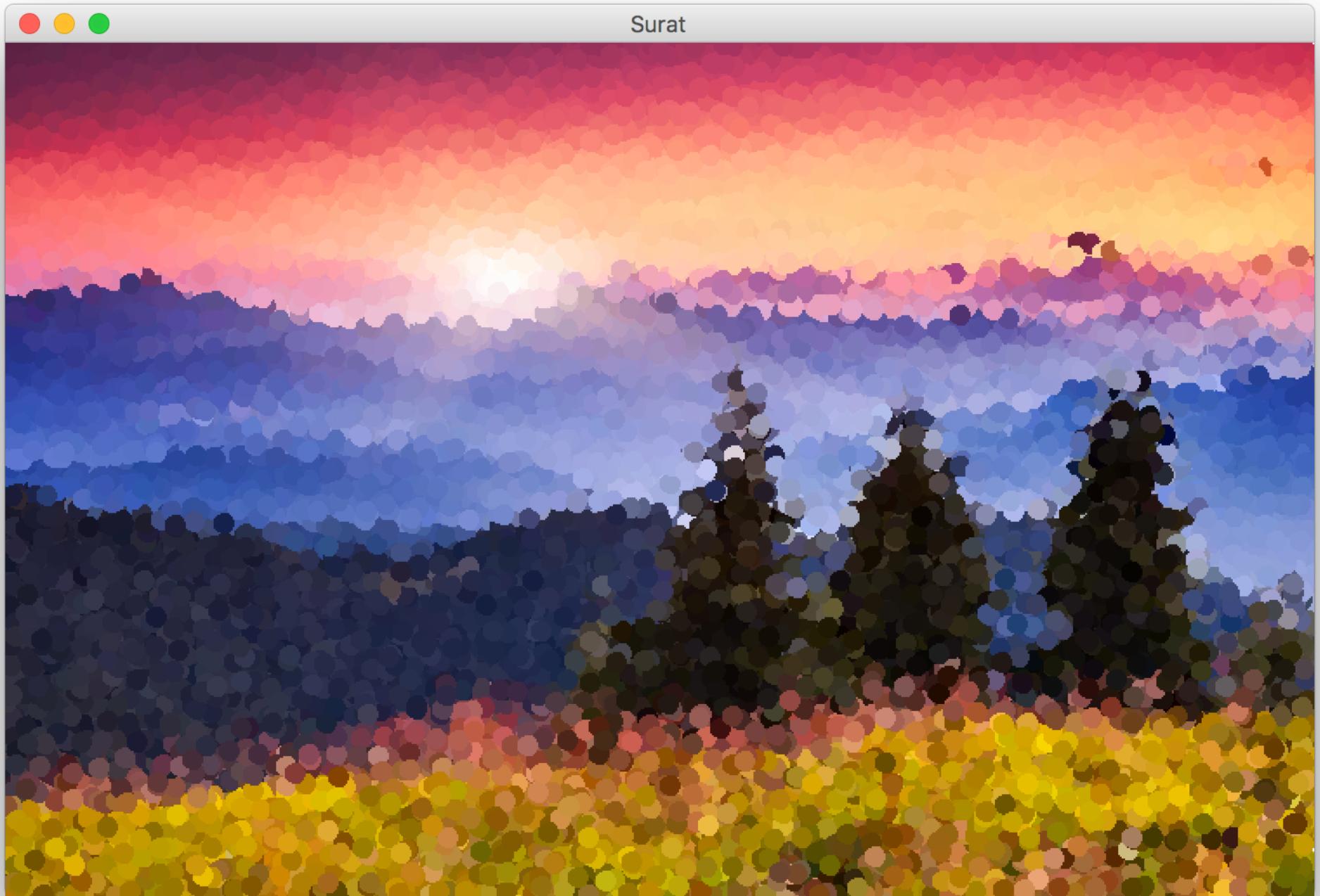


# The Matrix

Chris Piech

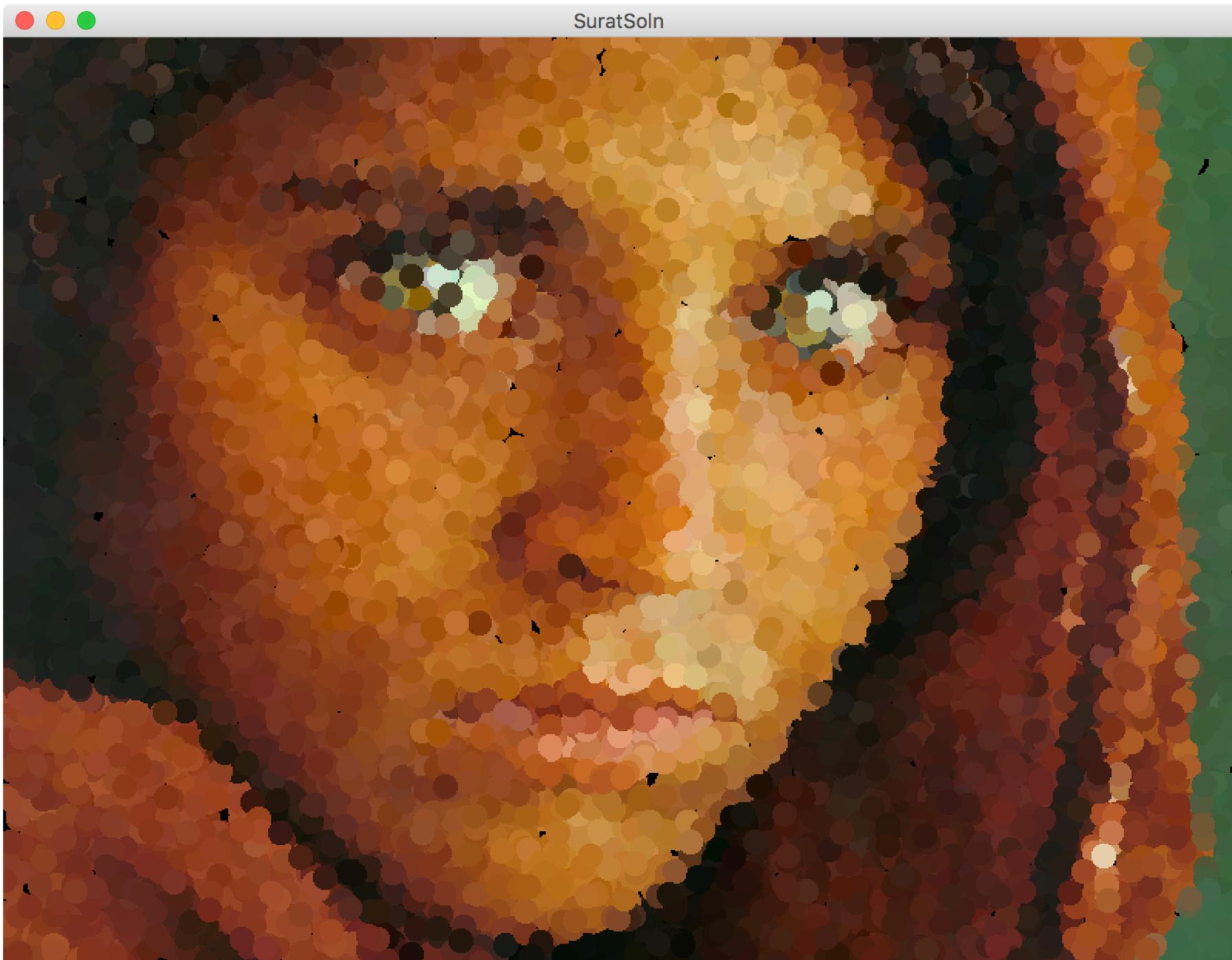
CS106A, Stanford University



Surat

Piech, CS106A, Stanford University



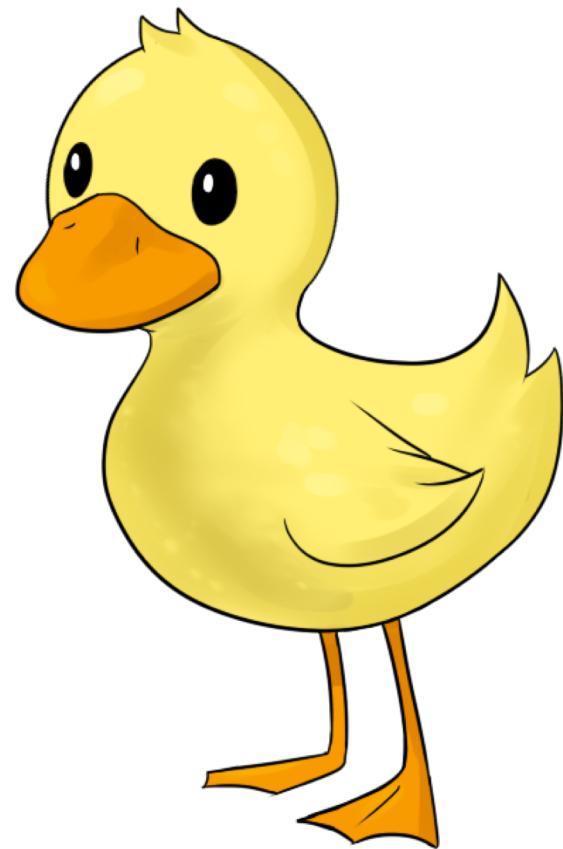


