Math interpreter Software

**ADVANCED PROGRAMMING CONCEPTS AND TECHNIQUES**

**Table of Contents**

[Introduction 4](#_Toc119002064)

[Aim 4](#_Toc119002065)

[Objectives 4](#_Toc119002066)

[Formation of Groups 7](#_Toc119002067)

[1. Non-functional Aspect: 8](#_Toc119002068)

[Literature Review 9](#_Toc119002069)

[Mathematical Visualization 9](#_Toc119002070)

[Mathematical Interpreter 10](#_Toc119002071)

[Software Testing 11](#_Toc119002072)

[User interface Design 12](#_Toc119002073)

[Applications of Math Interpreter 13](#_Toc119002074)

[Development Plan 13](#_Toc119002075)

[Requirements and Specification 15](#_Toc119002076)

[Project prototype development 15](#_Toc119002077)

[Requirements for developing the prototype 15](#_Toc119002078)

[Specifications of the software 16](#_Toc119002079)

[Analyzing the software based on the user demand 16](#_Toc119002080)

[First Sprint 17](#_Toc119002081)

[1. Product Survey: 17](#_Toc119002082)

[2. Design Test Strategy: 18](#_Toc119002083)

[3. Define test Objectives: 18](#_Toc119002084)

[4. Describe the Test Criteria: 18](#_Toc119002085)

[5. Give A Good Concept of Resource: 19](#_Toc119002086)

[Second sprint 21](#_Toc119002087)

[Reflection of the original objectives 22](#_Toc119002088)

[Main objectives 22](#_Toc119002089)

[Marking scheme guidelines 22](#_Toc119002090)

[Additional Marking scheme guidelines 23](#_Toc119002091)

[Representation of GUI 23](#_Toc119002092)

[Group management of the project 24](#_Toc119002093)

[Functional and Non-Functional requirements 25](#_Toc119002094)

[Mathematical software 26](#_Toc119002095)

[Third Sprint 27](#_Toc119002096)

[Fourth sprint 29](#_Toc119002097)

[Clean up and Finalization 31](#_Toc119002098)

[Bugs 31](#_Toc119002099)

[Final Developed Software 32](#_Toc119002100)

[Conclusion and Recommendations 35](#_Toc119002101)

[References 43](#_Toc119002102)

[Appendices 49](#_Toc119002103)

[Source Code 49](#_Toc119002104)

# 

# Introduction

An interpreter for maths visualizations software has been designed and a desktop solution has been created which can process expressions, perform and variable definition, and other tasks. A parser, lexer, and an executioner pass are used to process expressions commands and statements entered by the user. Visualizations of mathematical functions which are interactive have been provided alongside other elements such as function differentiation, zero crossings, and integration.

The size of a word in “MINI is 16 bits and the machine language of the language called MINI works on 16-bits word”. Arithmetic operations can be performed using the language and the size of the word is 16 bits. Floating point addition can also be performed using the software interpreter “MACRO/2 and MACRO/3”.

These interpreters have been developed to perform calculations using 2 or 3 words represented in floating point form. The software is mathematical software which is used to perform computations using numbers and the intermediate steps in the calculations are also displayed using the software. The software is based on the concept of an interpreter which converts the programs line by line into machine language and removes the bugs before moving to the next line of the code.

## Aim

The aim of the report is to develop programming software catering to the purpose of math software and to explore the different features of the software.

## Objectives

The objective of the report is as follows:

* To create a maths software with an interpreter
* To test the functionality of the software with appropriate test plan
* To develop a GI based math software with interpreter

For creating “Desktop Solution” to express evaluation expressions with defining the variables and function visualization for the two variables is an objective of this work. Another objective is to allow user to express their expression through the statements and commands using some regular “syntax” that is processed by a “laxer, parser, and execution pass” and to allow math interpreter differentiation and integration function. This math interpreter is useful for normal arithmetic operations (addition, subtraction, multiplication, division) for integer and floating types of variables. This interpreter is also helpful to do scientific operations like logarithmic function and other scientific functions according to the requirements. “Interpreter is a program which interprets code to machine code” here the objective is this math interpreter will take inputs from the users and return the operation result as a given input from the user.

**Background**

In the given assignment it is required to create maths software with an interpreter and the software must be a desktop application that can process expressions, define variables and define functions of two variables. The user should be able to enter statements, commands, and expressions using a custom syntax and then these are processed by a parser, lexer, and an executioner pass. Interactive and intuitive visualizations of maths functions will need to be provided (Bahrami and Soltani, 2020). Some of the other functions that can be added to this maths software are function differentiation, integration, and zero crossings. Some examples of similar software include Mathematica, Matlab, and Maple and some statistical software that are similar include SPSS, SAS, and R.

For the purpose of using mathematical software for exploring data, the resources available for computing are used and an attempt is made to enhance the algorithms and implementations. There are various components of mathematical software which include multi-precision numbers, rapid polynomial arithmetic, linear algebra, and other algorithms for computing large outcomes. The coding for these operations in a programming language for the software can be tricky and needs careful planning and there might be a number of bugs and errors encountered throughout the entire process (Cheung and Szpak, 2021). It is desirable for programmers not to write the code from scratch but to build on the algorithms present in some of the popular mathematical software such as Maple, Magma, and Mathematica. The programmers can make use of “open-source libraries” to add the desired functionalities in the code designed for the software.

Some of the challenges while building mathematical software can be the infringement of proprietary rights reserved for the code of other similar software and the programmer needs to design the software on his own. Also, there are a lot of errors and bugs encountered while developing software related to mathematical calculations. Creating a “Desktop Solution” to express evaluation expressions by defining the variables and function visualisation for the two variables is an objective of this work. Another objective is to allow users to express their expression through the statements and commands using some regular “syntax” that is processed by a “laxer, parser, and execution pass” and to allow maths interpreter differentiation and integration function. This maths interpreter is helpful for normal arithmetic operations (addition, subtraction, multiplication, division) for integer and floating types of variables. This interpreter is also helpful to do scientific operations like logarithmic functions and other scientific functions according to the requirements. “Interpreter is a program which interprets code to machine code” where the objective is this maths interpreter will take inputs from the users and return the operation result as a given input from the user.

**UI**

The UI can be used for a higher level of platforms or engines like “wxWidgets, WPF, Unity, Unreal, SDL, and many more” but here this UI is not a part of an “integral project” and here this UI is considered a “Supported Framework”

**Automated tools**

Here the automated tools ***“(lex,flex,yacc and many more)”*** are not “used until and unless the compiler includes any significant process”.

**“Tkinter” in python**

It is a standard “GUI” library for “Python” when it is used with “python” at that time it allows users to create fast and simple options for creating “GUI applications”. Tkinter allows an effective “object-oriented” interface in the “TK GUI toolkit”. Generating a “GUI application” by “Tkinter” is an easy task for this user must follow these steps. They are “importing the Tkinter module”, generating “GUI application main window”, “more than single widgets adding inside the GUI application”, insert “the main event loop for taking action against every event triggered by the user”.



**Figure 1: code script for “Tkinter”**

(Source: Collected from Vs code)

# Formation of Groups

In this project, the project team member decides to make mathematical software with some specific interpreter or language. In this study, the project team tries to execute the mathematical problems, expressions, and graphical problems with the help of the mentioned math’s visualization software. Basically, a math’s visualization software is used to calculate mathematical problems, simply saying it is an advanced mathematical calculator. It is mostly used in the concept of calculus but it is able to do any mathematical problems. Here to make this software team uses the mathematical visualization tool kit which is basically a group of visual and computational tools which is able to clear the concept of mathematical expression. An MVT (mathematical visualization toolkit) is very effective in mathematics departments, where the software needs lots of programming code, subscription fees, and also the installation process. In an MVT, there were set up lots of tools like plotting tools, numerical tools, linear algebra tools, differential equations tools, calculus visualization tools, and also includes a scientific calculator. After constructing the mathematical visualization tool kits, it is upgraded every year with the new application and also the new tools which brings the whole educational system basically the mathematical department to another level. The math visualization software comes up with a better opportunity for the students to understand the concept of software designing, and numerical analysis, and also improve programming skills.

To make the math’s visualization software first, the team has to decide on the appropriate language to make the software and then choose the acceptable platforms where the software runs smoothly.

Because the math’s visualization software can be made in any programming language like C#, python, c, and also java-based language. After that, there must use the proper variable, extensions, and functions to evaluate the mathematical problems. Here the software must be able to take the input or the expressions from the users and also be able to call a function for these commands and the function.

Lexar is used here to analyze or convert a set of characters into some lexical tokens and a custom syntax is used to process this also an execution passes and parser which is a program able to compile in the compiling process. Here the math visualization software basically understands the visualization of the mathematical expression in an automatic way. In the mathematical ground, there were a lot of applications or software that do the same thing like MATLAB, Mathematica, maple, and many others. So, it is difficult for the project team to make software for solving the mathematical expression. Visualization is not only developing the solution to the mathematical expressions but also inventing some relation between the mathematical objects.

## 1. Non-functional Aspect:

In the system engineering ground, there was a lot of use of non-functional software or aspect to do the operation of the system. Basically, non-functional software made with some functional requirements and also required some specific conditions where it dons its jobs. “Quality attributes” of systems are basically known as the non-functional requirements or aspects. Basically, non-functional software is an indicator that indicates or leads to the performance of the system which was done by some attributes (Silvis-Cividjian *et al.* 2021). And these attributes describe the system's availability, security, portability, and scalability.

Non-functional software points out the major problems in the quality of software systems and it is addressed the user's and developer's satisfaction. Identifying the unconditional software helps to save time and cost for the project. A non-functional aspect prefers some performance restrictions like security, response time, and related issues. It also helps to maintain the size and weight of the systems, availability, and system accessibility. In this aspect, there also include the interface constraints like how the system interacts with the environment and make it user-friendly (Vijayasree *et al.* 2022).

There was a lot of advantage to using the non-function aspect because it secures the software system with maintaining legal and authentic rules. They also mention the durability and availability of the system and also mention the quality attributes of the software. A non-functional aspect gives a good user experience making the software easy and minimizing the cost and also gives the security policy. But there were some issues to use non-functional software because of their security policy sometimes it harms the advanced software subsystem. And making these high-level software systems workable takes time and more paying costs than regular (Anwar *et al. 2019*). The major problem of the non-functional aspect is it is unchangeable after implementation in the architectural phase.

# Literature Review

Software visualizations are a process of development to generalize information. Basically, in modern days, there are so many websites and platforms available that provide solutions and techniques for mathematical problems. So, it is very useful for any student basically, to understand the solution. Many critical kinds of solutions are required, so the goal of the project is not only to build up the software but besides that, with the help an interpreter the exact explanation of the solution also can be provided properly. Visualizations are affecting every field to analyze the field of study with higher accuracy.

The visualization can be an individual or an overall sum of the specific field. Visualization is a process of any spatial thinking; it can affect the improvement of understanding of any concept. The reason for visualization has a specific need to energize the concept. In the study, in the recent study concept of the study, it has been seen that the visualization concept is highly introduced to energize the concept more smartly. People specifically are more energized by the particular study to visualize the exact thing like geometric size, three-dimensional objects, and all the scientific explanations in a new way (Kobayashi and Suzuki T.S, 2020). The geometrical or the three-dimensional size clears the particular geometrical concepts of the particular thing and besides that, the total objective, perspective, total organism, the three-dimensional analysis.

## Mathematical Visualization

So, in the research, the implementation of visualization techniques can be implemented in the study of mathematics. Many students are not interested in mathematical solutions because they can't understand the concept of visualization. In the new modern days, in the place of pen, pencil, and graph work, students are learning by digital technology, so mathematical visualization is a very structural and conceptual method. So, in this concept not only the solution but besides that, the whole concept and explanation of the process are represented in a visualization process. So, for the diagram, the graphical representation can easily be understood by every student.

So, the approach has a modification to grow the education purpose. Mathematical visualization is a process that can build up the ability of students to present the structural problem using diagram-based scenarios to support the solving skills technically. A sampling of the shape to express the problem stated. The main fact is that every individual needs a graphical representation to analyze the problem and configure the problem. It is a very helpful method.

