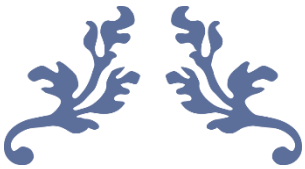
UNIVERSITÀ DEGLI STUDI
DI SALERNO



PROGRAMMABLE CALCULATOR

System Design Document

12th DECEMBER 2023

DE LUCA A. - D'AGOSTINO M. - DE LUCA D. - FESTA R.

- GROUP 21 -

Index

1 About

1.1 Team Information

Name	ID
D'Agostino Marco	*0612705817*
De Luca Aniello *(group leader)*	*0612705805*
De Luca Daniele	*0612705654*
Festa Raffaele	*0612705535*

1.2 Version

Revision	Date	Summary of changes
1.0	07/12/2023	First release
1.1	12/12/2023	Changes about methods visibility and Class Diagram replacements. Tracebility Matrix updated

2 Introduction

2.1 Purpose

The purpose of this project is to make a calculator app that uses a Stack to work with both complex and real numbers. By manipulating the Stack, the app enables users to carry out a range of mathematical operations, providing a flexible tool for numerical computation. Moreover, the app's interface shows the top elements of the Stack to simplify the input of complex expressions. Variable storage allows users to store and retrieve values when making calculations.

2.2 Document Scope

The purpose of this document is to present the system's general structure, showing classes with their attributes and modules that will be written in the *Implementation Phase* of the code, to facilitate reading and understanding the code.

3 Design Overview

3.1 Architectural Strategies

The architectural pattern used by the team to develop the calculator is MVC (Model View Controller). The code, in fact, will be divided into three main components: Model, View, and Controller.
The ***Model*** will be implemented in the "type" and "operation" packages, with classes defining the data types and the operations that will be performed on them.
The ***Visualization*** will be handled by JavaFX through the classes contained in the "gui" package, GuiApplication and GuiController, which deal with user interface and user interaction, respectively.
The ***Control*** will be handled in the GuiController class through the exceptions contained in the "exception" package, handling user input, errors that may occur and updating the visualization accordingly.

3.2 System Architeture

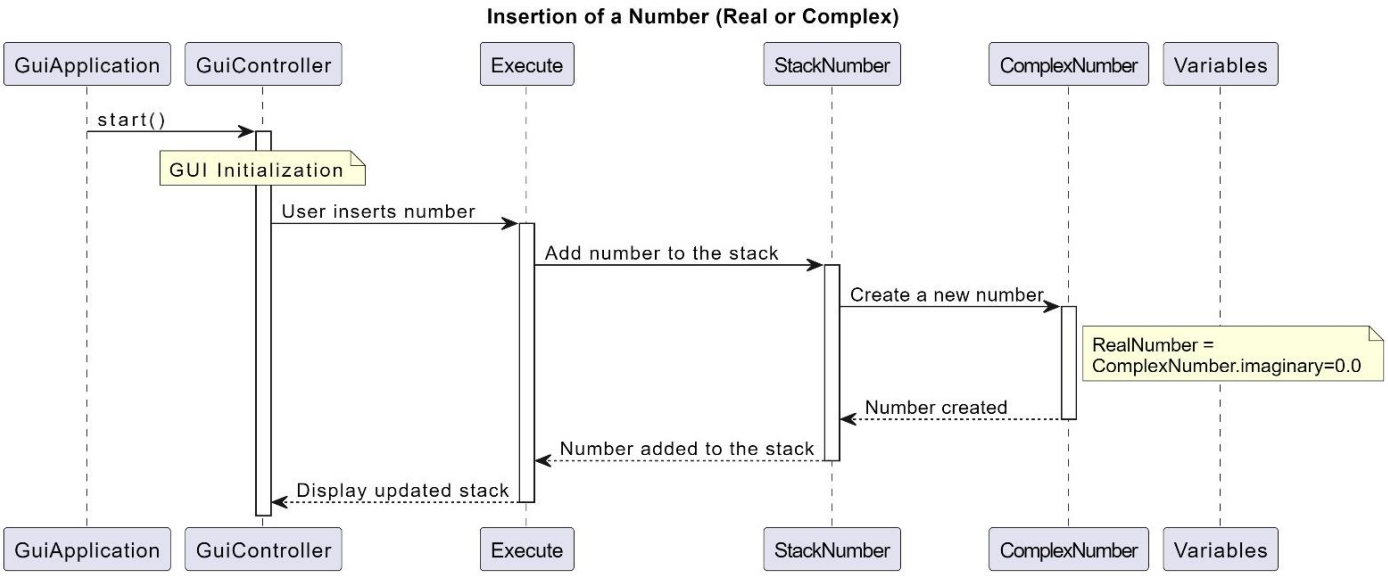
The system architecture is organized into different levels, each with a specific task.

