significant challenges when it came to implementing and managing complex mathematical computations. These challenges included the need for precision, the efficient handling of mathematical entities, and the availability of versatile tools that catered to a wide range of mathematical domains.

The mathVoyage library was born out of the recognition of these challenges and the conviction that every developer, regardless of their mathematical background, should have access to a comprehensive and user-friendly mathematical library that simplified intricate calculations and enhanced the precision of their applications.

Here were some key motivating factors behind the project of making the mathVoyage library:

#### Empowering Developers

The aim was to empower developers, from students to professionals, with a powerful set of mathematical tools. By providing an extensive library that covered various mathematical and geometric domains, the goal was to democratize access to advanced mathematical functionalities.

#### Precision Matters

In fields such as scientific research, engineering, and finance, even the smallest margin of error could have profound consequences. mathVoyage was designed to assist developers in achieving the highest levels of accuracy in their mathematical computations.

#### Time and Resource Efficiency

By offering efficient algorithms and data structures, mathVoyage aimed to help developers save valuable development time and computational resources. This efficiency was particularly important in real-time systems and applications that required rapid mathematical computations.

#### Cross-Disciplinary Applications

The mathVoyage library was versatile, making it suitable for a wide range of applications across different domains. Whether developers were working on physics simulations, game development, data analysis, or engineering projects, mathVoyage served as a valuable asset.

#### Education and Learning

mathVoyage also had educational value. It aided students and educators in understanding and visualizing mathematical concepts through practical implementation, making learning more engaging and tangible.

#### Community and Collaboration

mathVoyage was envisioned as a community-driven project where developers contributed to the growth and enhancement of the library. This collaborative spirit fostered innovation and ensured that mathVoyage remained a dynamic and evolving resource.

# Chapter 2

# Implemented Features

The mathVoyage library has been organized into distinct modules, each meticulously designed to address specific mathematical and geometric domains. These modules have been tailored to provide a comprehensive suite of mathematical functionalities, making mathVoyage a versatile and valuable resource for developers across various disciplines. Below, we will provide an overview of each module, highlighting its key features and objectives.

## 2.1 Coordinate Geometry Module

This module is dedicated to handle geometric entities in 2D space. It offers the following functionalities:

- Creation of Points
- Euclidean distance between 2 points
- Manhattan distance between 2 points
- Chebyshev distance between 2 points
- Area of a triangle given 3 points
- Area of a quadrilateral given 4 points
- Area of a circle given 2 endpoints of radius
- Triangle: is equilateral?
- Triangle: is isosceles?
- Triangle: is Scalane?

- Triangle: is Acute?
- Triangle: is Obtuse?
- Triangle: is Right?
- Quadrilateral: is rectangle?
- Quadrilateral: is square?
- Quadrilateral : is rhombus ?
- Quadrilateral : is Parallelogram ?
- Calculate Slope given point
- Creation of Line
- Check if same line?
- Check if parallel line?
- if perpendicular
- Intersection Point
- Calculate the Slope given line
- Distance of two parallel line
- Area of a triangle given 3 line
- point in polygon
- Area for convex hull polygon
- Area of a quadrilateral given 4 line
- Calculate the Midpoint between two given points
- Interleaver point
- Externalizer point
- Centroid of a Triangle
- Circumference of a circle
- Angle between two slopes

- Perpendicular distance of a line from a point
- The perpendicular line of the line passing through a point
- The parallel line of the line passing through a point

## 2.2 Trigonometry Module

Precise trigonometric calculations is essential in numerous scientific and engineering applications. The Trigonometry module provides functions for:

- $sin$ Calculation
- $cos$ Calculation
- $tan$ Calculation
- $cot$ Calculation
- $sec$ Calculation
- $cosec$ Calculation
- $sin^{-1}$ Calculation
- $cos^{-1}$ Calculation
- $tan^{-1}$ Calculation
- $sec^{-1}$ Calculation
- $cosec^{-1}$ Calculation
- $cot^{-1}$ Calculation
- Conversion from Sexagesimal to Circular and vice-versa.

