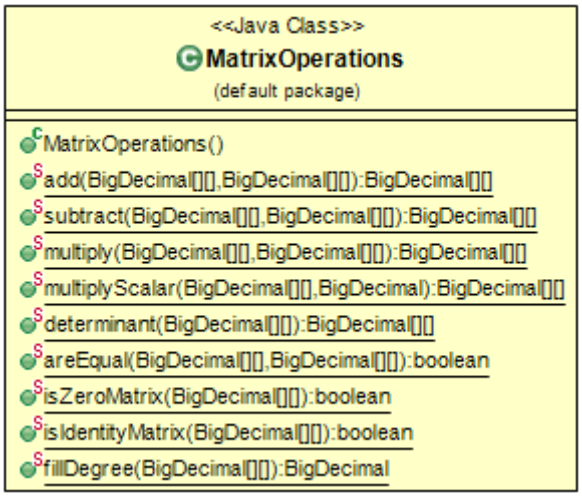
Create a *MatrixOperations* class with the following structure:




*Notes:*

- *Read up on UML diagrams*
- *General form of method declarations in UML:*
  **accessModifier name(paramType1, paramType2, … , paramTypeN) : returnType**
  ex: *S(tatic) multiply(BigDecimal[][], BigDecimal[][]) : BigDecimal[][]*

- *All methods are static!!*
- *BigDecimal is a java class from the **Math** package. Read up on it and play around with its methods*
- *The constructor can be absent in the first part of the implementation.*

Implement all the operations described below:

- ***add:*** adds two matrices together
- ***subtract:*** subtracts the second matrix from the first
- ***multiply:*** multiples the two matrices
- ***multiplyScalar:*** multiplies a matrix with a scalar
- ***determinant:*** calculates the determinant of a matrix
- ***areEqual:*** checks if two matrices are identical
- ***isZeroMatrix:*** checks if a matrix contains only 0s
- ***isIdentityMatrix:*** checks if a matrix is an identity matrix
- ***fillDegree:*** computes the % of the matrix which is filled (elements != **0**)

Twist:
Read up on the *singleton pattern* (from whichever sources you find). Understand it and apply it to this particular class. Can we now make the methods public? If yes, do so! If no, why not? What are the main advantages of using the *singleton* pattern?

Twist2: (hard)
A system of equations can be represented as a matrix. Find the solutions to an **NxN** system of equations by representing it as a matrix.