Spring 2016

Object-Oriented Programming

Programming Assignment #2 - Matrix

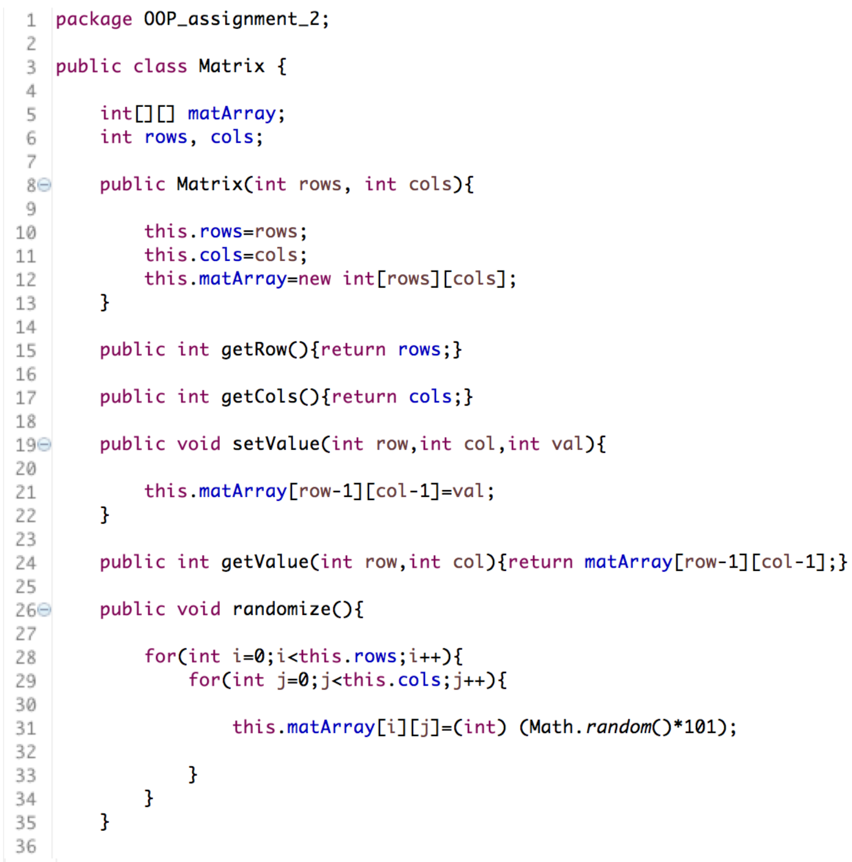
Name : Kim Da Bin
Student Id : 2015004375
Class NO : Class_1
Major : Computer Science Engineering