Many kinds of visualization can be done like isomorphic, homomorphic, or any analyzing diagram-based techniques available, so there has to be a specific method for a specific application. In the isomorphic kind of visualization, the concept of isomorphs can be applied to abstract the main concept to sense the concepts of application in the several domains of mathematics. There are so many other concepts of visualization that also can be applied, like diagrammatic representation, schematic representation of the features, which can reflect the objects and effective thinking ability.

## Mathematical Interpreter

So, here the total work representation is done with the help of an interpreter, to reduce the complexity of understanding the solution. The interpreter can be of many types several interpreters can be used on a particular problem started (Wang *et al.* 2021). In the case of a python-based interpreter python language can be used to highlight the syntax, and the command can be given in a synchronized way. Many attractive features can be used for improvements. Many plots can be stated, so in that case the interactive plotting, libraries can be used and are functionally available in matplotlib. There is a high level of visualization reflected in interactive two-dimensional or three-dimensional plots. It is efficiently capable of plotting the curved or curved surface.

There are kinds of objective debugging tools that can be used to debug the specified object to solve that. Here in the local variable’s namespace, changing the values of the arguments of the function can be done. Many web browsers can be used to debug objects. The modules are mostly classes in a python framework. The documentation is done in many doc browsers. There is automatic synchronization to implement the well-designed interface to structure the whole documentation. There are block-coding applications are there to inspect the arbitrary in a python framework. Its design makes it easy to experiment with complex programs. There are animations of the features present in the software database to animate the structured solution for educational content purposes (Ganose *et al.* 2018). The performance is planned in a smooth orientation. The zooming and frames come so smoothly in a second.

There are explorations available in the module, which are imported into the global appearance. In the interactive side panel, the direction of the content is smoothly displayed. With the use of the editor of block coding, the objects can be a drag or dropped. The entire project can be exported for sharing purposes; the hosting platform is present on many social platforms (Lux *et al.* 2018). So, in the educational background, and more specifically in the mathematical or science portion the visualization or the visualizing the explanation in a synchronized way. So, here the source of the explanation is also visualized graphically. Many attractive features of it give more attraction to the student's mind.

To implement the various scientific problems, graph theory can be used, so that the analysis can be done with the help of different software. Gephi is a software-based application to do mainly the network visualization of any exploration of the data stated on the bases of the problem. So here, multiple software, tools, and libraries are used to feature the network dynamics, there are many graphical editors to structure the graphical representation perfectly and it is very helpful to understand the complex graphical solutions easily.

## Software Testing

Software testing is a very important issue to perform well. In the testing domain, there is some understanding of logic that is the basic development of the software (Cybulski and Horbiński T, 2020). The design of the software-based model is required depending on the mathematical solution required. First, the sets of functions have to be introduced to the models, to manage the relations of the different modules. The logic has to be set depending on the performing task or the probability of the logical representation. There are many kinds of problems like triangle-based scenarios, where the integer variables have to be taken as an integer. Based on the development strategy, we have to increase the level of exploration in the respect of developed embedded systems. To implement embedded systems such as different software. The development involves many parallel testing’s partitioned over test systems. The data have to be analyzed quantitatively.

Many complexities have to be formed to make it user-friendly. The amount of information has to be generated so that the test cycles are more frequent. The data set can be represented in an organized way to make the decisions (Dudley and Kristensson, 2018) There are complex or unit testing to implement the artifacts to explore various metrics to make it potentially reach. In a testing process, the quality of features has to be implemented in a supporting consideration.

The power is distributed and the automation is done based on customer demand. The development of virtual test systems allows the user to test the realistic conditions stated. The trafficking of the networking should be performed well and every task can be performed smoothly. Various kinds of models are relevant and the implementation of Tim has an effect to explore the data. The data collection has to be in the order of preference, and the addressing of the data should be automated and analyzed to predict the required decisions. In addition, I have to add some technical databases to frame their statically input. The total explanations are represented as modified as possible to be understood so smoothly based on different requirements.

## User interface Design

The user interface is like how it looks or the accessibility purpose to the users. It is a front-end-based application for friendly interaction with the users to use the software. The users can easily manipulate the interface from every corner and it is accessible digitally. The interface or part of the software must be designed to provide insight accessible to the software. The users can be graphically represented or text or audio based depending upon the structural presentation of the software. The user interface is both software or hardware-based both, the basic combination of the software or hardware bases of the attractiveness, how it will be accessible to the users. The time it takes to perform any particular task is very important to how the user is benefited smoothly in a framed time gap. All features have to be understandable in every aspect of the software. The screens of the interface should be consistent with the performance. The interface is partialized in two ways the bases of the command line or graphical to the users.

**Command Line Interface**

Basically, the command line interface is the combination of tools or models with the controls until it displays monitors (Besançon *et al.* 2021). The command line is one of the best choices for users or programmers as of its accessibility purposes. This kind of interface is very popular and the smoother performance requirement it can provide. Here, command text-based instructions are executed by their systems (Muniz *et al.* 2020). It has a text-based notifying concept, generated by the system's software; it is an instruction that is executable. The execution of the command is shown inline.

**Graphical User Interface**

In the Graphical User Interface here, graphically how are the interactions of the systems the interface of the GUI is actually performed in both the hardware and software parts of the system. Basically, it is more consuming than other interface lines (Ziatdinov and Valles Jr, 2022). It complexly designs the framework with perfect accuracy. It provides smoothie accuracy to the users; it has an area where the set of applications contents can be displayed in the manner of lists of features and it is resizable (Semenikhina *et al.* 2020). It has a documental interface, where viewing the preferences of the application based on the requirement. It has implementations in the form of menus, and icons. It has several application windows, where it performs (Yang *et al.* 2022). It featured buttons, and a textbook and the lists are given based on their requirements. There are huge activities that are performed with the implementation of the spiral model. The User analysis and the analysis analyze the task and implement the design.

## Applications of Math Interpreter

The high-level languages have to convert the machine code to understand the requirements. The Source of the programs is stored as machine-independent code, it can execute compactly fully, and accurately. There are several interpreters, and each of the operators is basically the invocation of complexly routine. Virtualization is very often used when the architecture is basically unavailable. Sandboxing can be done with the application of interpreters. Here, the self-modifications of the code implementation which is interpreted. There are different languages to use in the case of interpreters, like Python, PHP, and MATLAB. There are so many subjective programming languages that can be implemented in the case of modification of any software-related scenario.

Basically, the interpreter is like such a programming concept, is a high level of languages modifications without converting the machine code in the programming concepts, there are many methods that can be possible throughout the compilation, and obviously, an interpreter, combines the programming concepts in a unit modified way. So that the applications are so user friendly and features of it are so impactful by their performance. Here, in the concept of mathematical format, interpreting means converting the symbols and formulas into an organized and synchronized informational format with accuracy.

## Development Plan

Software development starts with the identification of the requirements that are required in the software that is being developed. All requirements are not equally important and the priorities are established using the MoSCoW method. There are four components to determine the priority of the different requirements in the software using this method. MoSCoW is the acronym for must-have, should-have, and could-have and will not have which are the four elements of the method (Liang, 2020). The MoSCoW method is used in various areas of applications like business and allows every person attached with a project to understand the tasks that need to be completed first and how these tasks will lead to increase in revenue, a decrease in operational costs, enhance the productivity or increase the satisfaction levels of the customers.

The MoSCoW method helps the development teams to have an idea about the resources and effort that will be needed for executing the project. The ideas about these aspects help the development team’s time management, improve the manageability of the project, increase the probability of the project ending within the stipulated deadline, and help to generate revenue. At the beginning of the project, it has to be ensured that the development team and the stakeholders agree on the objectives of the project and the parameters to determine the priority of the requirements (Elbadawi *et al.* 2021). There should be ways and means to settle disagreements among the stakeholders. Also, the resources that should be allocated to the different categories like must-have or should-have needs to be determined.

After these aspects are agreed upon by the different stakeholders the requirements can be assigned to the four different categories. The first kind of category for prioritizing the software requirements is called must-have and it involves all the requirements that are needed for the software to operate properly. The software cannot be built successfully without including the must-have requirements in the software (Monnappa, 2018). The legal standards will not be met by the software unless the must-have requirements are satisfied by the software. Also, the safety of using the software will be compromised unless the must-have requirements are incorporated into the software (Nikravesh, 2018). The final software will deliver proper results when the must-have elements are included in the software.

The second category of requirements that can be included in the software is called Should-have requirements and the priority of the should-have requirements is lower than the priority of the must-have requirements. These requirements are important for the software but are not necessary that is even if the should-have requirements are not included in the software, it will work. When the should-have requirements are included in the software the quality of the software gets enhanced.

Could-have requirements are the third category of requirements for the software being developed and the priority of the could-have requirements is lower than the should-have requirements (Chen *et al*. 2019). These include some desirable qualities that can be included in the software but are not very important.

The requirements that are not given priority and need not be included in the software are called will not have requirements of the software. By determining the will not have requirements of the software it becomes easier for the software development team to focus on those requirements that are more important and separate the unimportant requirements from the important ones (Mathew *et al*. 2019). The identification of the will not have requirements helps to restrict the software size to the desirable limit. Sometimes, a few requirements in the will not have category are reconsidered and included in the other categories for future software that will be developed by the team.

# Requirements and Specification

## Project prototype development

This project is about developing an interpreter or desktop software for math’s visualization, which will be industry-based. This implementation of the languages should be tested on ***“Java, C++, Python”***, etc. this project is progressing bit by bit. Hence, in the fourth week of the project, the job is to find all the requirements and specifications that are required for the project. In the start fourth week of the project, the prototype of the software for the math’s visualization is tested using the programming languages. The interpreter is required the compiling the high-level assembly languages into the low-label language. After the testing of the prototype for visualization of the math’s function. The coding for the project is done in the compiler and the prototype of the software is created but because it is a prototype so there is many flaws in the prototype to develop the prototype according to the project requirement (Aronson, 2022).

## Requirements for developing the prototype

In the case of developing an interpreter first, a laxer need to be created for getting the token from the input Programming that is done in the software. After that, a parser also needs to be created which is able to take those tokens and this should be done using formal grammar. The returns for the AST (Abstract Syntax tree) of the input program are created and finally, the interpreter takes all the AST from the input program and after taking the input interprets the code in some way. The raw string representation of the program is taken by the compiler as well as the interpreter and after taking the string representation the compiler and interpreter makes a parse of the input and put together meaning from that string. But writing the code of the interpreter for simple operations like addition and multiplication is very easy rather to other complicated functions in mathematics like differentiation and integration. In case of developing the prototype focus of the project should be on the laxing and praising of the input and finding all the common interpreters as well as the compilers that involve in the software.

## Specifications of the software

In the case of interpreting the software, it is rather a complex task. First is the “laxer” which is a very important part of the interpreter, laxer is responsible for converting all the character sequences into the sequences of tokens (Han *et. al.* 2022). After that next important part of the interpreter is the “parser” which is responsible for producing the AST (Abstract Syntax tree) from the sequences of tokens which is converted from the sequences of characters by the laxer. There are specific rules for operating the parser which is deducted by formal grammar. The interpreter's main function is to interpret all the AST from the source of the program without compiling the languages. The use of the interpreter is to convert all the high-level languages into simple machine-learning languages. This project's main purpose is to interpret all the mathematical functions and calculations using the software which will interpret these complicated functions in maths like integration, differentiation exponential, and trigonometric functions.

## Analyzing the software based on the user demand

The main purpose of the software is to calculate different mathematical functions and equations using complicated formulas. This software mainly takes the inputs from the user and then analysis the characters, functions, and numerical values after that, the interpreter have been designed for converting all the complex assembled languages to simple machine language so the computer can understand the input in a simpler way. The computer mainly understands binary languages like (0, 1) so without the interpreter, the computer cannot understand the inputs or the AST collected from the input program. The user's demand is a simpler user interface for the software so that all the operations can be done without any sort of complications. The prototype of the software is created for the user to understand the expectations of the users so the demand and the necessities can be found. The main demand is the time taken to solve math’s problems (Reynolds and McDonell, 2021).

