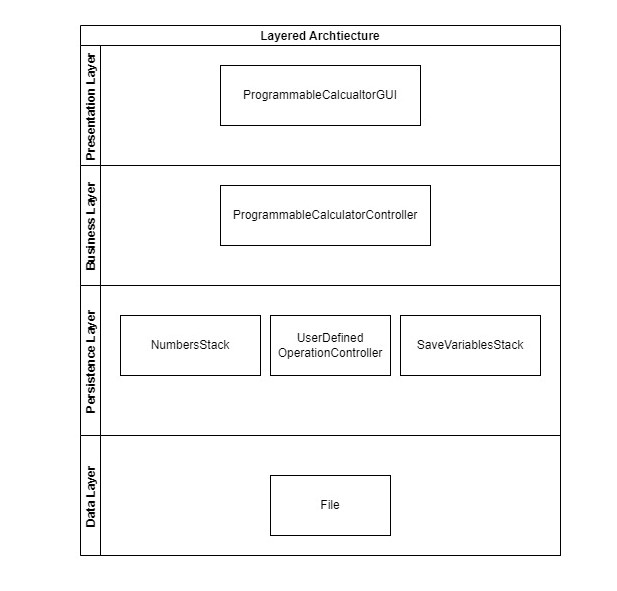
**Architecture**

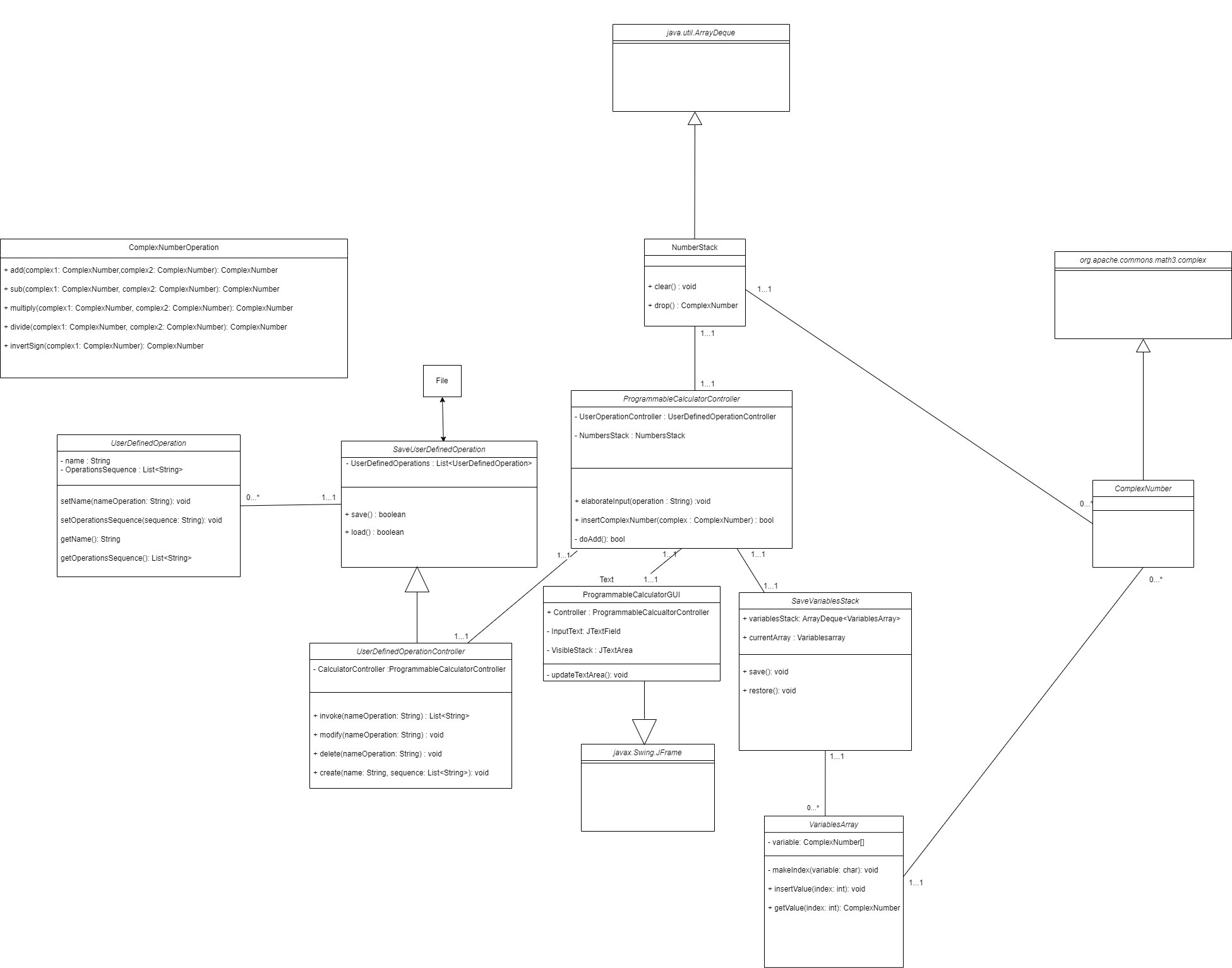
**Layered Style**

****

We used a layered approach for our program, grouping the components into layers. The components communicate with the other components in the layers immediately above and below their own layer.The GUI is the only part of the system that communicates with the user. Once the user inserts the input, it is passed on by the GUI to the controller, which is gonna implement the business logic of the program by understanding which operation to perform and which other components are influenced by such operation.

**Classes**:

* ComplexNumber: extends the Complex class of the Apache library, adding utility methods to our project
* ComplexNumberOperation: contains static methods for the execution of the operations requested by the software requirements
* NumberStack: Manages the stack of the complex numbers inserted from the user and implements the methods for the stack manipulation operations requested by the requirements
* ProgrammableCalculatorGUI: Implements the graphic interface of our calculator, in particular adding a text field for the inputs insert and a text area for the visualization of the stack
* ProgrammableCalculatorController: Implements the logic of the application, it chooses which operation to execute based on the input of the user
* VariablesArray: Represents the array of the 26 variables containing their current values
* SaveVariablesStack: Represents the stack containing all the values of the VariablesArray at the moments of the invocation of command “save” by the user
* UserDefinedOperation: Represents the operation defined by the user, with an unique name and an unique sequence of operations
* UserDefinedOperationController: Manages the user defined operations inserted by the user.
* SaveUserDefinedOperations: Manages saving on a file of the user-defined operations and also manages loading of these operations when starting the application.

**Class Diagram**