Prof. : Lee Choon Hwa
Assistant Teacher : Kena Alexander

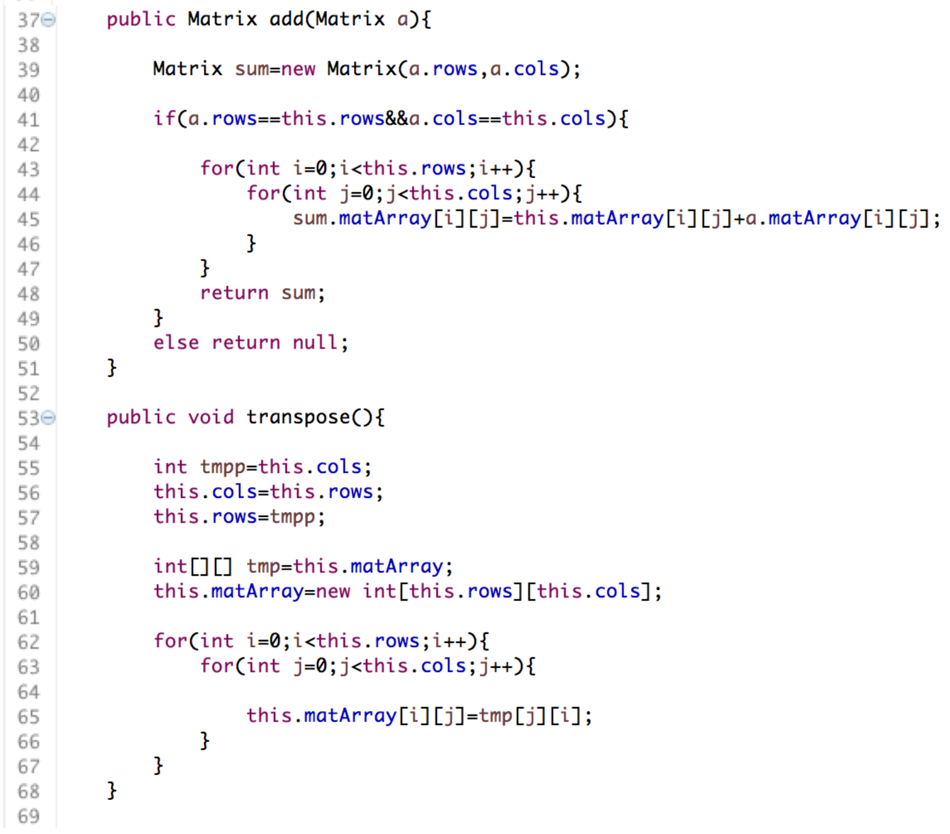
Due Date : 2016.05.12

1. Code of Assignment_2

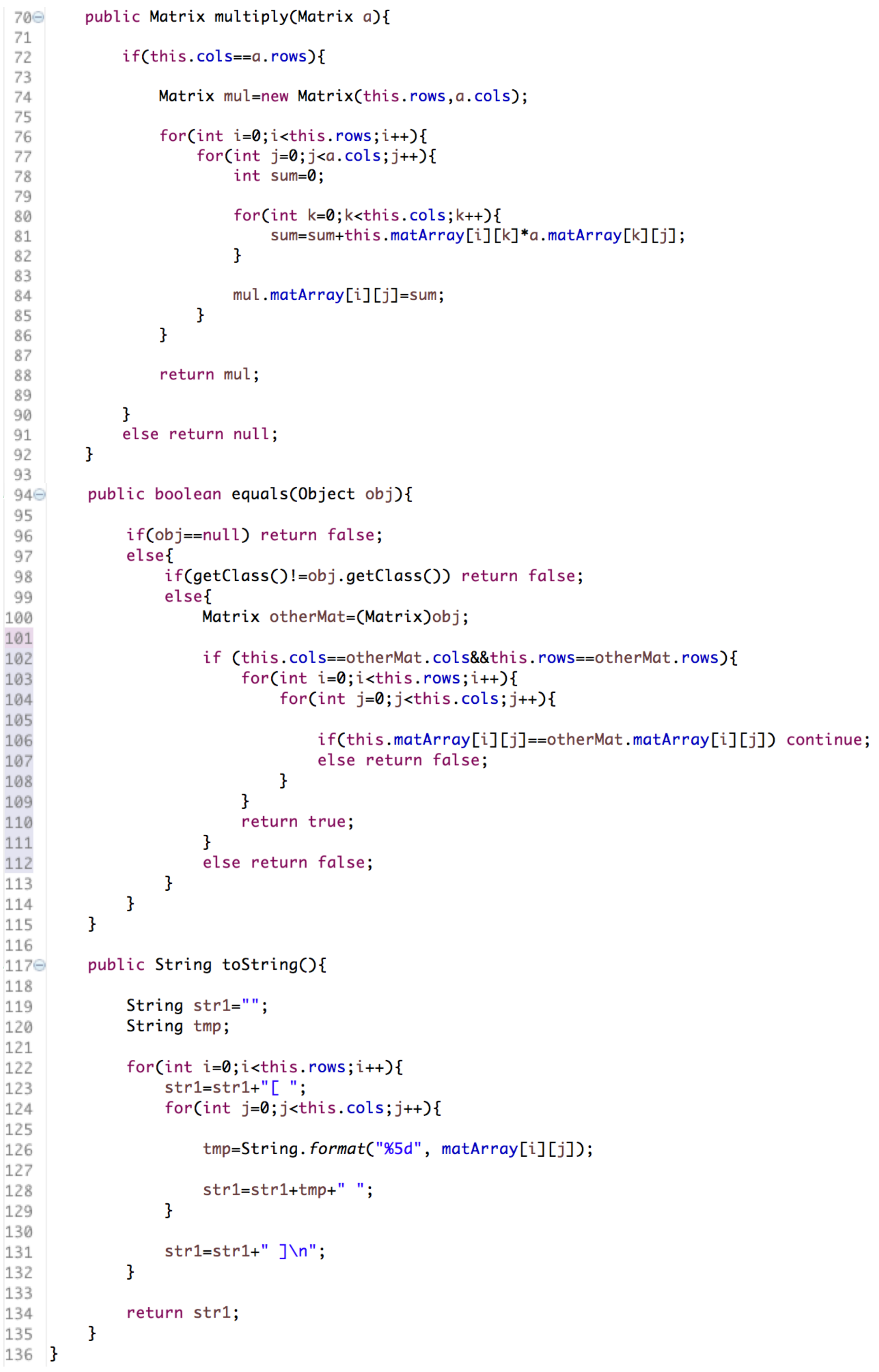
-Matrix Class



```java
package OOP_assignment_2;

public class Matrix {

    int[][] matArray;
    int rows, cols;

    public Matrix(int rows, int cols){

        this.rows=rows;
        this.cols=cols;
        this.matArray=new int[rows][cols];
    }

    public int getRow(){return rows;}

    public int getCols(){return cols;}

    public void setValue(int row,int col,int val){

        this.matArray[row-1][col-1]=val;
    }

    public int getValue(int row,int col){return matArray[row-1][col-1];}

    public void randomize(){

        for(int i=0;i<this.rows;i++){
            for(int j=0;j<this.cols;j++){

                this.matArray[i][j]=(int) (Math.random()*101);

            }
        }
    }