The interpreter needs to operate fast so the time to solve any sort of math’s problem can be shortened. Next is the structure of the software which is a very important part of the software because user satisfaction is very necessary for developing software. If the user faces any sort of problems like bugs in the software and the user interface of the software is difficult to operate then the main motto of the project is going to be failed so the first objective should be to create software for easy to access by any user after that interpreter that used in the software should perform accurately without any sort of syntax error. The prototype created in this project is analyzed thoroughly to perform according to the user demand (Elbadawi *et al.* 2021). The main necessary objective of the project is followed according to the given instruction. This software is cable of carrying out all types of mathematical operations like addition, subtraction, multiplication, division, integration, differentiation exponential, trigonometric functions, etc.

# First Sprint

After implementing the proper language and lots of code with variables and functions the project team came to take some tests the check the software manually. After understanding the logic and expressions which give a brief clean and better coverage of the quality of the system to develop the software system. Here, understanding and implementing mathematical calculation is the main key to developing the software for the developer (Xi *et al.2020*). Developers must need to clear all topics from discrete mathematics which sharps the concepts of designing test cases. Basically, in this software developer develop the language after giving the input it can show the desired output for a triangular problem the user has to put the input or take the integers a, b, and c and the software shows the type of triangle. To test software there were lots of steps covered by the developer which includes test strategy, objectives, timetable, and also resources. Here the developer follows some steps to test software which are.

## 1. Product Survey:

Here analyzing the product is the first and the most initial part of the testing process because the developer has to need all the information or developing background before testing. Developers have to survey thoroughly the needs of the project and also keep an eye on the customer expectation to test the software based on it. The testing team also needs to know before testing who and where the software is being used so that they can improve the durability and modify it on the basis of the environment. The entirety of the concerned product gets thoroughly looked into for the purpose of getting all of the attributes in this regard. This includes the very minute number of details along with the larger ones, so as to cover the full spectrum for that matter. In this manner, this process takes place on the whole.

## 2. Design Test Strategy:

In these steps, there was a lot of complexity and issues in making a proper plan to test the software. The test manager creates a documentary plan for the testing teams where all the testing objectives and the required resources are present to test the software. The test strategy is too difficult to handle because of high levels but it gives a good concept of the testing endeavors and the estimated cost. Here this study briefly mentioned the scope of the testing manuals and how to identify the type of testing. Here all components of the system hardware and the software are tested and give users and developers a piece of proper information and good confidence. This process there briefly indicates the types of testing which the developers have done which are unit tests, API tests, integration tests, system tests, install tests, uninstall tests, and many others. It gives alerts on the risk of documentation and all issues. Test logistics which is one of the parts of this step helps the test members with the further testing process. In the test, logistics mentioned who and when the test can occur and also understand the customer's needs and desires regarding the system quality.

## 3. Define test Objectives:

Here the developer gets the whole achievements of the testing results. Here testing team finds the defect in the software and they try to make the software bug free. At first testing team members note down all the features of the software which is basically about the functionality and performance of the systems. On the basis of the components and the features, the teams include the aims and results of the testing software and also check the software is working without giving any errors and also meets the customer's expectations.

## 4. Describe the Test Criteria:

Basically, test criteria are some rules of the test procedure. There were basically two types of test criteria suspension criteria and exit criteria. In the suspension criteria, the main motto is to notice that the software is running accurately without giving any errors and if it gives any errors then it is suspended immediately and wait6 for the solution. If in the test case failed percentage of 30% to 40% then the test will be suspended as per the test criteria. After a pass from the suspension criteria, the software enters the exit criteria which are basically a successful remark for the test manuals (Tebes *et al. 2022*). Here the exit criteria look into the result of the test cases and if it rates above 95% then it passes the test. Here the run rate is to be 100% unless the test team has to give the reason behind the rates. Here are the run rates between the numbers of total test cases of the types of tests. If there are a total of 120 test cases and the tester takes 100 test cases then the rate is nearly 83% which is not good enough for software testing. These pass rates are dependent basically on the project background and the test embers also chose a high pass rate for better results.

## 5. Give A Good Concept of Resource:

The project team has to be informed by a good concept of the resources of the project. Having a good concept of the resources of the project help to resolve the number of employees, types of equipment, and also such related things, and the test manager is also capable of giving the exact timetable and the estimation of the project. Here the test administration, test developers, SQA members, test managers and also testers all are under human resources which is keep an important role in the software testing period. For testing, there is another important and needful resource is system resources which are servers, testing tools, network connections, desktops, and many other things.

**Testing Environment:**

The environment is one of the important parts of testing the software. Here it needs to choose the proper environment and proper marketing of the software. To execute this there was a need for better cooperation between the test team and the developing team (Kubica *et al.* 2019). Here the team manager prefers the environment for the testers and also for the software because there was some complexity in the software manufacturing. A good business environment can make the software more usable and available in the market.

**Timetable and Estimation:**

In these steps, the project manager needs to clarify the concept about the team's members and the taken time behind the work, simply saying giving a work schedule to the testing team members. Here project manager gives the members the paper work and gives an estimate of the time to do the work. After that, the project manager strictly mentioned the deadlines of the project. And also alerts about the project risk. Here it helps the project manager to understand how much extra time needs to complete the testing.

**Test Result:**

In this step, the testing team members get a piece of brief information about all the documents, tools, and other materials. There were mostly three different test results before testing, during testing, and after testing (Olsina *et al. 2022*). Before the testing result, there include some documents about the test plans, test case, test design, and the test scripts, data, matrices, and simulations provided in during the period. After that when the test is done there was test results and related reports which are given by the test team members. And they also provide installation guidelines that ease to use of the software for the users and the customers.

# Second sprint

The second sprint of the software that is being developed for the purpose of performing mathematical operations has been finalized. The process begins with the planning for the software where the requirements and the functionalities that need to be documented for the development of the software are done. The software that is built is essentially mathematical software that is required to perform various mathematical operations. The client is asked to specify the different requirements that need to be incorporated into the software and then the requirements are documented. The client is shown the requirements that are asked for in the software and the client finalize the requirements after looking at the Software Requirement Specification. This process is lengthy process since the client might not be satisfied with the requirements set out in the SRS document and may need to alter some of the requirements (Coble and Bright, 2019). This process continues for multiple iterations since the changes might be incorporated into the SRS document by the development team and reviewed by the client who might want further alterations. The most important step of the software development is the finalization of the SRS document.

After the requirements are finalized from the client the feasibility analysis of the software is done where it is checked whether the software is feasible to be built in terms of economic requirements and the technical requirements of the software. Then the design of the software is created where the development team analyzes the tools required to develop the software. The outcome of the design of the software is a blueprint using which the software will be developed.

Once the design of the software has been prepared and a blueprint for the software is ready, the real development of the system begins. The different modules of the software are formed and then the modules are built using programming languages. The aim of the implementation is to break down the whole software into simpler and more manageable tasks and then the modules are developed separately. Once the different modules have been prepared they are unit tested that is the individual units or modules are given certain inputs and it is checked that the response to the given inputs is appropriate. The separate testing of the individual modules helps to identify the bugs in the software readily and fix the bugs. If the modules had been combined and then the entire software was tested it would have been very difficult to identify the errors in the software.

Once the individual modules are tested successfully the modules are successive integrated and then further tested. Only when the integrated module is tested successfully another module is combined and then tested. This process continues till all the modules are combined and then the entire software is integrated. Even after the final integration of the software it is tested and errors are found which need to be carefully handled. The software is tested thoroughly using a comprehensive set of values and then the software is given to the client for further testing by the client (David *et al.* 2021). When the client is satisfied with the performance of the software he accepts the software and the development team also provides solutions to the client if there are problems in the software when it is used in the future.

In case of the mathematical software the different models are tested initially using various sets of inputs and it is checked that the output of the modules is satisfactory. Then the modules are integrated and the entire software is built which is further tested using various sets of inputs. The outputs obtained from the integrated software are checked for the appropriateness of the required outcomes needed for the software. Then the software is finalized and ready to be delivered to the client.

# Reflection of the original objectives

## Main objectives

This project's main objective is to create desktop software for evaluating complex mathematical functions and expressions, as well as define all the functions and the variable also the visualizations of the two variables of the functions. The software should be capable of taking the user inputs like mathematical expressions, statements, and mathematical commands with the help of mathematical functions and it is done by using the custom syntax which is processed with the help of the laxer, parser as well as executional passes. The visualization of mathematical expressions and functions, which is Intuitive as well as potentially interactive, needs to be provided (Klabnik and Nichols, 2019). The Additional functionality is also a part of the zero crossings and functions like integration and differentiation. Examples and the name of some math’s software that is used by professionals are MATLAB, Maple, Mathematica and some

A variation of the themes that are acceptable for the users is statistics software and some Commercial examples of this software are SPSS, SAS, R, etc.

## Marking scheme guidelines

Marking scheme guidelines is a grading system in the basics of the certain guidelines that should follow along with the development of the project. First is creating a very well-evaluated and perfectly documented software or prototype of the software the coding for the interpreter should be clean and the product must contain any unnecessary major bugs like the software get freeze or get crashed or showing any sort of error in the output or any kind of unintentional and inoperative functionality, etc. (Mozelius and Humble, 2022). The intuitive GUI as well as the evidence in the case of adequately using iterative development and incremental development must be presented across the total cycle of the development. Next is the division of the task for the project should be fair and equal between the group members and these are the minimum requirements for getting the honors mark(above 60%).

## Additional Marking scheme guidelines

In case of getting additional marks, the software must have deep functionality which is able to perform successfully. Next, the UI/UX should be excellent and the optimization of the code must be smooth in order to get additional marks, the use of many language paradigms can be very helpful for extra marks. Marks will be deducted for the practice of poor programming if the programming has non-customizable as well as non-scalable architecture along with hard coding, etc. (Chakraborty and Elzarka, 2019). Also, if the Specifications of the project do not meet the requirements, then marks will be deducted. Marks also can be deducted if there was a major bug problem reported in the software as well as if the time management of the project including the division of the task is very poor. One important thing is there to remember that the representation of the project's basic and stable so using unnecessary non-functional additional features in the project could be a major reason for the deduction of the marks (Chis *et al.* 2018).

# Representation of GUI

In the case of developing software, the GUI (Graphical user interface) is very important because without a user preferable GUI there is no meaning in developing the project. Because

GUI is a very user-friendly environment that allows the user to perform any sort of action that they can think of without being given any prior knowledge in the programming field, some common examples of the GUI are “Windows” operating system, “MacOS”, and “Android” environments, etc. (Kirsan and Insanittaqwa, 2022). GUI is mainly used in many computers and electronic devices to provide the capability of users to easily manipulate graphical icons; scroll bars windows, tabs, menus, and cursors. It became the standard parameter for user-center design in the field of programming and software application. Nowadays GUI is used in many fields not only in computers but also in the case of mobile devices like portable media players, MP3 players, smart phones, gaming devices, office and industrial controls, smaller households, etc. A GUI must have an attractive and pleasing outlook for the user's eyes. It should allow the users to try many different options according to the user's preferences. The GUI should be easy to use and it should also have suitable color key portions, as well as the GUI, must include easy words so the user can comprehend the meaning properly (Mathew *et al*. 2019).

# Group management of the project

In the case of big projects handling the management of the project is very important because without a proper management system the project cannot be completed in time as well as the quality of the project will be very poor. The first thing to do in the group project is to divide all the tasks among the group members with fair and equal distribution. After that to complete the project successfully and within the time there are some steps to follow first is Initiating Phase where the discussion of the business value and the feasibility of the projects takes place. Next is the Planning Phase where the entire planning for the project from top to bottom is discussed is involves the collection of the data and the resources for the project, necessary requirements and specifications of the project along with how the final deliverable of the product can be done (Sánchez *et al.* 2019).

Next is Executing Phase of the project this is the longest part of the project where all the coding and programming-related work for the math software and interpreter is done using the resources and data that have been collected by overcoming many project constraints and the most of the time is getting invested in this part of the project. After the Executing Phase next is monitoring and controlling where the project manager's job is to ensure the tracking performance and the project progress is going according to the plan and whether all the necessary responsibilities are fulfilled or not and the time duration as well as managing the project budget within the range(Tyralis *et al.* 2019). Next is the final phase of the project which is called the closing phase of the project where the testing for the final product or the software is done the evaluation of the budget for the project is calculated as well as the project duration should be monitored. Because after completing the prototype there will be many loose ends to tie up in the project like non-functional features needing to be removed from the product, the coding must be clean for the interpretation of the software, and all the bugs must be removed from the product as well as make software features easily accessible for all users. The software GUI and UI must be basic and simple without attaching any complicated features to the prototype. Finally follow all the marking scheme guidelines to get a score for the project.