These functions empower users with accurate trigonometric operations for various tasks.

## 2.3 Matrix Module

Matrices will be fundamental in mathematical modeling and data manipulation. The Matrix module will facilitate:

- Creation of Matrix

- Identity Matrix Creation
- Zero matrix creation
- One matrix creation
- Determinant
- Matrix Addition
- Matrix Subtraction
- Matrix Multiply
- Matrix Multiply with Number
- Matrix Transpose
- Inverse Matrix
- Power Matrix

This module is indispensable for applications involving linear algebra and matrix transformations.

## 2.4 Vector Module

Vectors plays a crucial role in physics, engineering, and computer graphics. The Vector module allows users to work with vectors by providing operations for:

- Creation of Vector
- Vector Addition
- Vector Subtraction
- Vector Scalar Multiplication
- Vector Cross Multiplication

These functions simplifies vector calculations, making them accessible to developers.

## 2.5 Number Theory Module

Number conversions will often be required when working with different numeral systems. The Number Theory module will offer a range of functions for:

- Decimal to Binary
- Decimal to Octal
- Decimal to Hexadecimal
- Decimal to n-Base
- Binary to Decimal
- Binary to Octal
- Binary to Hexadecimal
- Binary to n-Base
- Octal to Binary
- Octal to Decimal
- Octal to Hexadecimal
- Octal to n-Base
- Hexadecimal to Decimal
- Hexadecimal to Binary
- Hexadecimal to Octal
- Hexadecimal to n-Base
- $n$-Base to $k$-Base
- LCM Calculation
- GCD Calculation

These functions will be vital for various applications, including digital systems and data encoding.

## 2.6 Basic Numerical Module

The Basic numerical module will include fundamental mathematical operations like:

- $n$-th root calculation
- Maximum of two numbers
- Maximum value of an array
- Index of the maximum value of an array
- Minimum of two numbers
- Minimum value of an array
- Index of minimum value of an array
- Factorial

These functions will cater to basic mathematical needs and will serve as building blocks for more complex computations in other modules.

## 2.7 Combinatorics

The Combinatorics module in mathVoyage will provide functions for:

- Combination (${}^{n}C_{r}$)
- Permutation (${}^{n}P_{r}$)
- number of subset of a set and return the subset

These tools are essential for solving a variety of mathematical problems, particularly in areas like probability and discrete mathematics. With this module, developers can expect accurate and efficient solutions to combinatorial challenges, enhancing the versatility of the mathVoyage library.

## 2.8 Bit-wise Operation

The Bit-wise Operation module in mathVoyage will offer key functions like:

- Bit-wise And
- Bit-wise OR

- Bit-Wise XOR
- Bit-Wise NOT
- Left Shift
- Right Shift
- Zero Fill Right Shift
- Get Byte
- Set bit 0
- Set bit 1
- Toggle bit
- Logical shift
- Right rotate
- Left rotate
- Add Overflowed

These operations are fundamental for low-level programming tasks, providing precise control over individual bits in binary data. This module will be a valuable resource for developers working on tasks that demand detailed bit-level computations and optimizations.

# Chapter 3

# Tools and Technologies

The development of the mathVoyage software project has relied on a carefully chosen set of tools and technologies to ensure its robustness, efficiency, and versatility. Leveraging industry-standard and well-established resources, we have been committed to creating a reliable mathematical library that is accessible to developers across diverse domains. In this section, we will provide a comprehensive overview of the software development tools, programming languages, libraries, and technologies that have underpinned the mathVoyage project. These tools have been thoughtfully selected to facilitate efficient development and seamless integration of mathematical functionalities, empowering users to harness the power of mathematics in their applications with ease and precision.

## 3.1 Programming Language

As we have built this library for **JAVA** based applications we chose **JAVA** for this library.