- **GUI level:**
  1. **GuiApplication:** Launches the JavaFx application.
  2. **GuiController:** Handles the user interaction and the GUI refreshing.
  3. **FXML:** defines the structure and appearance of the GUI.

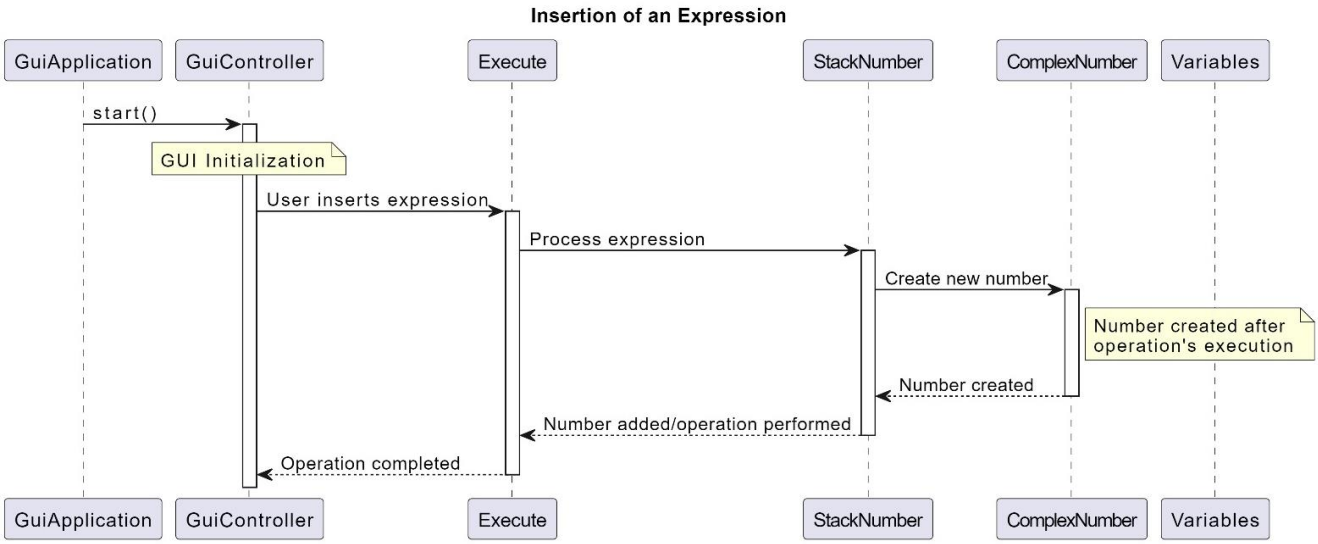
- **Operative level:**
  1. **Execute:** coordinates the execution of operations, analyzes inputs and uses the class "*Variables*" for variables handling.
  2. **Operation:** Contains specific logics for the arithmetic operations.
  3. **Variables:** Manages variables and their associated operations.
- **Types level:**
  1. **ComplexNumber:** Manages operations on complex numbers.
  2. **StackNumber:** Manages stack manipulation with operations as *pushNumber*, *popNumber*, *clearNumber* ecc.
- **Exceptions level:**
  1. **DivisionByZeroException:** thrown when the user tries to perform a division by zero.
  2. **InsufficientOperandsException:** thrown when the user tries to perform an operation with an insufficient number of operands.
  3. **InvalidExpressionException**: thrown when an invalid expression is detected.
  4. **NoValueInVariableException:** thrown when the user tries to access a variable without an associated value.
  5. **StackIsEmptyException:** lanciata quando si tenta di eseguire un'operazione su uno stack vuoto.