**Photo of Sharbat Gula by Steve McCurry**



Dim the lights

Value:  
Yellow

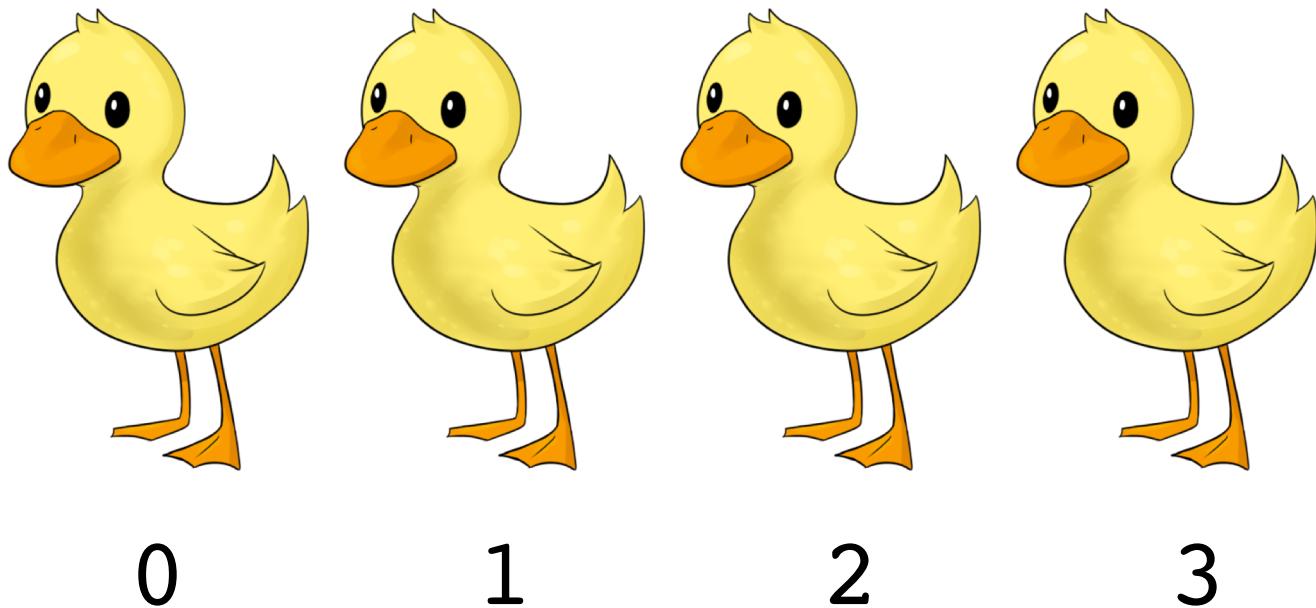


Type:  
Duck

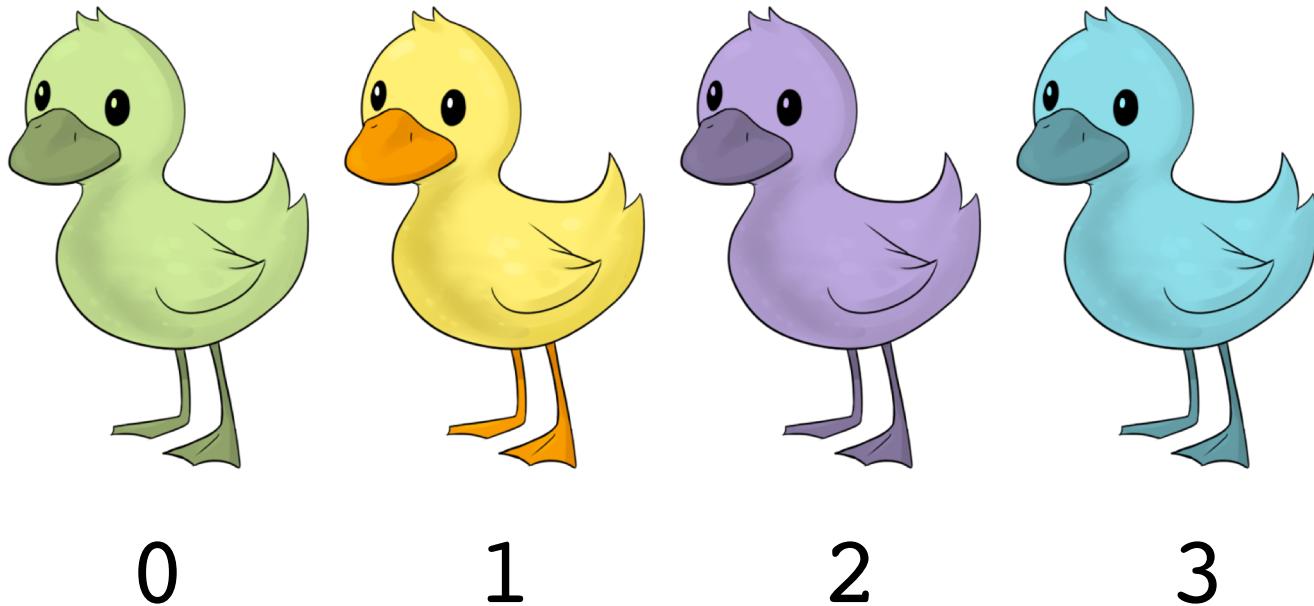
Metaphor for  
a bucket in memory



```