## 3.2 Development Environment

The mathVoyage software project was developed using Java within the IntelliJ IDEA integrated development environment for coding, debugging, and documenting.

## 3.3 Version Control

The project's source code was managed using a Git repository hosted on GitHub, with individual branches dedicated to each contributor to maintain the chronological order of contributions.

## 3.4 Documentation Tools

Project documentation, including code comments, user manuals, has been generated and maintained using Javadoc within the IntelliJ IDEA integrated development environment.

## 3.5 Collaboration Tools

Communication and collaboration among team members have been facilitated through a Discord server, while online meetings and discussions was conducted using Google Meet.

## 3.6 Progress Tracking Tools

In our project, Google Sheets served as a dynamic platform for seamlessly tracking and managing our progress.

## 3.7 Performance Optimization

### Algebraic Functions

Most of the algebraic functions are running in constant time. Finding the *Global minimum* or *Global maximum* in an array takes linear time.

One of the most challenging function in Algebraic module is the **Root Calculation** function. To efficiently and correctly solve the problem we have used **Newton-Raphson** method.
**Newton-Raphson Method Formula for Square Root:**

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{A}{x_n}\right)$$

In the Newton-Raphson method:

- $x_{n+1}$: The next approximation.
- $x_n$: The current approximation.
- $A$: The number for which the square root is approximated.

Here, $x_0$ the initial approximation, and the iterative formula for improving the approximation

Another challenge was to provide the precise factorial calculation of any number efficiently. By using memoization technique we are calculating factorials whenever any one factorial call is happening. Then we are storing all the values in an array with a linear time. After the first array instantiating, by the singleton principle we always get the same array so there will be no time returning the factorial of a number.

To calculate $GCD$ and $LCM$ we have used euclidean algorithm.

$$\gcd(a,b) = \begin{cases} a & \text{if } b = 0 \\ \gcd(b, a \mod b) & \text{otherwise} \end{cases}$$

This is the euclidean recurrence algorithm we have used in vmath.

**Trigonometric Functions**

In trigonometric functions we are using 'Expansion of Taylor Series' for each of the functions.

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!}$$

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!}$$

$$\sin^{-1}(x) = x + \frac{x^3}{6} + \frac{3x^5}{40} + \frac{5x^7}{112} + \ldots = \sum_{n=0}^{\infty} \frac{(2n)!}{4^n (n!)^2 (2n+1)} x^{2n+1}$$

$$\tan^{-1}(x) = \begin{cases} x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \ldots = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{2n+1} & \text{for } |x| < 1 \\ \frac{\pi}{2} - \left(\frac{1}{x} - \frac{1}{3x^3} + \frac{1}{5x^5} - \frac{1}{7x^7} + \ldots\right) = \frac{\pi}{2} - \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} \frac{1}{x^{2n+1}} & \text{for } x \geq 1 \\ -\frac{\pi}{2} - \left(\frac{1}{x} - \frac{1}{3x^3} + \frac{1}{5x^5} - \frac{1}{7x^7} + \ldots\right) = -\frac{\pi}{2} - \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} \frac{1}{x^{2n+1}} & \text{for } x \leq -1 \end{cases}$$

Here we are calculating $80$ terms for each expansion. That is giving us the more precious value of the function in constant time as we know $80$ is a very small number compared to the processor power and,

$$O(80) = O(1)$$

That's why each and every function of trigonometric module produces precise result in **constant time**.

### Geometry Functions

In this module, we have used different types of equations to optimize geometry module.