4 System Operation

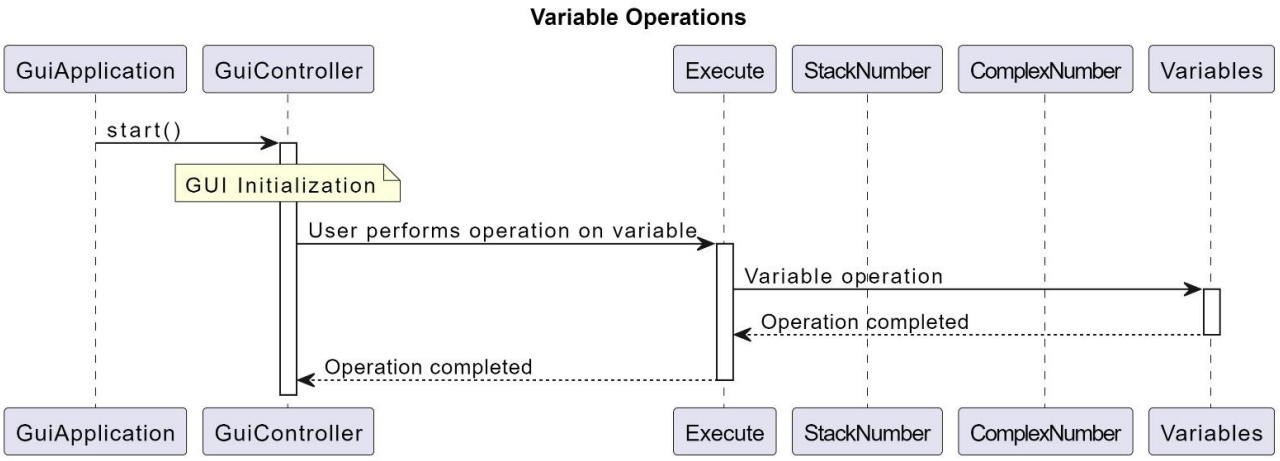
4.1 Insertion of a Number (Complex Number)



