Homework Turnin

Email:	rgalanos@fcps.edu
Section:	6G
Course:	TJHSST APCS 2016-17
Assignment:	02-06
Receipt ID:	bd9bbc08831e2ae2f8b23ef4d4c029e1

Warning: Your turnin is 3 days late. Assignment 02-06 was due Friday, October 21, 2016, 11:30 PM.

Replacing prior submission from Fri 2016/10/14 12:03pm.

Turnin Successful!

The following file(s) were received:

MatrixRecreate.java (4331 bytes)

```
// Name:   Date:
//import java.util.Arrays;
public class MatrixRecreate
{
   public static void main(String[] args)
   {
      int[][] matrix = TheMatrix.create();
      int[] rowcount = new int[matrix.length];
      int[] colcount = new int[matrix[0].length];
      TheMatrix.count(matrix, rowcount, colcount);
      TheMatrix.display(matrix, rowcount, colcount);
      TheMatrix.re_create(rowcount, colcount);
      TheMatrix.display(TheMatrix.getRecreatedMatrix(), rowcount, colcount);
   }
}
class TheMatrix
{
   //do not instantiate recreatedMatrix yet. Only instantiate and set that in recur.
   private static int[][] recreatedMatrix;

   public static int[][] getRecreatedMatrix()
   {
      return recreatedMatrix;
   }
   public static int[][] create()
   {
      int rows = (int)(Math.random()*5+2);
      int columns = (int)(Math.random()*5+2);
      double rand;

      int[][] matrix = new int[rows][columns];

      for(int i=0;i<matrix.length;i++)
      {
         for(int a=0;a<matrix[0].length;a++)
         {
            rand = Math.random();
            if(rand<0.5)
               matrix[i][a] = 0;
            else
```

```
                matrix[i][a] = 1;
            }
        }
        return matrix;
    }
    public static void count(int[][] matrix, int[] rowcount, int[] colcount)
    {
        for(int i=0;i<matrix.length;i++)
        {
            for(int a=0;a<matrix[0].length;a++)
            {

                if(matrix[i][a] == 1)
                {
                    rowcount[i]++;
                    colcount[a]++;
                }
            }
        }
    }
    public static void display(int[][] matrix, int[] rowcount, int[] colcount)
    {

        System.out.print("  ");
        for(int i=0;i<colcount.length;i++)
        {
            System.out.print(""+colcount[i]);
        }

        System.out.print("\n  ");

        for(int i=0;i<colcount.length;i++)
        {
            System.out.print("-");
        }

        System.out.print("\n");

        for(int i=0;i<matrix.length;i++)
        {
            System.out.print(rowcount[i]+"|");

            for(int a=0;a<matrix[0].length;a++)
            {
                System.out.print(""+matrix[i][a]);
            }
            System.out.print("\n");
        }


    }
    //should call recur.
    public static void re_create(int[] rowcount, int[] colcount)
    {
        recreatedMatrix = new int[rowcount.length][colcount.length];
        recur(recreatedMatrix, rowcount, colcount, 0, -1);
    }
    private static void recur(int[][] m, int[] rowcount, int[] colcount, int row, int col)
    {
        if(compare(m, rowcount, colcount))    //base case: if new matrix works, then copy over to recreatedMatrix
        {
            //copy over from m to recreatedMatrix (not just references)
            recreatedMatrix = new int[m.length][];
            for(int i = 0; i < m.length; i++)
            {
                recreatedMatrix[i] = new int[m[i].length];
                for (int j = 0; j < m[i].length; j++)
                {
                    recreatedMatrix[i][j] = m[i][j];
                }
            }    //we're done!

        }

        else
        {
            if(col<colcount.length-1)
            {
                col++;
                m[row][col] = 0;
                recur(m,rowcount,colcount,row,col);
```

```
            m[row][col] = 1;
            recur(m,rowcount,colcount,row,col);
         }
         else
         {
            if(row<rowcount.length-1)
            {
               m[row][col] = 0;
               recur(m,rowcount,colcount,row+1,-1);
               m[row][col] = 1;
               recur(m,rowcount,colcount,row+1,-1);

            }
         }
      }
   }
   private static boolean compare(int[][] m, int[] rowcount, int[] colcount)
   {
      int[] tempRowCount = new int[rowcount.length];
      int[] tempColCount = new int[colcount.length];

      boolean bool=true;

      count(m,tempRowCount,tempColCount);

      for(int i=0;i<rowcount.length;i++)
         if(tempRowCount[i]!=rowcount[i])
            bool = false;
      for(int i=0;i<colcount.length;i++)
         if(tempColCount[i]!=colcount[i])
            bool = false;
      return bool;
      /*
      if(Arrays.equals(tempRowCount, rowcount)&& Arrays.equals(tempColCount, colcount))
      {
         return true;
      }
      else
         return false;*/
   }
}
```