```

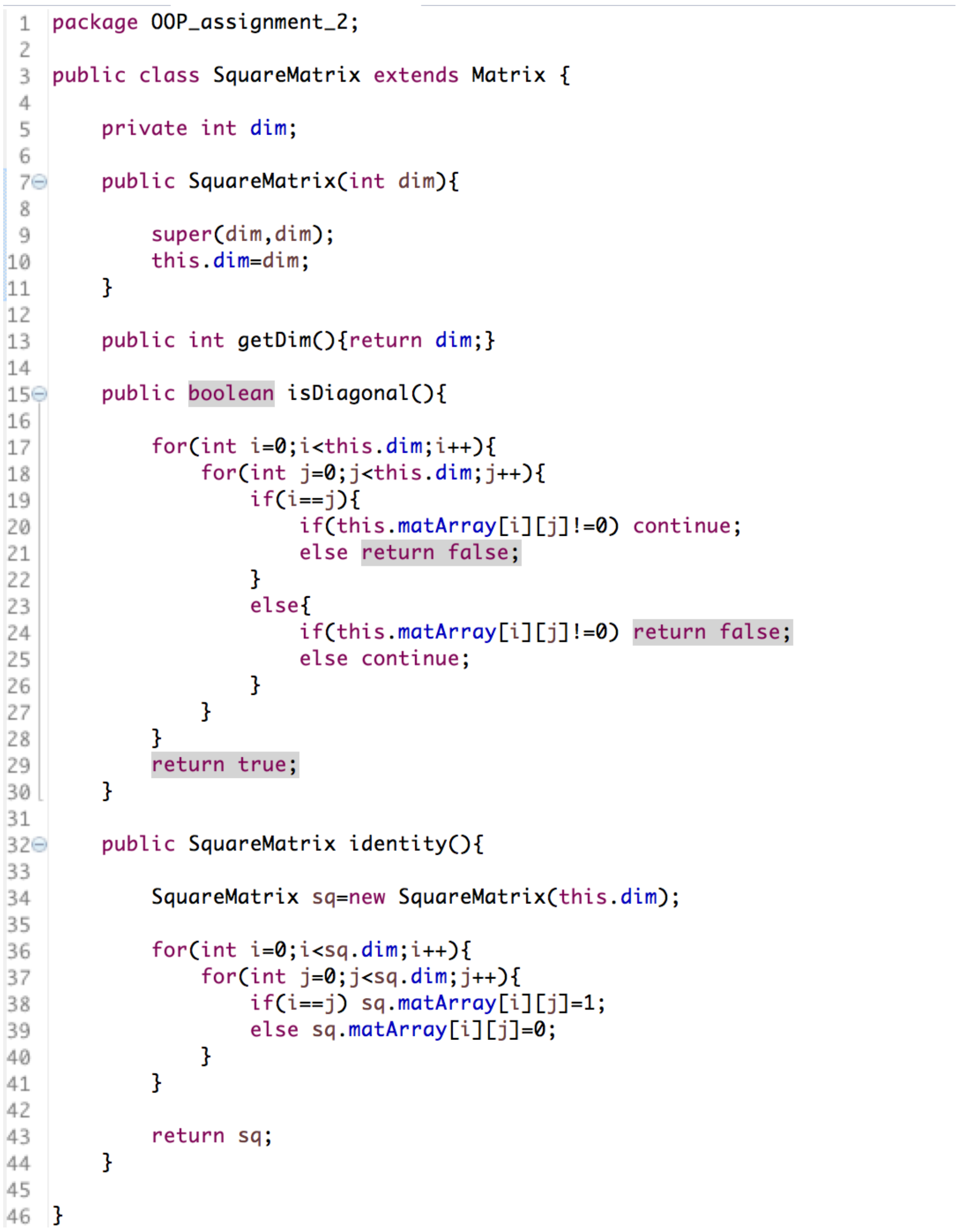


```java
    public Matrix add(Matrix a){

        Matrix sum=new Matrix(a.rows,a.cols);

        if(a.rows==this.rows&&a.cols==this.cols){

            for(int i=0;i<this.rows;i++){
                for(int j=0;j<this.cols;j++){
                    sum.matArray[i][j]=this.matArray[i][j]+a.matArray[i][j];
                }
            }
            return sum;
        }
        else return null;
    }

    public void transpose(){

        int tmpp=this.cols;
        this.cols=this.rows;
        this.rows=tmpp;

        int[][] tmp=this.matArray;
        this.matArray=new int[this.rows][this.cols];

        for(int i=0;i<this.rows;i++){
            for(int j=0;j<this.cols;j++){

                this.matArray[i][j]=tmp[j][i];
            }
        }
    }