For determining Euclidean distance we used,

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Here,

- $x_1$ = x coordinate of the first point.
- $y_1$ = y coordinate of the first point.
- $x_2$ = x coordinate of the second point.
- $y_2$ = y coordinate of the second point.
- $d$ = distance

For determining the Manhattan distance we used,

$$d = |x_2 - x_1| + |y_2 - y_1|$$

For the Chebyshev distance, we used,

$$d = max(|x_2 - x_1|, |y_2 - y_1|)$$

The slope of a line is a measure of how steep the line is. We have used two equations to determine the slope. For two points, slope is determined by this equation:

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

$$m = -\frac{a}{b}$$

Where,

- $a$ = coefficient of x of the line
- $b$ = coefficient of y of the line

To determine the interleaved point of two points that divide the connecting line in a ratio of m:n, we used,

$$x = \frac{m * x_2 + n * x_1}{m + n}, y = \frac{m * y_2 + n * y_1}{m + n}$$

And for the externalizer point of two points that divide the connecting line in a ratio of m:n, we used

$$x = \frac{m * x_2 - n * x_1}{m - n}, y = \frac{m * y_2 - n * y_1}{m - n}$$

To determine the angle between two slopes, we have used,

$$\theta = tan^{-1}\frac{m_1 - m_2}{1 + m_1 m_2}$$

To determine the intersection point of two lines we have used,

$$x = \frac{b_1 c_2 - b_2 c_1}{a_1 b_2 - a_2 b_1}, y = \frac{c_1 a_2 - c_2 a_1}{a_1 b_2 - a_2 b_1}$$

Here,

- $a_1$ = coefficient of x of the first line
- $b_1$ = coefficient of y of the first line
- $c_1$ = constant value of the first line
- $a_2$ = coefficient of x of the second line
- $b_2$ = coefficient of y of the second line

- $y_2$ = constant of the second line
- $x$ = x coordinate of intersection point
- $y$ = y coordinate of intersection point

We have determined the distance between two parallel lines by using,

$$d = \frac{|c_1 - c_2|}{\sqrt{a^2 + b^2}}$$

We also determined the perpendicular distance of a line from a point. We used,

$$d = \frac{|ax_1 + by_1 + c|}{\sqrt{a^2 + b^2}}$$

We also have methods that can determine a line that goes through a given point and which also is parallel or perpendicular to a given line.
To get a line that goes through a point $P(x_1, y_1)$ and is perpendicular to a line $ax+by+c = 0$, we used

$$bx - ay + k = 0$$

$$k = ay_1 - bx_1$$

To get the line that goes through a point $P(x_1, y_1)$ and is parallel to a line $ax + by + c = 0$, we used

$$ax + by + k = 0$$

$$k = -1(ax_1 - by_1)$$

We have also determined area of convex hull polygon by using this formula,

$$A = \frac{1}{2}\left|\sum_{i=1}^{n-1}(x_i y_{i+1} - x_{i+1} y_i) + (x_n y_1 - x_1 y_n)\right|$$

By using all these equations, we were able to reduce the time complexity to $O(1)$.

**Others**

In other modules we have implemented all the functions in pretty straightforward way. All the functions maintain well known algorithms related to them.

These performance optimization strategies reflects our unwavering commitment to providing users with a robust mathematical toolkit that excels in both speed and precision, catering to a wide range of mathematical and scientific applications.

In this chapter, we've outlined the foundational tools and technologies driving the development of the mathVoyage software project. With Java as our core programming language and IntelliJ IDEA as our development environment, we've poised to create a versatile mathematical library. Git and GitHub managed our source code, ensuring organized contributions. Javadoc within IntelliJ IDEA maintained clear documentation, while Discord and Google Meet will foster team collaboration.

Our commitment to performance optimization includes the Strassen algorithm for matrix multiplication, enhancing speed for large matrices. Trigonometric accuracy, with up to four decimal places, has been achieved through the Maclaurin and Taylor series methods. These choices reflect our dedication to providing a powerful, precise, and accessible mathematical toolkit for developers across various domains.

# Chapter 4

# Application Domain

In this section, we delve into the practical aspects and real-world applications of the VMath library.

## 4.1 Purpose of VMath

The VMath library, built using Java, is designed with a clear purpose - to provide a robust and versatile mathematical toolkit for developers. Whether you are working on scientific simulations, data analysis, or any application that involves mathematical operations, VMath strives to simplify complex computations and enhance the efficiency of your code.