# Functional and Non-Functional requirements

The requirements of the software that is developed can be either functional requirements or non-functional requirements. The assessment of the different requirements of software forms a critical part of the success of software development. The functional requirements are the features that the customer asks the development team to include in the software. These requirements can be verified by a set of inputs given to the system and the outputs obtained from the system (Shanmugamani, 2018). Thus the software must contain a collection of functional requirements as demanded by the customers from the software development team. These requirements can be directly verified in the final product formed at the end of the development phase.

The non-functional requirements are the quality parameters that should be adhered by the software that is being developed (Lambropoulos and Dimakos, 2022). The non-functional requirements are also called non-behavioral requirements of the software. Some of the issues that are included in the non-functional requirements include Security of the software, maintenance of the software, Portability of the software, Reliability of the software, Scalability of the software, Reusability of the software and other such features.

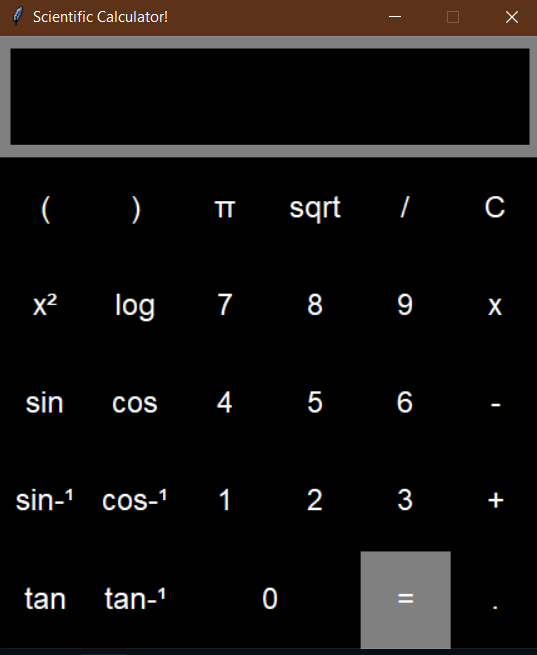
One of the functional requirements of the software is the user interface of the software which specifies the appearance of the software and the interactions of the users with the software. The user interface of the software is based on front-end development of the software that is developed. The user interface of the software can be modified by the user of the software according to the needs of the users (Han et al. 2022). The simplicity of the user interface is a key feature that must be ensured to make the software perform its function properly. The users can use the software with added confidence and ease if the software interface is easy to use. New features should be added to user interface only if there is urgent need for these features.

The user interface should be consistent and the design of the user interface should be homogeneous throughout the software to make it easier for the users to handle the software. Consistency of the interface helps to avoid confusion and ambiguity. The font size, color, background color, and other features should be the same throughout the software. The user interface of the software should be easy to understand and use for the users of the software. The buttons and other features of the software need to be properly labeled for the convenience of new users (Sánchez *et al*. 2019). There should be coherent icons used in the software for the convenience of the users.

The interface of the software should be able to prevent the user from doing any unwanted operation using the software (Karlsen *et al*. 2022). The software should be designed in such a way that the user can deal with unfortunate operations done on the software and seek help for using the software. The software interface of the mathematical software can be of two types namely Graphical User Interface where the user can access the different functions of the software using visual elements like pictures and icons. The software also has a Command User Interface where the software can be used employing the command prompt of the computer.

# Mathematical software

The software prototype for the mathematical software has been created using the programming language Python on the software platform Visual Studio Code. The software prototype is a calculator which can perform the basic operations of addition, subtraction, multiplication and division using different numbers.



**Figure 2: Python Code for the different functions which perform the different mathematical operations**

(Source: Obtained from Visual Studio Code)

The library that has been imported to perform the different operations using Python is called tkinter. Three functions have been defined as shown in the above figure pertaining to the calculator designed using Python. The three different functions are show (), clear () and solve (). The Math Interpreter Calculator has been shown alongside the code for the software.

The parameters for the first function named show are self and value and this function is about displaying the numbers that are input to the software and the result obtained from the software

The parameters for the second function called clear is self and this function pertains to clearing the previous inputs entered into the calculator after the previous input has been entered and the results have been obtained. The parameters for the final function named solve is self and this function pertains to the calculation of the results after the data have been inputted into the calculator. The final result after the operation has been performed has been provided in the variable called root. The root is then displayed and the whole process continues with the help of a loop. The different operations that get performed when inputs are entered into the program are also shown in the calculator interpreter interface shown alongside the code.

# Third Sprint

The priority of the assignment is to create software that is related to the performance of mathematical operations using an interpreter. Mathematical operations can be calculated manually but when the calculations required for the operations are large then computer programs are used to develop software which can perform the operations in a fast and easy manner. An interpreter converts the program codes line by line from the source code into machine language which can be processed by the computer. Each line of the code is checked and then errors are identified and the interpreter translates the following lines only when the code in the present line is free of all errors. The entire program is not interpreted in one go but it is interpreted line by line.

The MoSCoW method is relevant in the given context and it has various components inherent in the method which are called Must have, Should have, Could Have and will Not have. The MoSCoW methods help to have clarity about the amount of resources required for the individual project elements.

The Must-have component in the MoSCoW method outlines the requirements that are mandatory for the project and they are non-negotiable. Even if the project is completed within the stipulated deadline it is meaningless to finish the project without meeting these requirements which are mandatory (Hankeln *et al.* 2019). The product prepared at the end of the assignment will not be considered to conform with the standards unless the Must-have requirements have been successfully met. It is usually found that the product is not very useful unless these requirements have been satisfied.

The Should-have requirements are also required for the success of the product developed but are not mandatory in the strict sense. The product will continue to perform without these requirements but the performance of the product will improve manifold if the Should-Have elements are included in the project (Kraft, 2020). For instance small bug fixes, performance enhancements and other additions are considered as Should-have components of the product.

Another set of elements that are considered in this method is called Could-Have requirements which are not as important as the Must-Have and Should-Have requirements. The Could-haves are desirable but not mandatory for the project (Markova *et al.* 2018). The impact of not including these elements is not fatal for the performance of the product being developed.

The Will Not Have requirements are the requirements that are not required for the software that is developed using programming languages. By outlining these requirements the focus falls on the other requirements that need to be included in the software.

After all the Must-have requirements and Should-have requirements are included in the software requirement specification document it is sent to the client for verification. The client checks the requirements of the software and then gives the permission to proceed with the development of the software (Markova *et al.*  2019). The software that needs to be developed pertaining to various mathematical operations is divided into smaller modules which are created and separately tested. The modules are then integrated and the entire software is then formed and then the software is tested as a whole. The software might encounter many errors while it is being tested and then the software is modified and the coding is rectified carefully to remove the errors present in the software. This process proceeds continuously and the client is consulted to add any other requirements in the software. The additional requirements are included in the software and then the software is further tested. The third sprint is thus prepared and the sprints are draft versions of the software which are created and then inputs of the clients are taken and further changes are added into the software. After each change to the software, the application is tested for errors and for each of the sprints a modified version of the software is formed.

The third sprint of the software begins with the verification that the objectives for the previous sprints have been met and the MoSCoW method is used to categorize the requirements of the software into Must-have, Should-have, Could-have and will not have requirements (Neymotin *et al.* 2020). New features in the software can be added in the Third sprint of the software with respect to the second sprint of the software and then integrated to obtain the modified software. The different sprints are essential for the implementation of the software since the client can ask for changes in the software and new features need to be added to the software. The process of software development is a dynamic process and many alterations and modifications need to be done before the final product is built and the feedback of the client plays a crucial role in the making of the software. After each modification is done, the software is tested with a set of appropriate inputs to ensure that the modified version is working properly and then client feedback is received for confirmation of the change.

# Fourth sprint

The fourth sprint of the software application pertaining to mathematical operations is developed and implemented. The software can be developed in many possible programming languages and some of the programming languages which can be used to build the software include Java and Python (O’rinov Nodirbek Toxirjonovich and Abduvaliyevich, 2020). Both Java and Python are object oriented programming languages and support various functionalities which can help to build the software.

Fourth sprint of the software implies that many versions of the software has already been created and then modified. With each successive version of the software, some changes have been added to the software that has been created and modifications have been made (Yorgancı, 2018). The clients are usually not satisfied with the initial versions of the software and ask the developer team to constantly innovate and include additional features in the software.

In the beginning it must be ensured that the final software that is developed after the different sprints meet the requirements set out originally at the start of the development since many changes are made and modifications are implemented in the software and it becomes very easy to overlook the original objectives of building the software. The functional requirements that need to be added to the software will need to be revised and changes in the functional requirements should be finalized properly before the fourth sprint of the software is prepared (Ozudogru and Ozudogru, 2019). The results that have been achieved with the previous sprints of the software needs to be evaluated carefully before adding any new features into the software. The developers should have clarity about the features remaining to be implemented in the software and should form a list of these features which are yet to be added.

This stage is important since the developers have to assess the requirements and identify the requirements that need to be present in the final version of the software. The unwanted features that have been included in the software in the previous sprints of the software will need to be discarded and the appropriate requirements will need to be included in the fourth sprint of the software (Papadakis *et al.* 2021). The MoSCoW method is used in this sprint also to classify the requirements which are mandatory, that is Must-have, Should-Have, Could have and may not have requirements. The MoSCoW method is relevant in this sprint also to identify the requirements that need to be included in the software.

Some of the features that can be added to the software include that the software should show the intermediate results of the calculations performed pertaining to the different mathematical operations (Van Merriënboer *et al.* 2018). Multiple users should be able to sign in to the software and use the software simultaneously for different purposes like plotting of mathematical curves and large calculations. The software has a provision for registration for the users who need to create their accounts and finalize their passwords for using the software (Putri *et al.* 2020). These help to enhance the overall security of the software and then various mathematical operations are included in the software. The functionalities of similar mathematical software are also considered like Maple and Magma and included in the design of the software. Data structures are used for storing the data used for calculations and algorithms that help to compute the results of complex mathematical processes have also been included in the software. There is provision to plot the curves of the mathematical and trigonometric functions in the software.

The time taken for the implementation of the different mathematical functions has been reduced in the fourth sprint of the software. Some of the less important requirements of the software have been excluded from the design of the software and the necessary functionalities have been included in the software (Ramsay-Jordan *et al.* 2022). The purpose of this sprint is to optimize the performance of the software by removing the unnecessary features of the software. The cost of developing the software needs to be optimized given the satisfaction of the client.

The development team is for the purpose of designing the different modules of the software and the integration of the modules to form the whole software. The Unit testing and the integration testing of the whole software must be carried out by specialist testing teams. This is done since the development team spends a lot of time developing the software and sometimes it becomes difficult for the development team to identify their own errors after spending time and resources for designing the software (Tamur and Juandi, 2020). The separate testing team will not overlook the flaws in the software as they are fresh without the burden of developing the software and perform better in this respect. There can be scope for unit testing and integration testing by the developers but the software testing has to be performed by the testing team.

Also software cannot be fully perfect and repeated testing for errors using different inputs and testing the response of the software will lead to better quality in the software. Testing should begin as early as possible and unit testing is crucial since it is easier to remove the errors in the software in the unit testing phase. The more important sections of the software have to be tested earlier with respect to the other parts of the software since they play a more critical role in the software. The testing has to be done keeping in mind that the deadline for the software has to be met (Triana and Zubainur, 2019). A test plan is developed for testing the mathematical software where the possible inputs and the expected outputs of the software are listed. It is best if the testing of the software is done using surprising values to check the response of the software against all kinds of inputs provided to the software. The tests that are done should be properly documented with the inputs provided and the results obtained from the software. After each testing of the software the results of the tests must be properly analyzed to understand the performance of the software. The software is then provided to the client for his opinion and if needed further modifications is made in the software.