```



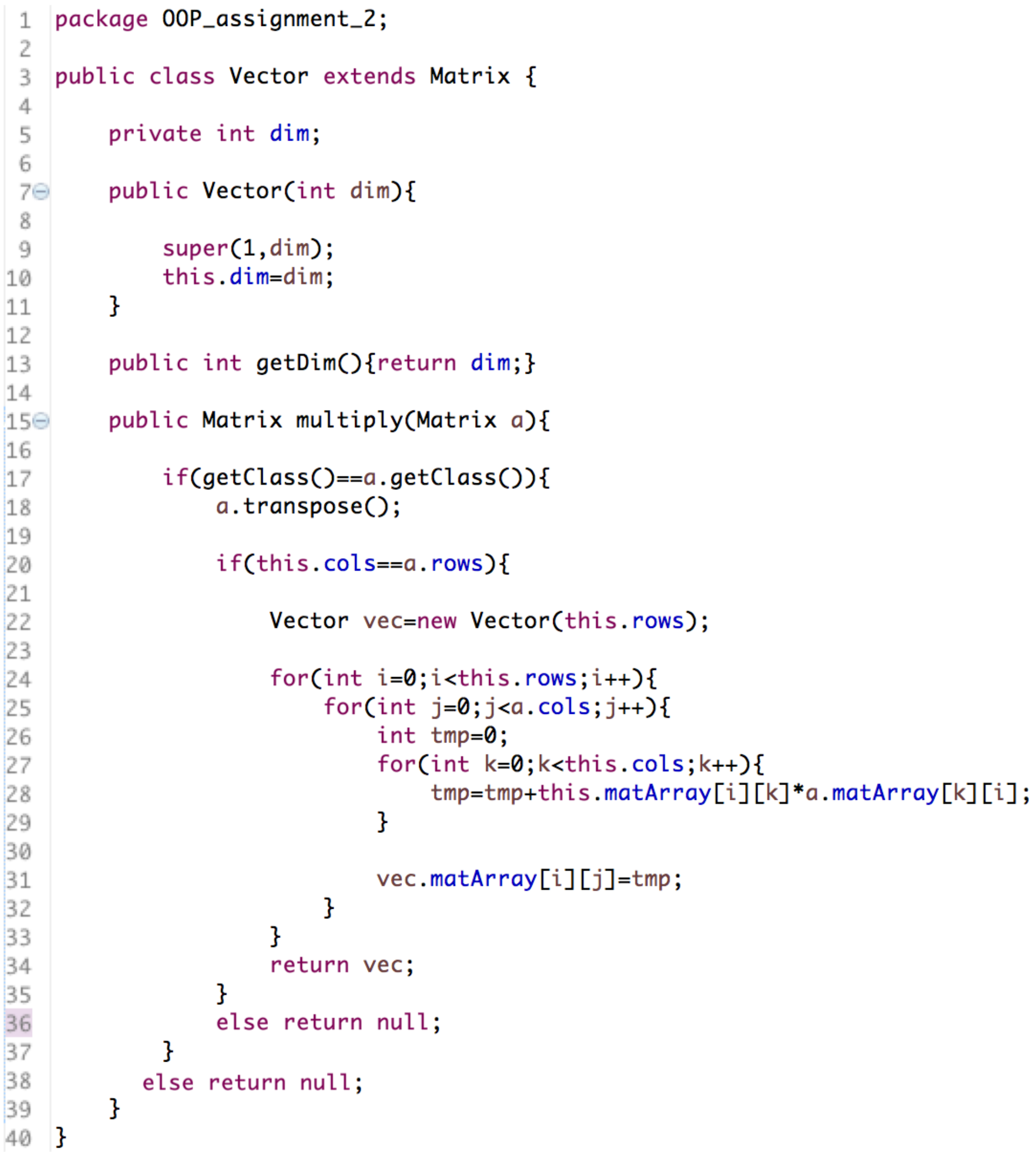
```java
    public Matrix multiply(Matrix a){

        if(this.cols==a.rows){

            Matrix mul=new Matrix(this.rows,a.cols);

            for(int i=0;i<this.rows;i++){
                for(int j=0;j<a.cols;j++){
                    int sum=0;

                    for(int k=0;k<this.cols;k++){
                        sum=sum+this.matArray[i][k]*a.matArray[k][j];
                    }

                    mul.matArray[i][j]=sum;
                }
            }

            return mul;

        }
        else return null;
    }

    public boolean equals(Object obj){

        if(obj==null) return false;
        else{
            if(getClass()!=obj.getClass()) return false;
            else{
                Matrix otherMat=(Matrix)obj;

                if (this.cols==otherMat.cols&&this.rows==otherMat.rows){
                    for(int i=0;i<this.rows;i++){
                        for(int j=0;j<this.cols;j++){

                            if(this.matArray[i][j]==otherMat.matArray[i][j]) continue;
                            else return false;
                        }
                    }
                    return true;
                }
                else return false;
            }
        }
    }

    public String toString(){

        String str1="";
        String tmp;

        for(int i=0;i<this.rows;i++){
            str1=str1+"[ ";
            for(int j=0;j<this.cols;j++){

                tmp=String.format("%5d", matArray[i][j]);

                str1=str1+tmp+" ";
            }

            str1=str1+" ]\n";
        }

        return str1;
    }
}
```