## 4.2 Target Audience

Understanding the intended audience is crucial for any software project. VMath aims to cater to a diverse range of developers, including students, researchers, and professionals involved in fields such as physics, engineering, finance, and computer science. By providing a user-friendly interface and a comprehensive set of mathematical functions, VMath empowers developers to focus on solving problems rather than wrestling with intricate mathematical implementations.

## 4.3 Key Features

### 4.3.1 Modularity and Extensibility

VMath follows a modular design, allowing developers to seamlessly integrate specific components into their projects. This modularity promotes code reuse and ensures that users can adapt the library to their unique requirements.

Efficiency is a primary concern in mathematical computations. VMath incorporates optimization techniques to deliver high-performance results. This is particularly valuable for applications that involve numerical simulations and data processing.

#### 4.3.2 Ease of Use

For developers, the usability of a library is paramount. VMath is crafted with well-documented functions, making it easy for users to harness the power of mathematics without unnecessary complexity.

### 4.4 Real-World Applications

The VMath library finds relevance in a multitude of real-world scenarios, including:

- **Scientific Research:** Assist researchers in conducting complex mathematical analyses and simulations.
- **Education:** Provide students with a practical tool for understanding and implementing mathematical concepts.
- **Engineering:** Support engineers in designing and optimizing systems through mathematical modeling.
- **Finance:** Enable financial analysts to perform intricate calculations for risk assessment and modeling.

As developers delve into the exploration of VMath, they encounter a versatile mathematical library with diverse applications. VMath stands as a testament to the transformative potential inherent in well-crafted software tools. Whether applied to streamline scientific research, elevate educational experiences, or contribute to various domains, VMath serves as a powerful resource in advancing computational capabilities and fostering innovation.

# Chapter 5

# Contributions

## 5.1 Numeric Discussion

As discussed in earlier chapter there are total 8 modules in this library. These are,

1. Algebraic Module
2. Trigonometry Module
3. Matrix Module
4. Vector Module
5. Bitwise Module
6. Base Conversion Module
7. Geometry Module
8. Combinatorics Module

In these 8 modules there are total 104 working public methods.

All the methods are created inside some relevant classes. There are total 15 classes but all the classes are not for use directly. A user is supposed to use **vmath** class for most of the functions. In this class all the classes are called and inputs are formatted with some factory rules.

There are total 2 interfaces in the project for dynamic references and open extension supportive calls.

Total 6 enumerations are working to provide correct input and unit typing to the system.

## 5.2 Individual Contribution

### 5.2.1 Adib Sakhawat - 210042106

#### Codes

1. Algebraic Module - 15 methods
2. Base Conversion Module - 18 methods
3. Matrix Module - 12 methods
4. Vector Module - 8 methods

In total 53 methods of this library is made by Adib Sakhawat.

#### Others

In the process of building project proposal and progress presentations are partially made by Adib and both proposal and final reports are partially made by Adib.

### 5.2.2 Tahsin Islam - 210042137

#### Codes

1. Trigonometry Module - 14 Methods
2. Geometry Module - 26 Methods

In total 40 methods are created by Tahsin in this library.

#### Others

Proposal Presentation is partially made by Tahsin. Readme documentation is created by Tahsin on github.

### 5.2.3 Takia Farhin - 210042117

**Codes**

1. Bitwise Module - 7 Methods
2. Combinatorics Module - 4 Methods
3. 6 enums

In total 11 Methods and 6 enums are created by Takia.

**Others**

Entire JavaDoc documentation is made by Takia in this library. All the presentations and reports are partially made by Takia.

At the end of the day we worked in this project as a team. Some of us focused more in coding part and some of us handled other important jobs in this software project.

## Chapter 6

# Useful Links

- Project Repository: https://github.com/sakhadib/vmath
- Project Readme https://github.com/sakhadib/vmath/blob/main/README.md
- JavaDoc Documentation of the Project: https://spldoc.sakhawatadib.com