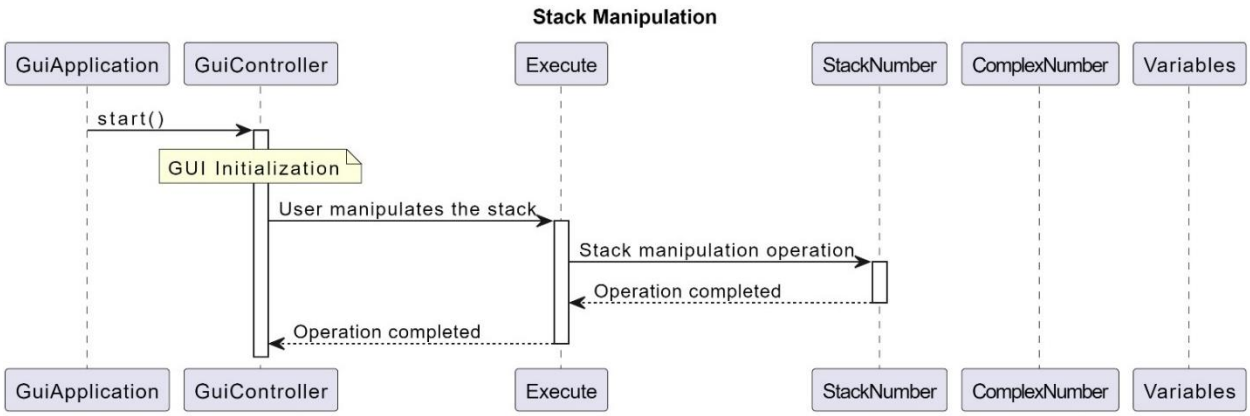
4.2 Insertion of an Expression



4.3 Variable Operation

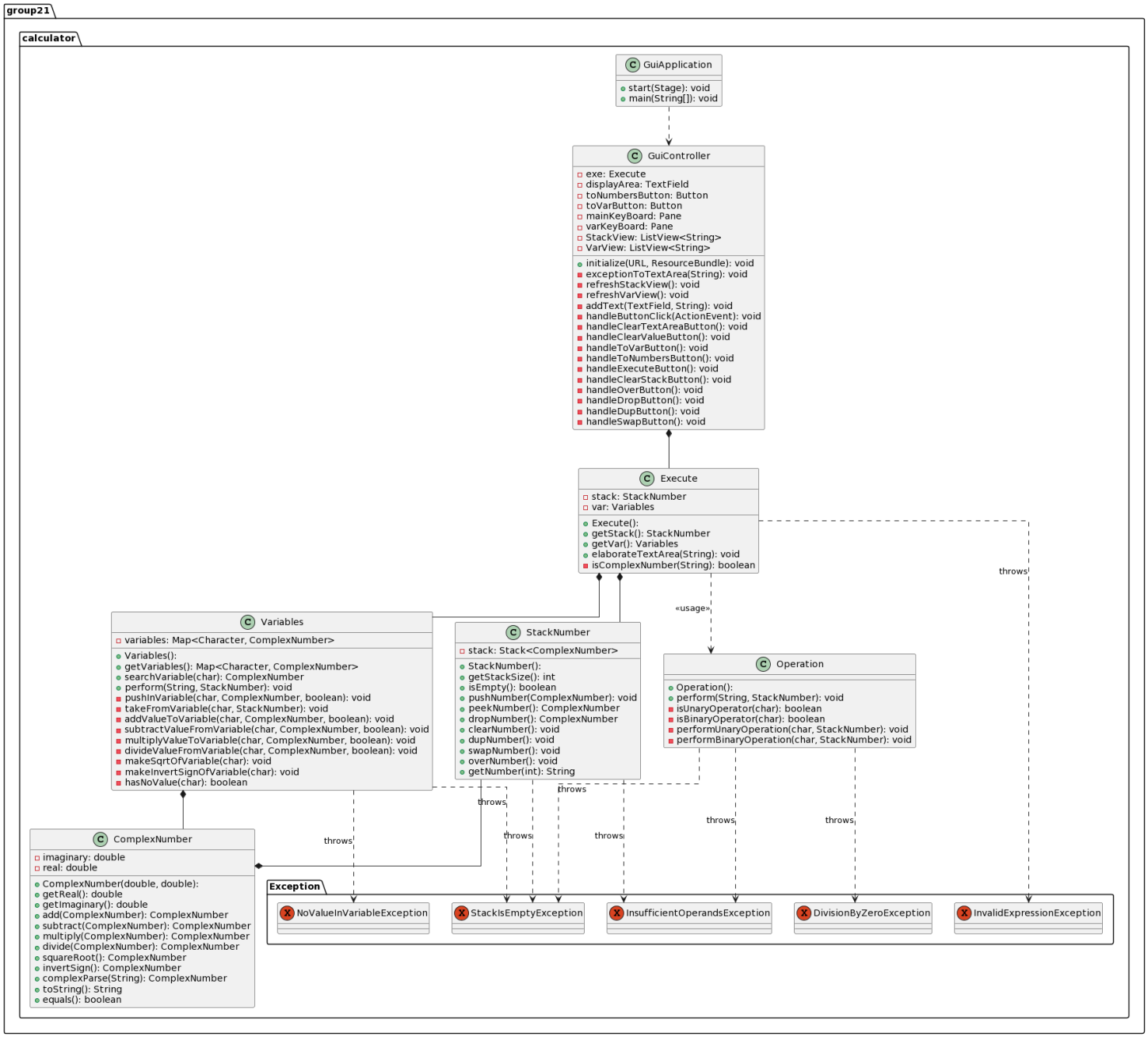


4.4 Manipulation of Stack

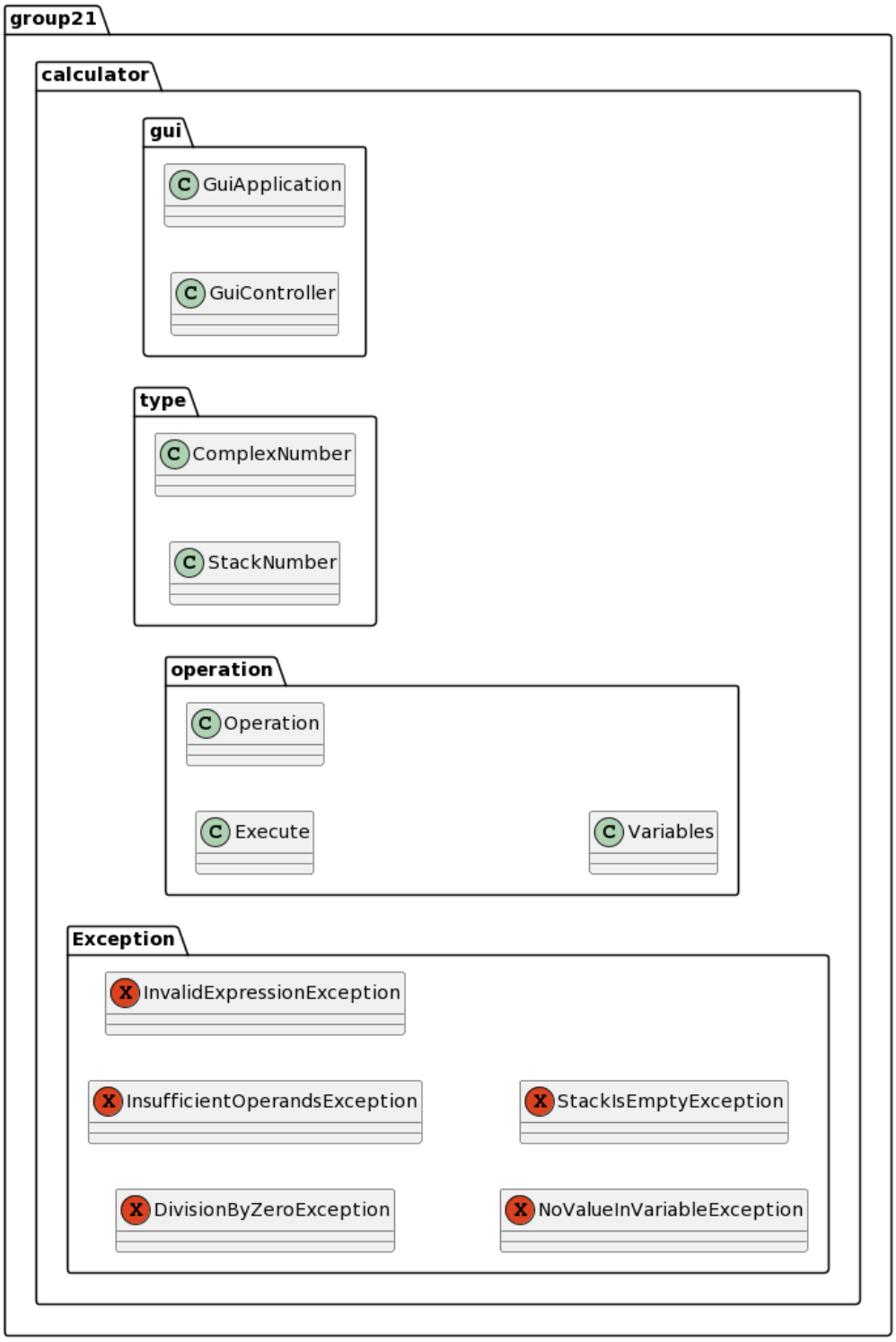


5 Calculator Interface

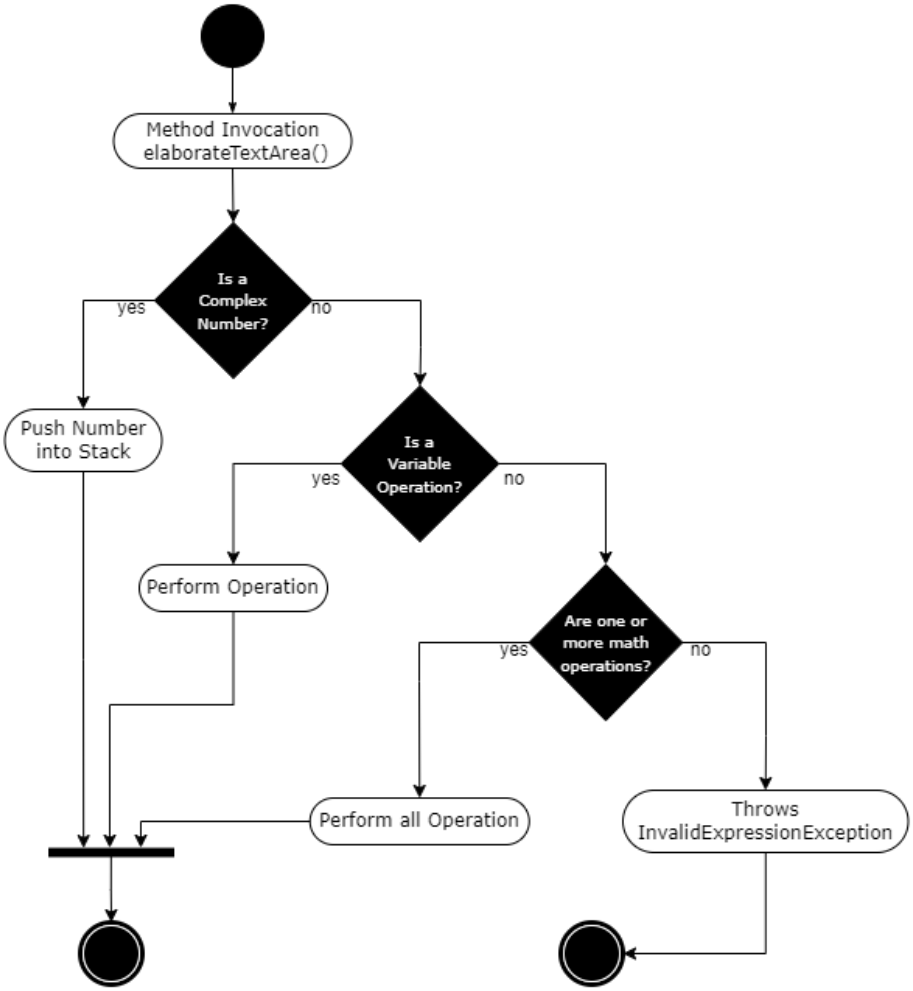
5.1 Class Diagram



5.2 Class Package Diagram



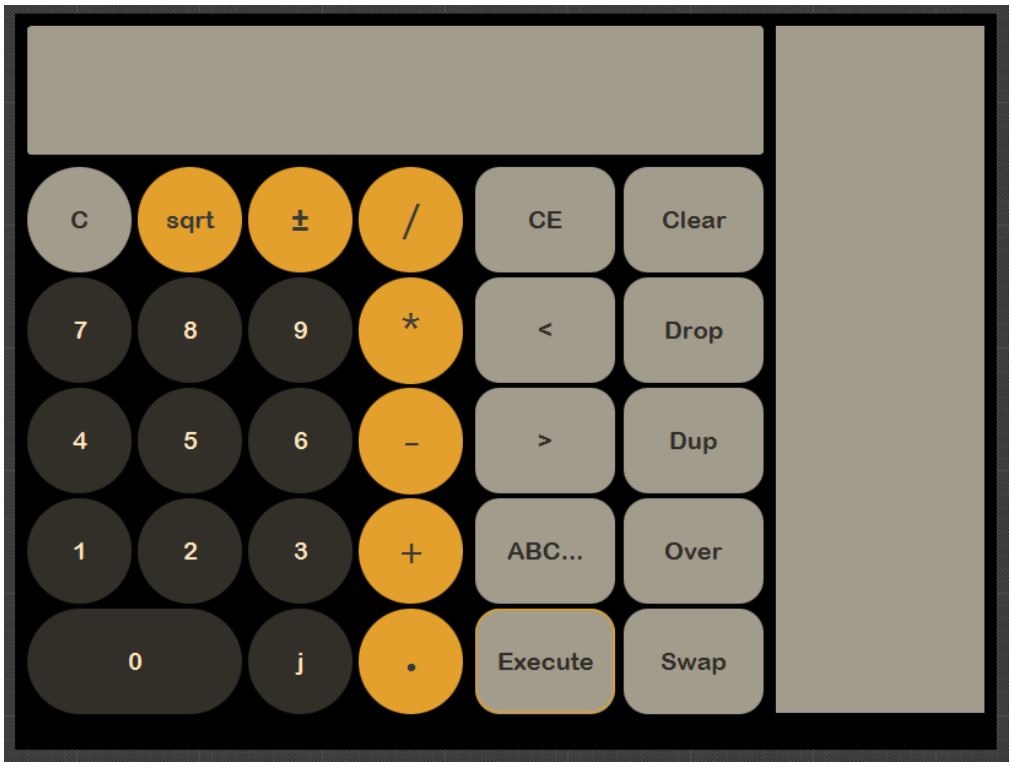