-SquareMatrix Class



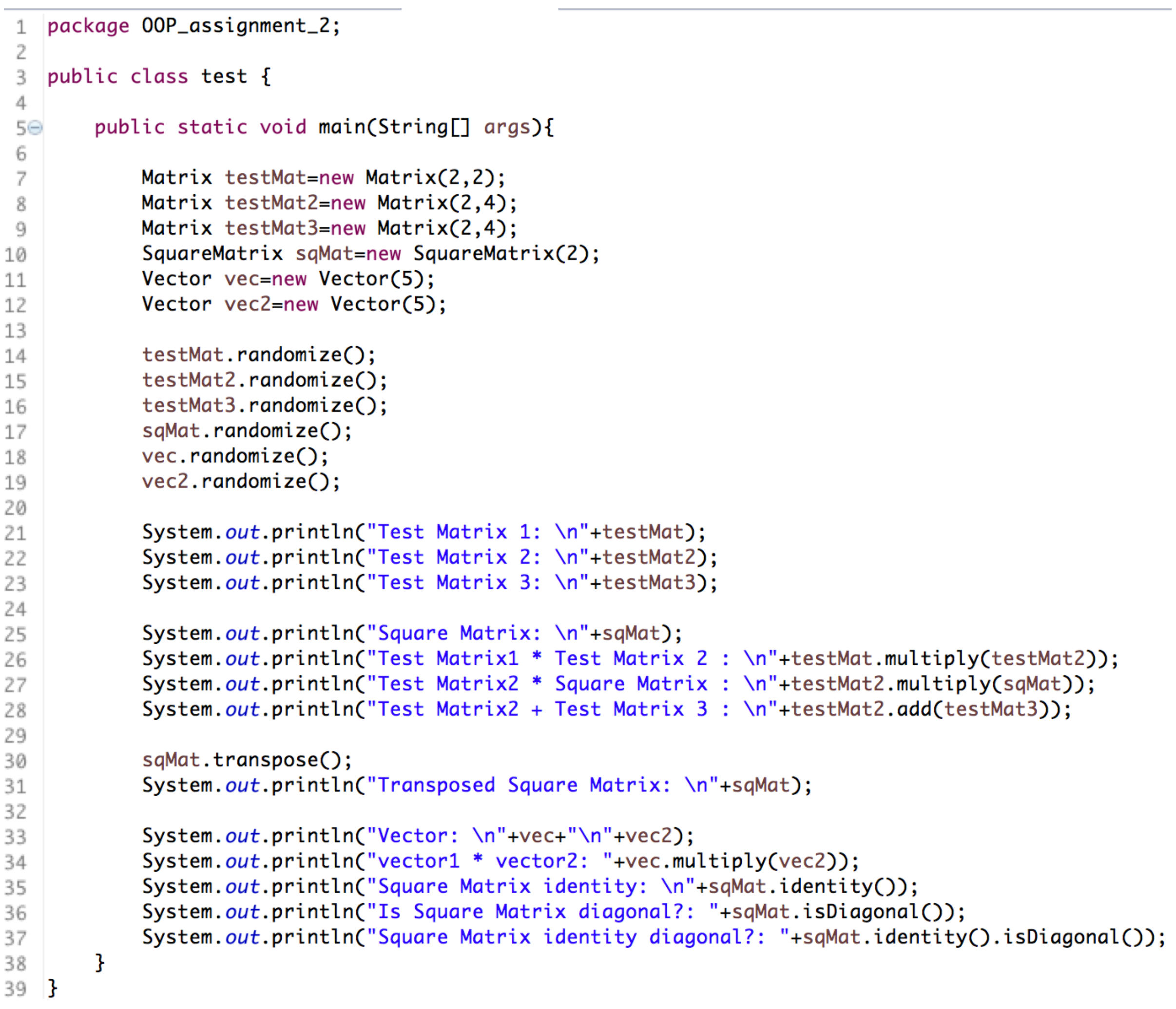
```java
package OOP_assignment_2;

public class SquareMatrix extends Matrix {

    private int dim;

    public SquareMatrix(int dim){

        super(dim,dim);
        this.dim=dim;
    }

    public int getDim(){return dim;}

    public boolean isDiagonal(){

        for(int i=0;i<this.dim;i++){
            for(int j=0;j<this.dim;j++){
                if(i==j){
                    if(this.matArray[i][j]!=0) continue;
                    else return false;
                }
                else{
                    if(this.matArray[i][j]!=0) return false;
                    else continue;
                }
            }
        }
        return true;
    }

    public SquareMatrix identity(){

        SquareMatrix sq=new SquareMatrix(this.dim);

        for(int i=0;i<sq.dim;i++){
            for(int j=0;j<sq.dim;j++){
                if(i==j) sq.matArray[i][j]=1;
                else sq.matArray[i][j]=0;
            }
        }

        return sq;
    }

}
```

