**Date** 2014-05-02

*Author: Gal Yehezkel and Gal Tfilin*

*Assignment 2 - Object Oriented Programming course*

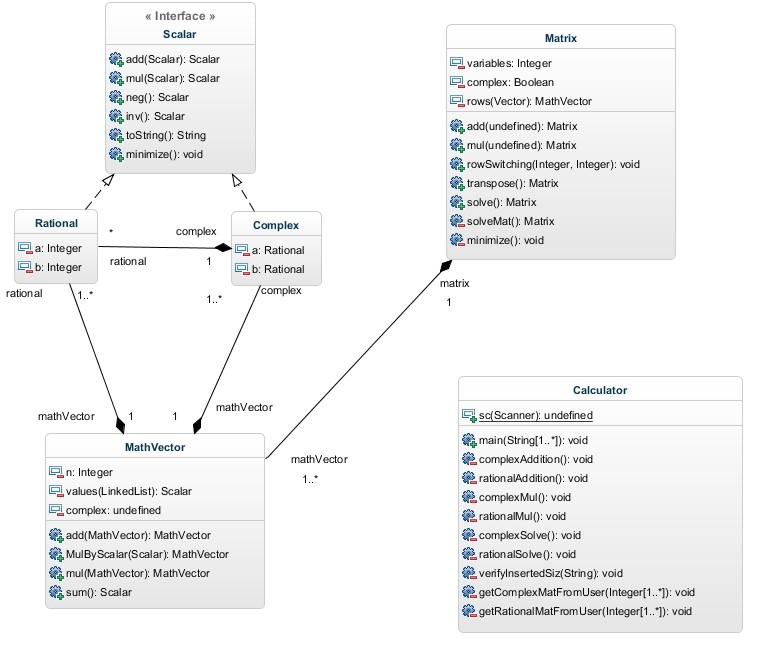
*Date: 2014-05-02*

HIDDEN TEXT TO MARK THE BEGINNING OF THE TABLE

## 1. Overview

**1.1 Model Description**

### 1.2 Diagrams



## 2 Classifiers

### 2.1 Class Rational

#### 2.1.1 Attributes

* **a** : Integer [1] - numerator
* **b** : Integer [1] – denominator

#### 2.1.4 Associations

* **complex** : Complex [1]
* **mathVector** : MathVector [1]

### 2.2 Class Complex

#### 2.2.1 Attributes

* **a** : Rational [1] - rational value
* **b** : Rational [1] – complex value

#### 2.2.4 Associations

* **rational** : Rational [\*]
* **mathVector** : MathVector [1]

### 2.3 Class MathVector

#### 2.3.1 Attributes

* **n** : Integer [1] – the vector’s amount of values
* **values(LinkedList)** : Scalar [1] – a linked list of the vector’s values
* **Complex** : Boolean [1] – indicates if the vector is within the rational field or the complex field

#### 2.3.3 Operations

* MathVector **add** (parameter : MathVector) – an addition between this vector and another vector
* MathVector **MulByScalar** (parameter : Scalar) – a multiplication between this vector and a scalar
* MathVector **mul** (parameter : MathVector) – a multiplication between this vector and another vector
* Scalar **sum** () – returns the sum of all of this vector’s values

#### 2.3.4 Associations

* **complex** : Complex [1..\*]
* **matrix** : Matrix [1]
* **rational** : Rational [1..\*]

### 2.4 Class Matrix

#### 2.4.1 Attributes

* **variables** : Integer [1] – the amount of variables in all of this matrix’s vectors
* **complex** : Boolean [1] – indicates of the matrix is within the rational field or the complex
* **rows(Vector)** : MathVector [1] – a Vector(data structure) that contains the matrix’s vectors

#### 2.4.3 Operations

* Matrix **add** (parameter : Matrix) – adds a matrix with this matrix
* Matrix **mul** (parameter : Matrix) – multiplies a matrix with this ma
* Void **rowSwitching** (parameter : Integer, parameter2 : Integer) – switches between two rows in the matrix
* Matrix **transpose** () – commits a transpose on this matrix
* Matrix **solve** () – solves this matrix
* Matrix **solveMat** () – the private implementation of solving the matrix
* Void **minimize** () – minimizes the matrix’s values into smaller fractions

**2.4.4 Associations**

* **mathVector** : MathVector [1..\*]

### 2.5 Class Calculator

**2.5.1 Attributes**

- **sc**: Scanner[1]

#### 2.5.3 Operations

* Void **main** (parameter : String) – the main method that manages the application’s functionality
* Void **complexAddition** () – in charge of adding two complex matrixes
* Void **rationalAddition** () – in charge of adding two rational matrixes
* Void **complexMul** () – in charge of multiplying two complex matrixes
* Void **rationalMul** () – in charge of multiplying two rational matrixes
* Void **complexSolve** () – in charge of solving a complex matrix
* Void **rationalSolve** () – in charge of solving a rational matrix
* Void **verifyInsertedSize** (parameter : String) – verifies the matrix size that was inserted by the user
* Void **getComplexMatFromUser** (parameter : Integer) – gets a complex matrix from the user
* Void **getRationalMatFromUser** (parameter : Integer) – gets a rational matrix from the user

## 3 Interfaces

### 3.6 Interface Scalar

#### 3.6.1 Operations

* Scalar **add** (parameter : Scalar) – adds this scalar with another scalar
* Scalar **mul** (parameter : Scalar) – multiplies this scalar with another scalar
* Scalar **neg** () – returns the negative value of this scalar
* Scalar **inv** () – returns the inverted value of this scalar
* String **toString** () – returns this scalar a string
* Void **minimize** () – minimizes this scalar into a smaller fraction if possible