5.3 Activity Diagram



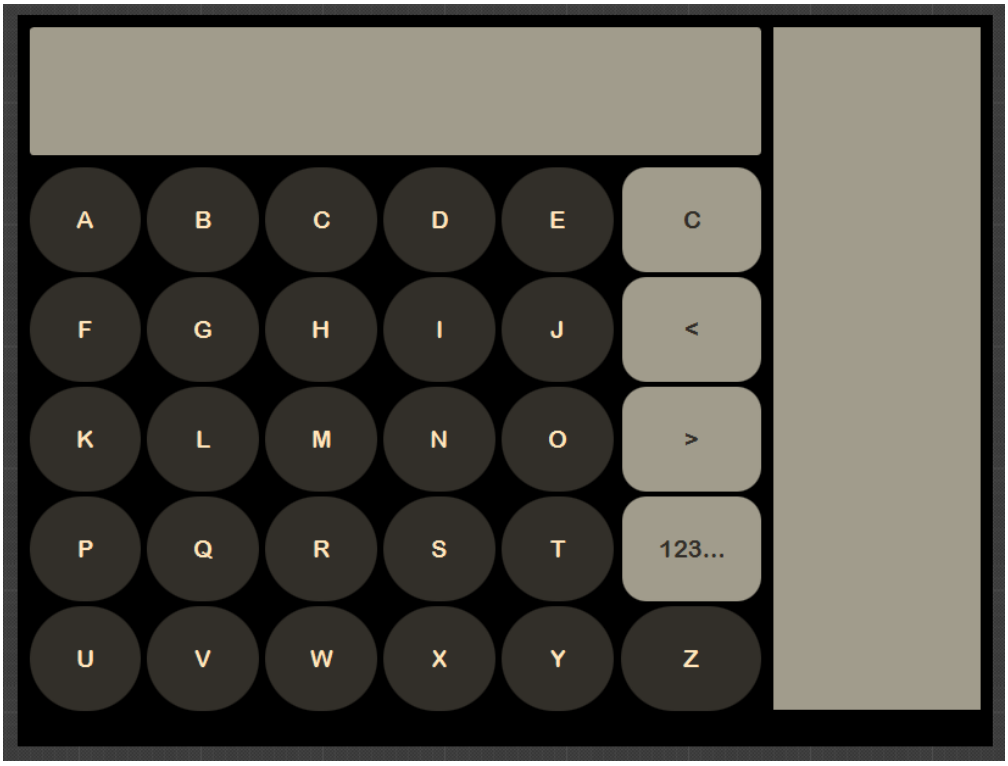
This diagram represents the text area processing algorithm. If the input in text area is invalid an exception will be thrown.

6 Detailed Interface Mock-Up

6.1 Main Interface



6.2 Variables Interface



Visualized when user click on “ABC…”

7 Tracebility Matrix