-Vector Class



```java
package OOP_assignment_2;

public class Vector extends Matrix {

    private int dim;

    public Vector(int dim){

        super(1,dim);
        this.dim=dim;
    }

    public int getDim(){return dim;}

    public Matrix multiply(Matrix a){

        if(getClass()==a.getClass()){
            a.transpose();

            if(this.cols==a.rows){

                Vector vec=new Vector(this.rows);

                for(int i=0;i<this.rows;i++){
                    for(int j=0;j<a.cols;j++){
                        int tmp=0;
                        for(int k=0;k<this.cols;k++){
                            tmp=tmp+this.matArray[i][k]*a.matArray[k][i];
                        }

                        vec.matArray[i][j]=tmp;
                    }
                }
                return vec;
            }
            else return null;
        }
      else return null;
    }
}
```

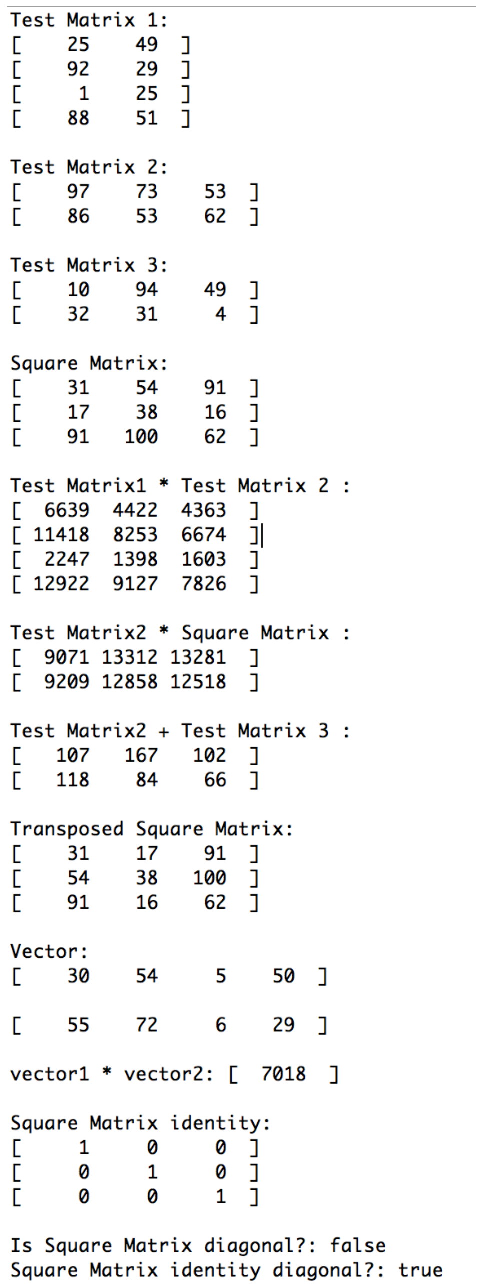
- Main body of program(test for result2)



```java
package OOP_assignment_2;

public class test {

    public static void main(String[] args){

        Matrix testMat=new Matrix(2,2);
        Matrix testMat2=new Matrix(2,4);
        Matrix testMat3=new Matrix(2,4);
        SquareMatrix sqMat=new SquareMatrix(2);
        Vector vec=new Vector(5);
        Vector vec2=new Vector(5);

        testMat.randomize();
        testMat2.randomize();
        testMat3.randomize();
        sqMat.randomize();
        vec.randomize();
        vec2.randomize();

        System.out.println("Test Matrix 1: \n"+testMat);
        System.out.println("Test Matrix 2: \n"+testMat2);
        System.out.println("Test Matrix 3: \n"+testMat3);

        System.out.println("Square Matrix: \n"+sqMat);
        System.out.println("Test Matrix1 * Test Matrix 2 : \n"+testMat.multiply(testMat2));
        System.out.println("Test Matrix2 * Square Matrix : \n"+testMat2.multiply(sqMat));
        System.out.println("Test Matrix2 + Test Matrix 3 : \n"+testMat2.add(testMat3));

        sqMat.transpose();
        System.out.println("Transposed Square Matrix: \n"+sqMat);

        System.out.println("Vector: \n"+vec+"\n"+vec2);
        System.out.println("vector1 * vector2: "+vec.multiply(vec2));
        System.out.println("Square Matrix identity: \n"+sqMat.identity());
        System.out.println("Is Square Matrix diagonal?: "+sqMat.isDiagonal());
        System.out.println("Square Matrix identity diagonal?: "+sqMat.identity().isDiagonal());
    }
}
```