# Clean up and Finalization

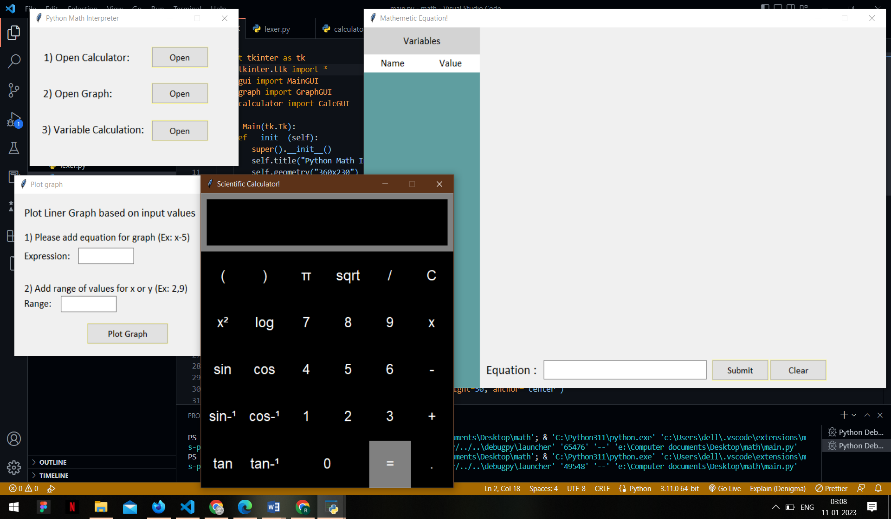
## Bugs

Bugs are errors encountered in the software that may lead to the software not meeting the requirements and not operating successfully. Software is tested through a set of input values and the results obtained are analyzed to check whether the software is working properly. If an error occurs in the working of the software when it is being tested, it is called a bug. Bugs can occur due to lack of communication among the members of the development team when the software is being developed. Due to a lack of proper communication, there can be misunderstandings regarding the working of software. Sometimes the customers lack clarity about the way their software can serve them and are unable to pass that information to the development team (Tyralis *et al*. 2019). These circumstances can lead to the occurrence of malfunctions in the software and then they are removed very carefully.

The requirements of the customers for the software continue to change and this often leads to confusion among the members of the development team and the testing team and this can lead to the malfunction of the software. Software that is developed is a complex system with multiple components like User Interface, database, different modules, and other such features and the development team is always under pressure to deliver the software within the stipulated time, incorporate the changing needs of the customers and deal with other distractions that might occur as the software is being developed. The software development team needs to make additions and alterations to the software and all these factors can cause bugs to occur in the software when it is being tested or developed. There might be issues with the design of the software and the overall architecture of the software.

Reallocation of resources, redoing or cancelling some aspects of the software, and alterations in the requirements of the software can cause errors to creep into the software. Sometimes confusion can be caused if new developers are asked to join the development team without prior knowledge of the development already done (Tirkolaee *et al.* 2020). Also, errors can be caused if the standards pertaining to the coding of the software are not strictly adhered to. The bugs are discovered if the software does not produce the required results or the software does not run properly. The software might stop working or may show errors that are difficult to locate among the different modules of the software (Bagheri *et al*. 2019). The errors or the bugs are essentially discovered during the testing phase of the software. The tests can be of different types like the separate and smaller modules are tested separately in the beginning. Then these modules are integrated one at a time and for every integration, the new software formed is tested and this testing is called integration testing. After the whole software has been integrated, software testing is carried out. By increasing the number of software tests the bugs in the software can be detected and then they are carefully removed from the software. Some of the bugs that have been discovered in the mathematical software include problems with the user interface (Alyahyan and Düştegör, 2020). Some tests show that the user interface of the software crashed and the software momentarily stopped working. These errors had to be handled and then the software was validated by the customer.

# Final Developed Software



**Figure 6: View of developed GUI math interpreter with codes**

(Source: Acquired from VS Code)

The math interpreter calculator has been developed with a proper GUI framework view using Python language and “TKinter” module.

So here, the design of the GUI framework of math interpreter calculator has been done successfully with tkinter. It can make some understanding of the software and the module to develop a GUI application (Sarkar *et al.* 2018). Hence, it can help to extend some new techniques with a history tab to find out the previous calculation. Adding a background different image can also make the calculator more attractive. This future prospect can help to make more efficient math interpreter. In the first step math interpreter ask user about choosing one option from three (Open Calculator, Open Graph and Variable Calculation). Second user click his option and open Calculator or Graph or even perform Variable Assignments.

# Conclusion and Recommendations

The aim of this report is to describe the work done to develop programming software to perform various mathematical functions and understand and analyse the different features of the software. The main purpose of this report has been to create mathematical software having an interpreter, to test the software using a collection of inputs and analyse the outputs, and to form a Graphical User Interface of the software alongside the interpreter for the software. Background of the software has been provided in the report where the different functionalities of the software have been provided and some of the critical features of the software have been described. The challenges of developing software have been described in the background section of the report.

The formation of groups and teams in the context of developing the software has been discussed in detail in the report. The difficulties that were faced by the development team while developing the software were discussed in this section. The non-functional aspects that were included in the software were discussed in the report alongside a literature review of the assignment. Adding a background different image can also make the calculator more attractive. This future prospect can help to make more efficient math interpreter. The aspect of mathematical visualization has been discussed in the report where the feature of mathematical visualizations in the software has been dealt with. The concept of a mathematical interpreter has been discussed in the report where the differences between a compiler and an interpreter and the interpreter developed for the software has been covered in detail. The aspect of software testing and the procedures followed while testing the software have been discussed in the report. The different tests like unit tests, integration tests and other kinds of tests have been discussed at length.

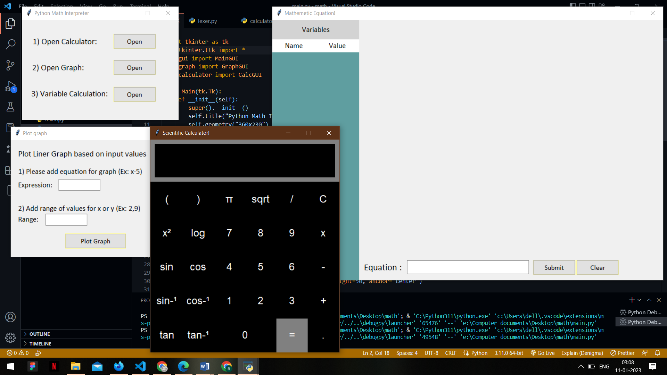
The design of the user interface has been discussed for the software that is developed to perform the basic mathematical operations of addition, subtraction, multiplication and division. The two categories of interfaces have been discussed namely the command line interface and the graphical user interface. The graphical user interface uses various icons whereas the command line interface is used by employing the command prompt. Also, the different applications of the math interpreter have been discussed at length in the report. The different requirements needed to be implemented in the prototype have been provided in the report. The specific features of the software that is developed have been provided in the report. The different sprints of the software have been described in the report which is the draft versions of the software.

The first sprint is developed and the description of the features of the first sprint has been provided in the report. The product has been surveyed and the test strategy of the software has been formed. The test objectives of the mathematical software have been defined in this section of the report. The test criteria of the first sprint have been provided. Then the second sprint of the software has been described alongside a reflection of the original objectives set out at the start of the research. Then the third sprint of the software has been provided. The fourth sprint of the software is described where the requirements for the final draft of the software have been refined. The MoSCoW method has been used to categorise the requirements into the essential requirements and other additional requirements of the software.

The user interface that has been designed is easy for the users and the functionality of the software complement the software developed using Python programming language. The coding of the software has been done on the software platform called Visual Studio Code.

**Improve scientific calculator**

This is a updated “GUI” for the “math interpreter” in this updated interpreter the functions updated are basically focused on scientific operations such as logarithmic operations, integration and differentiation and for this operation and to update it “Tkinter” and “Automated tools ” is used.



**Figure 7: Scientific Math interpreter after update**

(Source: Collected from vs code)

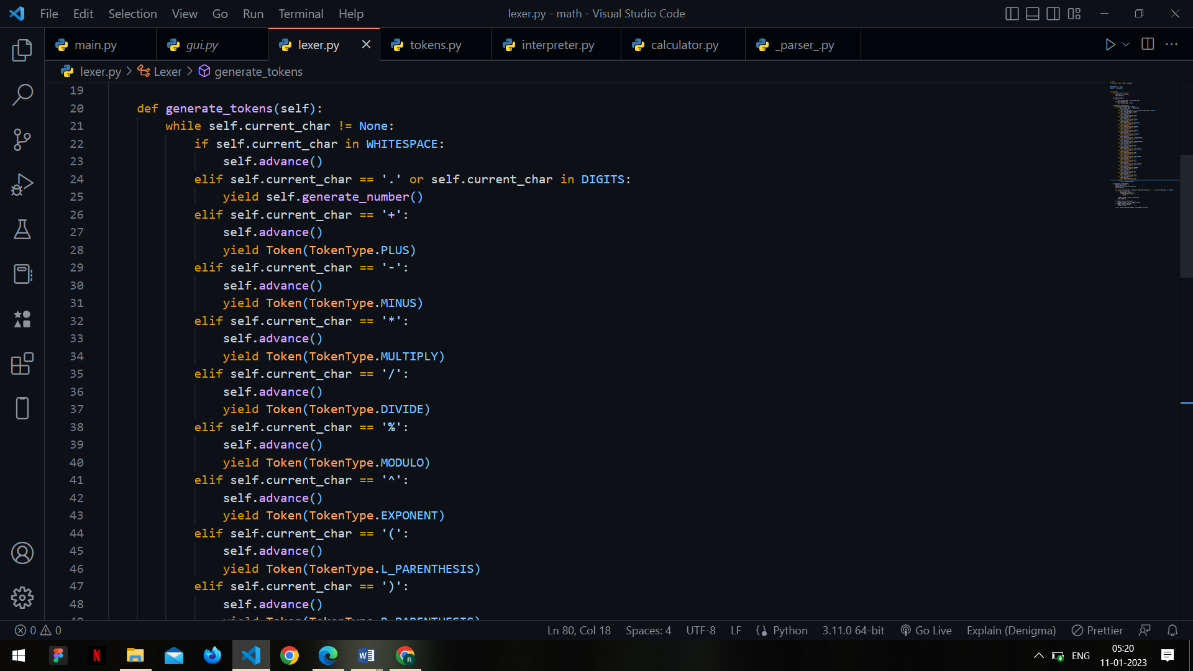
This figure is the updated figure of scientific math interpreter and it has the option like trigonometric operations and logarithmic functions and various other functions. This updated “GUI” allows users to switch between the standard or scientific for the better user experience according to their requirements. The scientific calculator is presented in the above image by the help of codes presented in Vs code. The analysis of colours for each button is white. The background colour of each and every button in this calculator is black in colour.



**Figure 8: Configuration of input functions**

(Source: Collected from vs code)

The following diagram is for configuring input functions and the following figure is focussed on the operation based on number entering, sum of the total.