Duck[ ] duckArray = new Duck[ 4 ];
```



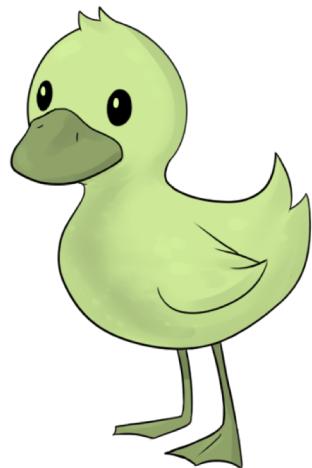
```
Duck[ ] duckArray = new Duck[ 4 ];
```



```
ArrayList<Duck> duckList = new ArrayList<Duck>();
```



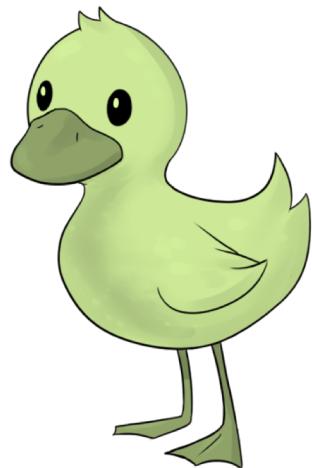
```
ArrayList<Duck> duckList = new ArrayList<Duck>();  
duckList.add(  );
```



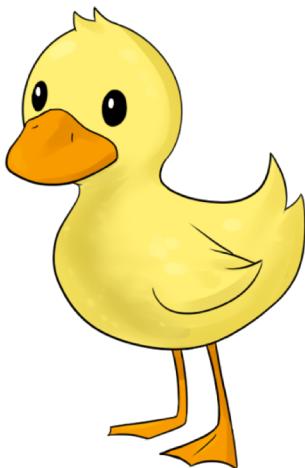
0



```
ArrayList<Duck> duckList = new ArrayList<Duck>();  
duckList.add(  );  
  
duckList.add(  );
```