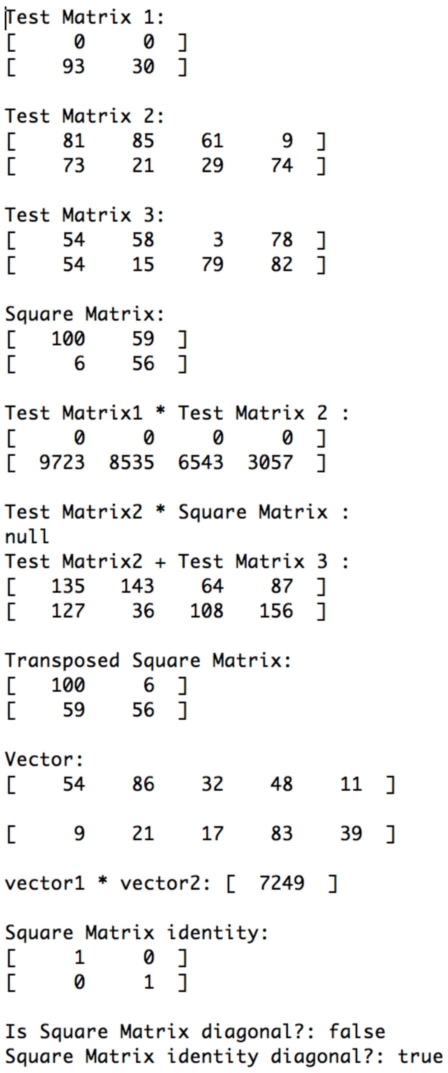
2. Output of Assignment_2

-result_1(In assignment PDF)



```
Test Matrix 1:
[    25    49  ]
[    92    29  ]
[     1    25  ]
[    88    51  ]

Test Matrix 2:
[    97    73    53  ]
[    86    53    62  ]

Test Matrix 3:
[    10    94    49  ]
[    32    31     4  ]

Square Matrix:
[    31    54    91  ]
[    17    38    16  ]
[    91   100    62  ]

Test Matrix1 * Test Matrix 2 :
[  6639  4422  4363  ]
[ 11418  8253  6674  ]
[  2247  1398  1603  ]
[ 12922  9127  7826  ]

Test Matrix2 * Square Matrix :
[  9071 13312 13281  ]
[  9209 12858 12518  ]

Test Matrix2 + Test Matrix 3 :
[   107   167   102  ]
[   118    84    66  ]

Transposed Square Matrix:
[    31    17    91  ]
[    54    38   100  ]
[    91    16    62  ]

Vector:
[    30    54     5    50  ]

[    55    72     6    29  ]

vector1 * vector2: [  7018  ]

Square Matrix identity:
[     1     0     0  ]
[     0     1     0  ]
[     0     0     1  ]

Is Square Matrix diagonal?: false
Square Matrix identity diagonal?: true
```