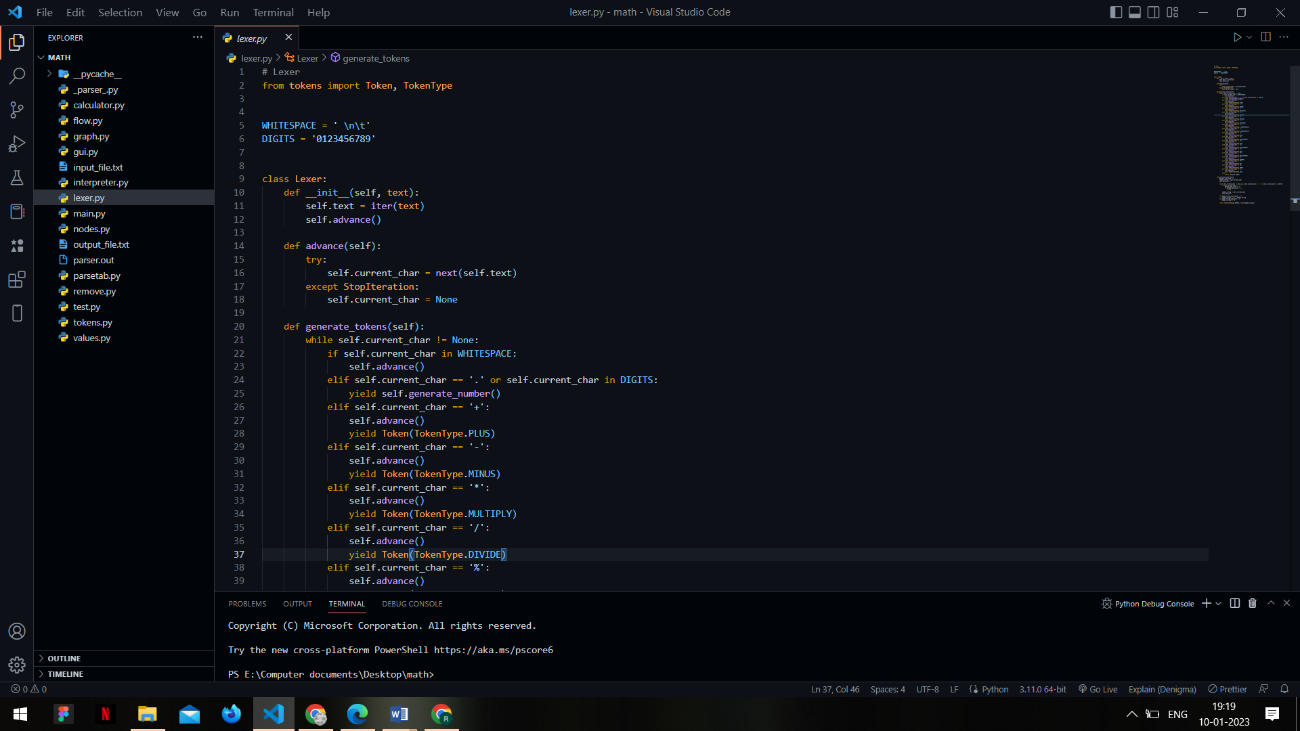


**Figure 9: Configuration of input functions**

(Source: Collected from vs code)

This is the figure used for describing the configuration process in the coding the function named “Valid function” and defining “operation” is used in the section following portion and the image is taken from the Vs code.

The calculator which gives the single values results and also the result required for the scientific purpose are being made with the help of python programming language. The calculator can be made advanced with the help of lexer and tokens that can be added with the help of python programming language. The features of the lexer and tokens gives that it can able to add the features to the calculator such that it can able to do the calculation of multiple values and multiple expression at one time. The features to add the calculation at one time helps to save the time and also the extra steps required for the calculation purpose.



**Figure 10: Description of the buttons of calculator**

(Source: Collected from vs code)

The working that has been done for developing the calculator is such that the writing of the programs is being done on the python which helps to write the commands easily. The library of the pythons which has been taken for the calculator is such that the tkinter and math library is being used. The scientific calculator features is also being added in the calculator which helps to reduce the effort to calculate the complex scientific calculations which are such that it contains the logarithmic problems and the problems which includes sin, cos type. The scientific problems are very vast so this type of calculator helps in reduces the time and effort of the student or the person who is doing it. The colours of each button are made black by applying the codes. The background colour and the font colour is made black and white.

# References

Alyahyan, E. and Düştegör, D., 2020. Predicting academic success in higher education: literature review and best practices. International Journal of Educational Technology in Higher Education, 17(1), pp.1-21.

Anwar, N. and Kar, S., 2019. Review paper on various software testing techniques & strategies. *Global Journal of Computer Science and Technology*.

Aronson, S., 2022. *Understanding functional and equational programming techniques on graph-based problems* (Doctoral dissertation).

Bagheri, M., Akbari, A. and Mirbagheri, S.A., 2019. Advanced control of membrane fouling in filtration systems using artificial intelligence and machine learning techniques: A critical review. Process Safety and Environmental Protection, 123, pp.229-252.

Bahrami, R. and Soltani, S., 2020. The advanced educational standards in mathematics and engineering. *International Journal of Modern Engineering Technologies*, *2*(1).

Besançon, L., Ynnerman, A., Keefe, D.F., Yu, L. and Isenberg, T., 2021, February. The state of the art of spatial interfaces for 3d visualization. In *Computer Graphics Forum* (Vol. 40, No. 1, pp. 293-326).

Chakraborty, D. and Elzarka, H., 2019. Advanced machine learning techniques for building performance simulation: a comparative analysis. *Journal of Building Performance Simulation*, *12*(2), pp.193-207.

Chen, C., Haduong, P., Brennan, K., Sonnert, G. and Sadler, P., 2019. The effects of first programming language on college students’ computing attitude and achievement: a comparison of graphical and textual languages. Computer Science Education, 29(1), pp.23-48.

Cheung, C. and Szpak, P., 2021. Interpreting past human diets using stable isotope mixing models. *Journal of Archaeological Method and Theory*, *28*(4), pp.1106-1142.

Chis, A.E., Moldovan, A.N., Murphy, L., Pathak, P. and Muntean, C.H., 2018. Investigating flipped classroom and problem-based learning in a programming module for computing conversion course. *Journal of Educational Technology & Society*, *21*(4), pp.232-247.

Coble, M.D. and Bright, J.A., 2019. Probabilistic genotyping software: an overview. *Forensic Science International: Genetics*, *38*, pp.219-224.

Collier, J.E., 2020. Applied structural equation modeling using AMOS: Basic to advanced techniques. Routledge.

Cybulski, P. and Horbiński, T., 2020. User experience in using graphical user interfaces of web maps. *ISPRS International Journal of Geo-Information*, *9*(7), p.412.

David, R., Duke, J., Jain, A., Janapa Reddi, V., Jeffries, N., Li, J., Kreeger, N., Nappier, I., Natraj, M., Wang, T. and Warden, P., 2021. Tensorflow lite micro: Embedded machine learning for tinyml systems. *Proceedings of Machine Learning and Systems*, *3*, pp.800-811.

Dudley, J.J. and Kristensson, P.O., 2018. A review of user interface design for interactive machine learning. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, *8*(2), pp.1-37.

Elbadawi, M., Gaisford, S. and Basit, A.W., 2021. Advanced machine-learning techniques in drug discovery. *Drug Discovery Today*, *26*(3), pp.769-777.

Elbadawi, M., Gaisford, S. and Basit, A.W., 2021. Advanced machine-learning techniques in drug discovery. Drug Discovery Today, 26(3), pp.769-777.

Ganose, A.M., Jackson, A.J. and Scanlon, D.O., 2018. sumo: Command-line tools for plotting and analysis of periodic\* ab initio\* calculations. *Journal of Open Source Software*, *3*(28), p.717.

Han, J., Pei, J. and Tong, H., 2022. *Data mining: concepts and techniques*. Morgan kaufmann.

Han, J., Pei, J. and Tong, H., 2022. Data mining: concepts and techniques. Morgan kaufmann.

Hankeln, C., Adamek, C. and Greefrath, G., 2019. Assessing sub-competencies of mathematical modelling—Development of a new test instrument. In *Lines of inquiry in mathematical modelling research in education* (pp. 143-160). Springer, Cham.

Karlsen, A.C.B., Akerbæk, T.M. and Stigberg, S.K., 2022. Towards a Metaphor-Based Tangible Toolkit for Learning Programming Concepts. In International Conference on Human-Computer Interaction (pp. 72-88). Springer, Cham.

Kirsan, A.S., Arisa, N.N. and Insanittaqwa, V.F., 2022. Development of SIAKAD Applications in Balikpapan Schools using APXP: Advanced Personal Extreme Programming. *INOVTEK Polbeng-Seri Informatika*, *7*(1), pp.97-109.

Klabnik, S. and Nichols, C., 2019. *The Rust Programming Language (Covers Rust 2018)*. No Starch Press.

Kobayashi, K. and Suzuki, T.S., 2020. Development of impedance analysis software implementing a support function to find good initial guess using an interactive graphical user interface. *Electrochemistry*, *88*(1), pp.39-44.

Kraft, M.A., 2020. Interpreting effect sizes of education interventions. *Educational Researcher*, *49*(4), pp.241-253.

Kubica, V., 2019. Implementation and Improvement of Software Testing in Telco.

Lambropoulos, N. and Dimakos, I., 2022. STEM+ ARTS= STEAM Skills: Innovation Management and Scratch Programming for Year 4 Students. Innovating STEM Education: Increased Engagement and Best Practices, p.119.

Liang, Y.D., 2020. Introduction to Java programming and data structures, comprehensive version.

Lux, E., Adam, M.T., Dorner, V., Helming, S., Knierim, M.T. and Weinhardt, C., 2018. Live biofeedback as a user interface design element: A review of the literature. *Communications of the Association for Information Systems*, *43*(1), p.18.

Markova, O., Semerikov, S. and Popel, M., 2018. CoCalc as a Learning Tool for Neural Network Simulation in the Special Course" Foundations of Mathematic Informatics". *arXiv preprint arXiv:1807.00795*.

Markova, O., Semerikov, S., Striuk, A., Shalatska, H., Nechypurenko, P. and Tron, V., 2019. Implementation of cloud service models in training of future information technology specialists.

Mathew, R., Malik, S.I. and Tawafak, R.M., 2019. Teaching Problem Solving Skills using an Educational Game in a Computer Programming Course. *Informatics in education*, *18*(2), pp.359-373.

Mathew, R., Malik, S.I. and Tawafak, R.M., 2019. Teaching Problem Solving Skills using an Educational Game in a Computer Programming Course. Informatics in education, 18(2), pp.359-373.

Monnappa, K.A., 2018. Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware. Packt Publishing Ltd.

Mozelius, P. and Humble, N., 2022. Programming in K-12 Mathematics–A two-step rocket. In *INTED2022 Proceedings* (pp. 2389-2397). IATED.

Muniz, S.C.S., Peixoto, J.L.B. and Magina, S.M.P., 2020. A INCLUSÃO DOS SURDOS NAS AULAS DE MATEMÁTICA: ANÁLISE DAS RELAÇÕES PEDAGÓGICAS NA TRÍADE PROFESSORA-INTÉRPRETE-SURDO/THE INCLUSION OF DEAF PEOPLE ON MATH CLASSES: PEDAGOGICAL ANALYSES OF THE TRIAD RELATION TEACHER-INTERPRETER-DEAF STUDENT. *Revista Dynamis*, *26*(2), pp.23-39.

Neymotin, S.A., Daniels, D.S., Caldwell, B., McDougal, R.A., Carnevale, N.T., Jas, M., Moore, C.I., Hines, M.L., Hämäläinen, M. and Jones, S.R., 2020. Human Neocortical Neurosolver (HNN), a new software tool for interpreting the cellular and network origin of human MEG/EEG data. *Elife*, *9*, p.e51214.

Nikravesh, P.E., 2018. Planar Multibody Dynamics: Formulation, Programming with MATLAB®, and Applications. CRC press.

O’rinov Nodirbek Toxirjonovich, M.A. and Abduvaliyevich, S.S., 2020. Mathematical modeling: from the classroom to the real world. *Solid State Technology*, *63*(2s).

Olsina, L., Lew, P. and Tebes, G., 2022. Supplementary Results of a Comparative Syntactic and Semantic Study of Terms for Software Testing Glossaries. *arXiv preprint arXiv:2205.10668*.

Ozudogru, M. and Ozudogru, F., 2019. Technological pedagogical content knowledge of mathematics teachers and the effect of demographic variables. *Contemporary educational technology*, *10*(1), pp.1-24.

Papadakis, S., Kalogiannakis, M. and Zaranis, N., 2021. Teaching mathematics with mobile devices and the Realistic Mathematical Education (RME) approach in kindergarten. *Advances in Mobile Learning Educational Research*, *1*(1), pp.5-18.

Putri, D.P., Ferdianto, F. and Fauji, S.H., 2020. Designing a Digital Teaching Module Based on Mathematical Communication in Relation and Function. *Journal on Mathematics Education*, *11*(2), pp.223-236.

Ramsay-Jordan, N., Crenshaw, A. and Chestnutt, C., 2022. Examining purposeful researchable questions in mathematics education. *Journal of Honai Math*, *5*(2), pp.127-146.

Reynolds, L. and McDonell, K., 2021, May. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (pp. 1-7).

Sánchez, C., Schneider, G., Ahrendt, W., Bartocci, E., Bianculli, D., Colombo, C., Falcone, Y., Francalanza, A., Krstić, S., Lourenço, J.M. and Nickovic, D., 2019. A survey of challenges for runtime verification from advanced application domains (beyond software). *Formal Methods in System Design*, *54*(3), pp.279-335.