Tracebility Matrix – Group 21 – v1.2				
Req-ID	**Design**	**Code**	**Test**	**Related Requirements**
UI-1.1	**x**			N/A
UI-1.2	**x**			N/A
IS-1.1	**x**			N/A
IF-1.1	**x**	**x**		N/A
DF-1.1	**x**	**x**		**IF-1.1**
DF-1.2	**x**	**x**		**IF-1.1**
IF-1.2	**x**	**x**		N/A
DF-1.3	**x**	**x**		**IF-1.2**
IF-1.3	**x**	**x**		N/A
DF-1.4	**x**	**x**		**IF-1.3**
DF-1.5	**x**	**x**		**IF-1.3**
DF-1.6	**x**	**x**		**IF-1.3**
UI-2.1	**x**			N/A
IF-2.1	**x**	**x**		N/A
UI-2.2	**x**			**IF-2.1**
IS-2.1	**x**	**x**		**IF-2.1**
IS-2.2	**x**	**x**		**IF-2.1**
IS-2.3	**x**	**x**		**IF-2.1**
UI-3.1	**x**			N/A
IS-3.1	**x**			N/A
IF-3.1	**x**	**x**		N/A
IF-3.2	**x**	**x**		N/A
IF-3.3	**x**	**x**		N/A
IF-3.4	**x**	**x**		N/A
IF-3.5	**x**	**x**		N/A