-result_2(Previous Page)



```
Test Matrix 1:
[     0     0  ]
[    93    30  ]

Test Matrix 2:
[    81    85    61     9  ]
[    73    21    29    74  ]

Test Matrix 3:
[    54    58     3    78  ]
[    54    15    79    82  ]

Square Matrix:
[   100    59  ]
[     6    56  ]

Test Matrix1 * Test Matrix 2 :
[     0     0     0     0  ]
[  9723  8535  6543  3057  ]

Test Matrix2 * Square Matrix :
null
Test Matrix2 + Test Matrix 3 :
[   135   143    64    87  ]
[   127    36   108   156  ]

Transposed Square Matrix:
[   100     6  ]
[    59    56  ]

Vector:
[    54    86    32    48    11  ]

[     9    21    17    83    39  ]

vector1 * vector2: [  7249  ]

Square Matrix identity:
[     1     0  ]
[     0     1  ]

Is Square Matrix diagonal?: false
Square Matrix identity diagonal?: true
```

3. Explanation of Assignment_2

- Matrix Class

  - instance variables:
    - int[][] matArray : Array, int type, 2D, variable. It will contain Matrix in various dimension.
    - int rows : Int type variable. It will contain the row of Matrix.
    - int cols : Int type variable. It will contain the columns of Matrix.
  - public Matrix(int rows, int cols): It's a constructor of Matrix class which is create an empty matrix. This Matrix's dimension will be decided by the variable rows and cols.
  - public int getRow() : This function help to access the rows of the Matrix , and return rows.
  - public int getCols() : This function help to access the cols of the Matrix , and return cols.
  - public void setValue(int row, int col, int val) : This function help to insert the value in the proper position(row-1,col-1) of Matrix.
  - public void setValue(int row, int col, int val) : This function help to access the value in the (row-1,col-1) position of Matrix, and return it.
  - public void randomize() : This function can initialize the Matrix numbers between 0 and 100 randomly. The numbers in Matrix are initialize during loop is continue.
  - public Matrix add (Matrix a) : The function which add the elements of itself and Matrix a. It will return a Matrix (sum of two Matrix). If the rows and cols of a and itself are same, it will add the elements at same position during the loop is continue. But it's not, it will return a null.
  - public void transpose() : The function which transpose the column and row of the Matrix. Matrix will be reinitialized by transposed cols and rows. During the loop is continue, the other array, which is same with the Matrix before transposing, will insert all of elements of cols and raws in proper area.
  - public Matrix multiply(Matrix a) : The function which multiply the elements of itself and Matrix a. It will return a Matrix(multiply of two Matrix). If it's cols and a's row are same, during the loop is continue, the elements at same position will be multiplied and the result of multiply in the same row will be added. If it's not, it will return a null.
  - public boolean equals(Object obj) : The function which decide two matrix are equal or not. If object is null or class is different or two matrixes have different cols,row or elements, it will return false. Else, it will return true.
  - public String toString() : It will return a Matrix in String type. String tmp will get Matrix elements by String.format and insert it in str1. So, Matrix will be returned by String str1.

- SquareMatrix Class

  - This class extends the Matrix class.
  - public SquareMatrix(int dim) : It's a constructor of SquareMatrix class which is create an empty SquareMatrix. This constructor extend the Matrix constructor. It use variable "dim" in spite of cols and rows.
  - public int getDim() : This function help to access the dim of the SquareMatrix , and return dim.

  - **public boolean isDiagonal() :** The function which decide the Matrix is diagonal or not. During the loop is continue, when i==j, if the element of Matrix is not 0, it will continue the matrix checking. Else, it will return false. When i is different with j, if the element of Matrix is not 0, it will return false. Else, it will continue checking. If checking is end, it will return true.
  - **public SquareMatrix identity() :** The function which get dim of the SquareMatrix and return a identity Matrix which has same dimension of SquareMatrix. In the function, a new SquareMatrix is created and be initialized in the loop. It will be 1, if i and j are same. It will be 0 in the other cases.

- Vector Class

  - This class extends the Matrix class.
  - **public Vector(int dim)** : It's a constructor of Vector class which is create an empty Vector. This constructor extend the Matrix constructor. It use variable dim and 1 in spite of cols and rows.
  - **public int getDim :** This function help to access the dim of the SquareMatrix , and return dim.
  - **public Matrix multiply(Matrix a) :** The function which multiply the elements of itself and Matrix a. If two of them have same class(Vector), It will transpose "a" and when this.cols and a.rows are same, during the loop is continue, the elements at same position will be multiplied and the result of multiply in the same row will be added, and return Vector. If it's not, it will return a null. If two of them have different class, it will also return a null.