Sánchez, C., Schneider, G., Ahrendt, W., Bartocci, E., Bianculli, D., Colombo, C., Falcone, Y., Francalanza, A., Krstić, S., Lourenço, J.M. and Nickovic, D., 2019. A survey of challenges for runtime verification from advanced application domains (beyond software). Formal Methods in System Design, 54(3), pp.279-335.

Sarkar, D., Bali, R. and Ghosh, T., 2018. Hands-On Transfer Learning with Python: Implement advanced deep learning and neural network models using TensorFlow and Keras. Packt Publishing Ltd.

Semenikhina, O., Kudrina, O., Koriakin, O., Ponomarenko, L., Korinna, H. and Krasilov, A., 2020. The Formation of Skills to Visualize by the Tools of Computer Visualization. *TEM Journal*, *9*(4), p.1704.

Shanmugamani, R., 2018. Deep Learning for Computer Vision: Expert techniques to train advanced neural networks using TensorFlow and Keras. Packt Publishing Ltd.

Silvis-Cividjian, N., Went, M., Jansma, R., Bonev, V. and Apostolov, E., 2021, June. Good Bug Hunting: Inspiring and Motivating Software Testing Novices. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1* (pp. 171-177).

Tamur, M. and Juandi, D., 2020. Effectiveness of constructivism based learning models against students mathematical creative thinking abilities in Indonesia: a meta-analysis study. *Pervasive Health: Pervasive Computing Technologies for Healthcare*, *1*, pp.107-114.

Tebes, G., Lew, P. and Olsina, L., 2022, June. Syntactic and Semantic Similarities and Discrepancies between Terms of Glossaries for Software Testing. In *Anais do XXV Congresso Ibero-Americano em Engenharia de Software* (pp. 278-291). SBC.

Tirkolaee, E.B., Goli, A. and Weber, G.W., 2020. Fuzzy mathematical programming and self-adaptive artificial fish swarm algorithm for just-in-time energy-aware flow shop scheduling problem with outsourcing option. IEEE transactions on fuzzy systems, 28(11), pp.2772-2783.

Triana, M. and Zubainur, C.M., 2019. Students' Mathematical Communication Ability through the Brain-Based Learning Approach Using Autograph. *Journal of Research and Advances in Mathematics Education*, *4*(1), pp.1-10.

Tyralis, H., Papacharalampous, G. and Langousis, A., 2019. A brief review of random forests for water scientists and practitioners and their recent history in water resources. *Water*, *11*(5), p.910.

Tyralis, H., Papacharalampous, G. and Langousis, A., 2019. A brief review of random forests for water scientists and practitioners and their recent history in water resources. Water, 11(5), p.910.

Van Merriënboer, B., Breuleux, O., Bergeron, A. and Lamblin, P., 2018. Automatic differentiation in ML: Where we are and where we should be going. *Advances in neural information processing systems*, *31*.

Vijayasree, D., Roopa, N.S. and Arun, A., 2022. A Review on the Process of Automated Software Testing. *arXiv preprint arXiv:2209.03069*.

Wang, V., Xu, N., Liu, J.C., Tang, G. and Geng, W.T., 2021. VASPKIT: A user-friendly interface facilitating high-throughput computing and analysis using VASP code. *Computer Physics Communications*, *267*, p.108033.

Xi, W., Liu, W., Bai, T., Ye, W.P. and Shi, J., 2020. An automation test strategy based on real platform for digital control system software in nuclear power plant. *Energy Reports*, *6*, pp.580-587.

Yang, J., Yang, Y. and Li, M., 2022. OptControl. jl: An interpreter for optimal control problem. *arXiv preprint arXiv:2207.13229*.

Yorgancı, S., 2018. A study on the views of graduate students on the use of GeoGebra in mathematics teaching. *European Journal of Education Studies*.

Ziatdinov, R. and Valles Jr, J.R., 2022. Synthesis of modeling, Visualization, and programming in GeoGebra as an effective approach for teaching and learning STEM topics. *Mathematics*, *10*(3), p.398.

**Appendices**

(Gui.py)

import tkinter as tk

from tkinter.ttk import \*

class MainGUI(tk.Toplevel):

def \_\_init\_\_(self, parent):

super().\_\_init\_\_(parent)

# root = tk.Tk()

# b = MainGUI(root)

self.title("Mathemetic Equation!")

self.geometry("900x600")

self.resizable(0,0)

# gui variables

self.equation = tk.StringVar()

self.variables = {}

self.display\_var = 0

self.var\_name\_1 = tk.Label()

self.var\_val\_1 = tk.Label()

self.data = []

self.dis\_data = []

self.dis\_variable = {}

# implement UI

self.top\_frame = tk.Frame(self, width=200, height=700,

bd=10, relief='flat', bg='cadet blue')

self.top\_frame.place(x=0,y=0)

variables = tk.Label(self.top\_frame, text="Variables", font=('calibri', 13), bg='lightgray').place(x=90, y=13, width=200, height=50, anchor="center")

var\_name\_lbl = tk.Label(self.top\_frame, text="Name", font=('calibri', 13), bg='white').place(x=40, y=50, width=100, height=30, anchor="center")

var\_val\_lbl = tk.Label(self.top\_frame, text="Value", font=('calibri', 13), bg='white').place(x=140, y=50, width=100, height=30, anchor="center")

equation\_lbl = tk.Label(self, text="Equation : ", font=('calibri', 16)).place(x=257, y=570, anchor="center")

equation\_input = Entry(self, textvariable=self.equation, font = ('calibri', 16, ''), width=25)

equation\_input.place(x=310, y=554)

s = Style()

s.configure('my.TButton', font=('calibri', 12))

submit = Button(self, text="Submit",style='my.TButton', command=self.submit)

submit.place(x=600, y=553, height=35)

clear = Button(self, text="Clear",style='my.TButton', command=self.clear)

clear.place(x=700, y=553, height=35)

self.eq\_frame = tk.Frame(self, width=700, height=510,

bd=10, relief='flat')

self.eq\_frame.place(x=200,y=15)

def submit(self):

eq = self.equation.get()

if(eq):

self.equation.set('')

self.display\_var = 0

self.display\_varx = 10

# Variable setup

if(eq.\_\_contains\_\_("->")):

try:

var\_name, var\_value = eq.split('->')

if(var\_name.isalpha() and var\_value.isdigit()):

self.variables[var\_name.strip()] = var\_value

for i in self.data:

i.destroy()

self.var\_name\_x = 41

self.var\_name\_y = 82

self.var\_value\_x = 140

self.var\_value\_y = 82

count = 1

for var\_key, var\_val in self.variables.items():

if(count > 1):

self.var\_name\_y += 31

self.var\_value\_y += 31

var\_name\_1 = tk.Label(self.top\_frame, text=var\_key, borderwidth=3, relief="groove", font=('calibri', 13), bg='grey88')

var\_name\_1.place(x=self.var\_name\_x, y=self.var\_name\_y, width=98, height=30, anchor="center")

var\_val\_1 = tk.Label(self.top\_frame, text=var\_val, borderwidth=3, relief="groove", font=('calibri', 13), bg='grey88')

var\_val\_1.place(x=self.var\_value\_x, y=self.var\_value\_y, width=98, height=30, anchor="center")

self.data.append(var\_name\_1)

self.data.append(var\_val\_1)

count += 1

self.display\_variable(var\_name, var\_value, equation=None)

else:

print("Invalid variable declaration equation")

except Exception as error:

print(error)

print('Invalid Equation')

else:

self.display\_variable(var\_name=None, var\_value=None, equation=eq)

def display\_variable(self, var\_name, var\_value, equation):

new\_eq = equation

if(equation):

for eq\_key, eq\_val in self.variables.items():

new\_eq = new\_eq.replace(eq\_key, eq\_val)

if(new\_eq.\_\_contains\_\_('^')):

new\_eq = new\_eq.replace('^', '\*\*')

var\_value = eval(new\_eq)

self.dis\_variable[equation] = var\_value

else:

equation = f"{var\_name.strip()} -> {var\_value}"

self.dis\_variable[equation] = var\_value

for i in self.dis\_data:

i.destroy()

count = 0

for key\_name, key\_value in self.dis\_variable.items():

if(count == 10):

self.display\_varx += 300

self.display\_var = 0

eq\_1 = tk.Label(self.eq\_frame, text=f"> {key\_name}", font=('calibri', 14))

eq\_1.place(x=self.display\_varx,y=self.display\_var)

eq\_2 = tk.Label(self.eq\_frame, text=f"= {key\_value}", font=('calibri', 14))

eq\_2.place(x=self.display\_varx,y=self.display\_var+26)

self.dis\_data.append(eq\_1)

self.dis\_data.append(eq\_2)

self.display\_var += 54

count += 1

def clear(self):

self.dis\_variable = {}

for i in self.dis\_data:

i.destroy()

(Calculator.py)

import tkinter as tk

from math import \*

from tokens import Token

from lexer import Lexer

from \_parser\_ import Parser

from interpreter import Interpreter

# used to switch between units of rad, and deg

convert\_constant = 1

inverse\_convert\_constant = 1

btn\_params = {'padx': 16, 'pady': 1, 'bd': 4, 'fg': 'white', 'bg': 'black', 'font': ('arial', 18),

'width': 2, 'height': 2, 'relief': 'flat', 'activebackground': "black"}

class CalcGUI(tk.Toplevel):

def \_\_init\_\_(self, parent):

super().\_\_init\_\_(parent)

self.title("Scientific Calculator!")

self.geometry("435x490+50+50")

self.resizable(0, 0)

# expression that will be displayed on screen

self.expression = ""

# be used to store data in memory

self.recall = ""

# self.answer

self.sum\_up = ""

# create string for text input

self.text\_input = tk.StringVar()

# set frame showing inputs and title

top\_frame = tk.Frame(self, width=600, height=10,

bd=10, relief='flat', bg='gray')

top\_frame.pack(side=tk.TOP)

# set frame showing all buttons

bottom\_frame = tk.Frame(

self, width=600, height=470, bd=2, relief='flat', bg='black')

bottom\_frame.pack(side=tk.BOTTOM)

# Here is the code for Display of Calculator.

# entry interface for inputs

self.equation = tk.Entry(top\_frame, font=('arial', 36), relief='flat', bg='black', fg='white', textvariable=self.text\_input, width=15, bd=10, justify='right')

self.equation.pack()

# row 0

self.btn\_left\_brack = tk.Button(

bottom\_frame, \*\*btn\_params, text="(", border=1, command=lambda: self.clickButton('(', ''))

self.btn\_left\_brack.grid(row=0, column=0)

self.btn\_right\_brack = tk.Button(

bottom\_frame, \*\*btn\_params, text=")", command=lambda: self.clickButton(')', ''))

self.btn\_right\_brack.grid(row=0, column=1)

self.btn\_pi = tk.Button(

bottom\_frame, \*\*btn\_params, text="π", command=lambda: self.clickButton('pi', ''))

self.btn\_pi.grid(row=0, column=2)

self.btn\_sqrt = tk.Button(

bottom\_frame, \*\*btn\_params, text="sqrt", command=lambda: self.clickButton('sqrt(', ''))

self.btn\_sqrt.grid(row=0, column=3)

self.btn\_div = tk.Button(

bottom\_frame, \*\*btn\_params, text="/", command=lambda: self.clickButton('/', ''))

self.btn\_div.grid(row=0, column=4)

self.btn\_clear = tk.Button(

bottom\_frame, \*\*btn\_params, text="C", command=self.btn\_clear\_all)

self.btn\_clear.grid(row=0, column=5)

# row 1

self.btn\_sqr = tk.Button(bottom\_frame, \*\*btn\_params, text=u"x\u00B2", command=lambda: self.clickButton('\*\*2', ''))

self.btn\_sqr.grid(row=1, column=0)

self.btn\_log = tk.Button(bottom\_frame, \*\*btn\_params, text="log", command=lambda: self.clickButton('log(', ''))

self.btn\_log.grid(row=1, column=1)

self.btn\_7 = tk.Button(bottom\_frame, \*\*btn\_params, text="7", command=lambda: self.clickButton('7', ''))

self.btn\_7.configure(activebackground="black", bg='black')

self.btn\_7.grid(row=1, column=2)

self.btn\_8 = tk.Button(bottom\_frame, \*\*btn\_params, text="8", command=lambda: self.clickButton('8', ''))

self.btn\_8.configure(activebackground="black", bg='black')

self.btn\_8.grid(row=1, column=3)

self.btn\_9 = tk.Button(bottom\_frame, \*\*btn\_params, text="9", command=lambda: self.clickButton('9', ''))

self.btn\_9.configure(activebackground="black", bg='black')

self.btn\_9.grid(row=1, column=4)

self.btn\_mult = tk.Button(bottom\_frame, \*\*btn\_params, text="x", command=lambda: self.clickButton('\*', ''))

self.btn\_mult.grid(row=1, column=5)

# row 2

self.btn\_sin = tk.Button(

bottom\_frame, \*\*btn\_params, text="sin", command=lambda: self.clickButton('sin(', ''))

self.btn\_sin.grid(row=2, column=0)

self.btn\_cos = tk.Button(

bottom\_frame, \*\*btn\_params, text="cos", command=lambda: self.clickButton('cos(', ''))

self.btn\_cos.grid(row=2, column=1)

self.btn\_4 = tk.Button(bottom\_frame, \*\*btn\_params, text="4", command=lambda: self.clickButton('4', ''))

self.btn\_4.configure(activebackground="black", bg='black')

self.btn\_4.grid(row=2, column=2)

self.btn\_5 = tk.Button(bottom\_frame, \*\*btn\_params, text="5", command=lambda: self.clickButton('5', ''))

self.btn\_5.configure(activebackground="black", bg='black')

self.btn\_5.grid(row=2, column=3)

self.btn\_6 = tk.Button(bottom\_frame, \*\*btn\_params, text="6", command=lambda: self.clickButton('6', ''))

self.btn\_6.configure(activebackground="black", bg='black')

self.btn\_6.grid(row=2, column=4)

self.btnSub = tk.Button(bottom\_frame, \*\*btn\_params, text="-", command=lambda: self.clickButton('-', ''))

self.btnSub.grid(row=2, column=5)

# row 3

self.btn\_sin\_inverse = tk.Button(bottom\_frame, \*\*btn\_params, text=u"sin-\u00B9", command=lambda: self.clickButton('arcsin(', ''))

self.btn\_sin\_inverse.grid(row=3, column=0)

self.btn\_cos\_inverse = tk.Button(bottom\_frame, \*\*btn\_params, text=u"cos-\u00B9", command=lambda: self.clickButton('arccos(', ''))

self.btn\_cos\_inverse.grid(row=3, column=1)

self.btn\_1 = tk.Button(bottom\_frame, \*\*btn\_params, text="1", command=lambda: self.clickButton('1', ''))

self.btn\_1.configure(activebackground="black", bg='black')

self.btn\_1.grid(row=3, column=2)

self.btn\_2 = tk.Button(bottom\_frame, \*\*btn\_params, text="2", command=lambda: self.clickButton("2", ''))

self.btn\_2.configure(activebackground="black", bg='black')

self.btn\_2.grid(row=3, column=3)

self.btn\_3 = tk.Button(bottom\_frame, \*\*btn\_params, text="3", command=lambda: self.clickButton('3', ''))

self.btn\_3.configure(activebackground="black", bg='black')

self.btn\_3.grid(row=3, column=4)

self.btn\_add = tk.Button(

bottom\_frame, \*\*btn\_params, text="+", command=lambda: self.clickButton('+', ''))

self.btn\_add.grid(row=3, column=5)

# row 4

self.btn\_tan = tk.Button(bottom\_frame, \*\*btn\_params, text="tan", command=lambda: self.clickButton('tan(', ''))

self.btn\_tan.grid(row=4, column=0)

self.btn\_tan\_inverse = tk.Button(bottom\_frame, \*\*btn\_params, text=u"tan-\u00B9", command=lambda: self.clickButton('arctan(', ''))

self.btn\_tan\_inverse.grid(row=4, column=1)

self.btn\_0 = tk.Button(bottom\_frame, \*\*btn\_params, text="0", command=lambda: self.clickButton('0', ''))

self.btn\_0.configure(activebackground="black", bg='black', width=7, bd=5)

self.btn\_0.grid(row=4, column=2, columnspan=2)

self.btn\_eq = tk.Button(bottom\_frame, \*\*btn\_params, text="=", command=lambda: self.clickButton('VALUE', ''))

self.btn\_eq.configure(bg='Gray', activebackground='#009999')

self.btn\_eq.grid(row=4, column=4)

self.btn\_dec = tk.Button(bottom\_frame, \*\*btn\_params, text=".", command=lambda: self.clickButton('.', ''))

self.btn\_dec.grid(row=4, column=5)

# functions

# allows button you click to be put into self.expression

def clickButton(self, value, text):

current\_equation = str(self.equation.get())

text = current\_equation

if value == 'CLEAR':

text = ""

self.equation.delete(-1, tk.END)

elif value == 'VALUE':

text = self.mathematical\_formula(text)

current\_equation = text

lexer = Lexer(current\_equation)

tokens = lexer.generate\_tokens()

parser = Parser(tokens)

tree = parser.parse()

if not tree:

self.equation.insert(0, "Error")

interpreter = Interpreter()

answer = interpreter.visit(tree)

self.equation.delete(-1, tk.END)

self.equation.insert(0, answer)

else:

self.equation.delete(0, tk.END)

self.equation.insert(-1, current\_equation + value)

def mathematical\_formula(self, text):

# short name use's for math trigonometry functions

if(text.\_\_contains\_\_('arctan(')):

text = text.replace('arctan', 'A')

if(text.\_\_contains\_\_('arcsin(')):

text = text.replace('arcsin', 'I')

if(text.\_\_contains\_\_('arccos(')):

text = text.replace('arccos', 'O')

if(text.\_\_contains\_\_('tan(')):

text = text.replace('tan', 'T')

if(text.\_\_contains\_\_('sin(')):

text = text.replace('sin', 'S')

if(text.\_\_contains\_\_('cos(')):

text = text.replace('cos', 'C')

if(text.\_\_contains\_\_('\*\*2')):

text = text.replace('\*\*2', '')

text = f'E{text}'

if(text.\_\_contains\_\_('pi')):

text = text.replace('pi', '')

text = f'P{text}'

if(text.\_\_contains\_\_('sqrt(')):

text = text.replace('sqrt', '')

text = f'Q{text}'

if(text.\_\_contains\_\_('log(')):

text = text.replace('log', '')

text = f'L{text}'

return text

# clears self.expression

def btn\_clear\_all(self):

self.expression = ""

self.text\_input.set("")

(lexer.py)

from tokens import Token, TokenType

WHITESPACE = ' \n\t'

DIGITS = '0123456789'

class Lexer:

def \_\_init\_\_(self, text):

self.text = iter(text)

self.advance()

def advance(self):

try:

self.current\_char = next(self.text)

except StopIteration:

self.current\_char = None

def generate\_tokens(self):

while self.current\_char != None:

if self.current\_char in WHITESPACE:

self.advance()

elif self.current\_char == '.' or self.current\_char in DIGITS:

yield self.generate\_number()

elif self.current\_char == '+':

self.advance()

yield Token(TokenType.PLUS)

elif self.current\_char == '-':

self.advance()

yield Token(TokenType.MINUS)

elif self.current\_char == '\*':

self.advance()

yield Token(TokenType.MULTIPLY)

elif self.current\_char == '/':

self.advance()

yield Token(TokenType.DIVIDE)

elif self.current\_char == '%':

self.advance()

yield Token(TokenType.MODULO)

elif self.current\_char == '^':

self.advance()

yield Token(TokenType.EXPONENT)

elif self.current\_char == '(':

self.advance()

yield Token(TokenType.L\_PARENTHESIS)

elif self.current\_char == ')':

self.advance()

yield Token(TokenType.R\_PARENTHESIS)

elif self.current\_char == 'T':

self.advance()

yield Token(TokenType.TAN)

elif self.current\_char == 'A':

self.advance()

yield Token(TokenType.TAN\_INVERSE)

elif self.current\_char == 'S':

self.advance()

yield Token(TokenType.SIN)

elif self.current\_char == 'I':

self.advance()

yield Token(TokenType.SIN\_INVERSE)

elif self.current\_char == 'C':

self.advance()

yield Token(TokenType.COS)

elif self.current\_char == 'O':

self.advance()

yield Token(TokenType.COS\_INVERSE)

elif self.current\_char == 'E':

self.advance()

yield Token(TokenType.SQUARE)

elif self.current\_char == 'P':

self.advance()

yield Token(TokenType.PI)

elif self.current\_char == 'Q':

self.advance()

yield Token(TokenType.SQRT)

elif self.current\_char == 'L':

self.advance()

yield Token(TokenType.LOG)

else:

return "Invalid Input"

def generate\_number(self):

decimal\_pts\_count = 0

number\_string = self.current\_char

self.advance() # .

while self.current\_char != None and (self.current\_char == '.' or self.current\_char in DIGITS):

if self.current\_char == '.':

decimal\_pts\_count += 1

if decimal\_pts\_count > 1:

break

number\_string += self.current\_char

self.advance()

if number\_string.startswith('.'):

number\_string = '0' + number\_string

if number\_string.endswith('.'):

number\_string += '0'

return Token(TokenType.NUMBER, float(number\_string))

(interpreter.py)

from nodes import \*

from values import Number

from math import \*

class Interpreter:

def visit(self, node):

method\_name = f'visit\_{type(node).\_\_name\_\_}'

method = getattr(self, method\_name)

return method(node)

def visit\_NumberNode(self, node):

try:

return Number(node.value)

except:

return "Error in Calculation"

def visit\_AddNode(self, node):

try:

return Number(self.visit(node.node\_a).value + self.visit(node.node\_b).value)

except:

return "Error in Calculation"

def visit\_SubtractNode(self, node):

try:

return Number(self.visit(node.node\_a).value - self.visit(node.node\_b).value)

except:

return "Error in Calculation"

def visit\_MultiplyNode(self, node):

try:

return Number(self.visit(node.node\_a).value \* self.visit(node.node\_b).value)

except:

return "Error in Calculation"

def visit\_DivideNode(self, node):

try:

return Number(self.visit(node.node\_a).value / self.visit(node.node\_b).value)

except:

return "Error in Calculation"

def visit\_ModuloNode(self, node):

try:

return Number(self.visit(node.node\_a).value % self.visit(node.node\_b).value)

except:

return "Error in Calculation"

def visit\_ExponentNode(self, node):

try:

return Number(self.visit(node.node\_a).value \*\* self.visit(node.node\_b).value)

except:

return "Error in Calculation"

def visit\_PlusNode(self, node):

try:

return self.visit(node.node\_t).value

except:

return "Error in Calculation"

def visit\_MinusNode(self, node):

try:

return Number(-self.visit(node.node\_t).value)

except:

return "Error in Calculation"

def visit\_TanNode(self, node):

try:

return tan(self.visit(node.node\_t).value)

except:

return "Error in Calculation"

def visit\_TanInverseNode(self, node):

try:

return atan(self.visit(node.node\_t).value)

except:

return "Error in Calculation"

def visit\_SinNode(self, node):

try:

return sin(self.visit(node.node\_t).value)

except:

return "Error in Calculation"

def visit\_SinInverseNode(self, node):

try:

return asin(self.visit(node.node\_t).value)

except:

return "Error in Calculation"

def visit\_CosNode(self, node):

try:

return cos(self.visit(node.node\_t).value)

except:

return "Error in Calculation"

def visit\_CosInverseNode(self, node):

try:

return acos(self.visit(node.node\_t).value)

except:

return "Error in Calculation"

def visit\_SquareNode(self, node):

try:

return Number(self.visit(node.node\_t).value \*\* 2)

except:

return "Error in Calculation"

def visit\_PiNode(self, node):

try:

try:

return eval(str(self.visit(node.node\_t).value \* 3.1415926535))

except:

return 3.1415926535

except:

return "Error in Calculation"

def visit\_SqrtNode(self, node):

try:

return eval(str(sqrt(self.visit(node.node\_t).value)))

except:

return "Error in Calculation"

def visit\_LogNode(self, node):

try:

return eval(str(log(self.visit(node.node\_t).value)))

except:

return "Error in Calculation"