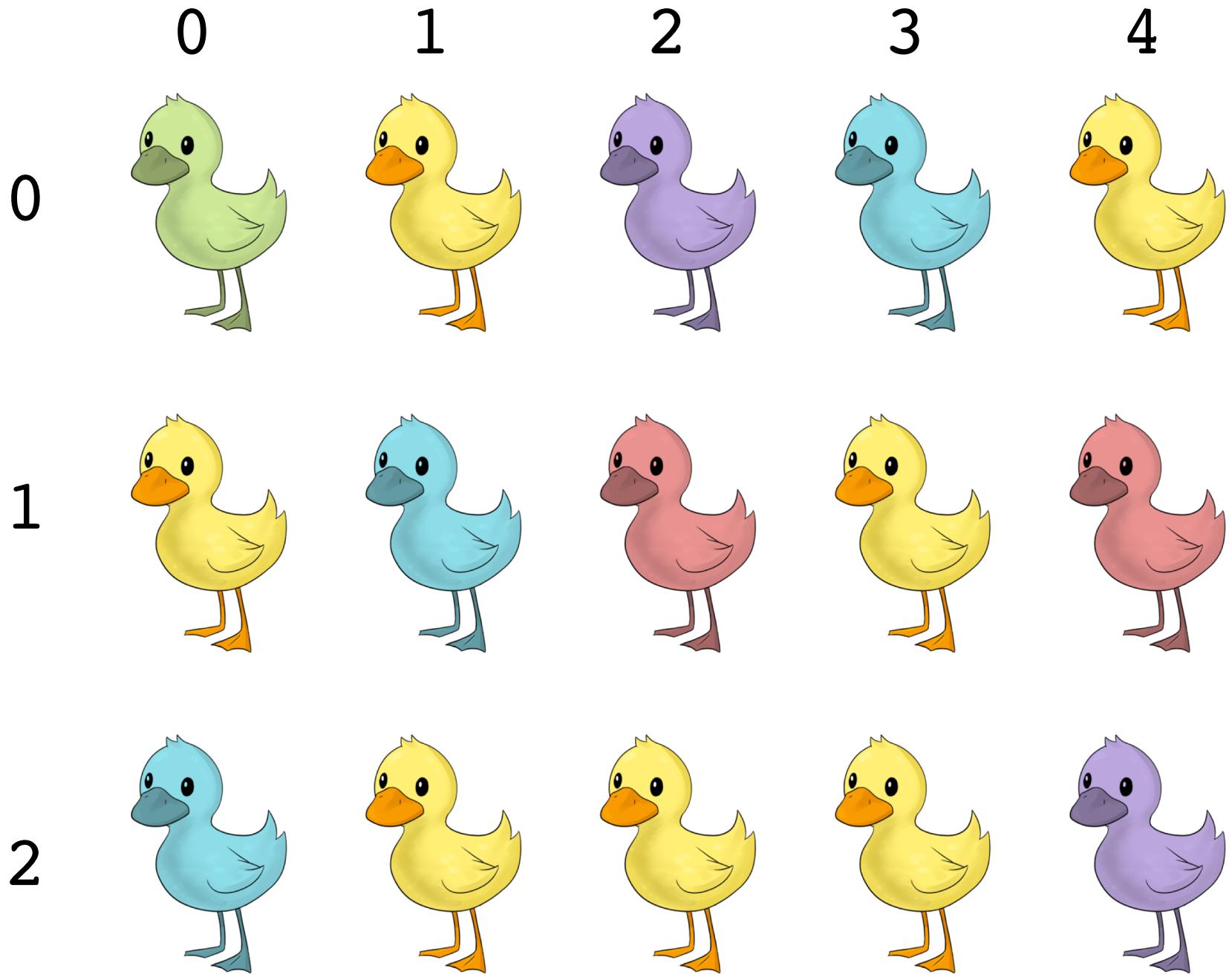


0



1





\* Attack of the clone army of ducks! Hide your children!



# The Matrix



# The Matrix



WELCOME .... TO  
THE MATRIX!!!!!!

a.k.a. 2D arrays



# My First Matrix

```
int[][][] morpheus = new int[2][4];
```



# My First Matrix

```
int[][] morpheus = new int[2][4];
```



# My First Matrix

```
int[][] morpheus = new int[2][4];
```



# My First Matrix

```
int[][] morpheus = new int[2][4];
```



# My First Matrix

```
int[][] morpheus = new int[2][4];
```



# My First Matrix

```
int[][][] morpheus = new int[2][4];
```

Number of cols  
Number of rows



# My First Matrix

```
int[][] morpheus = new int[2][4];
```

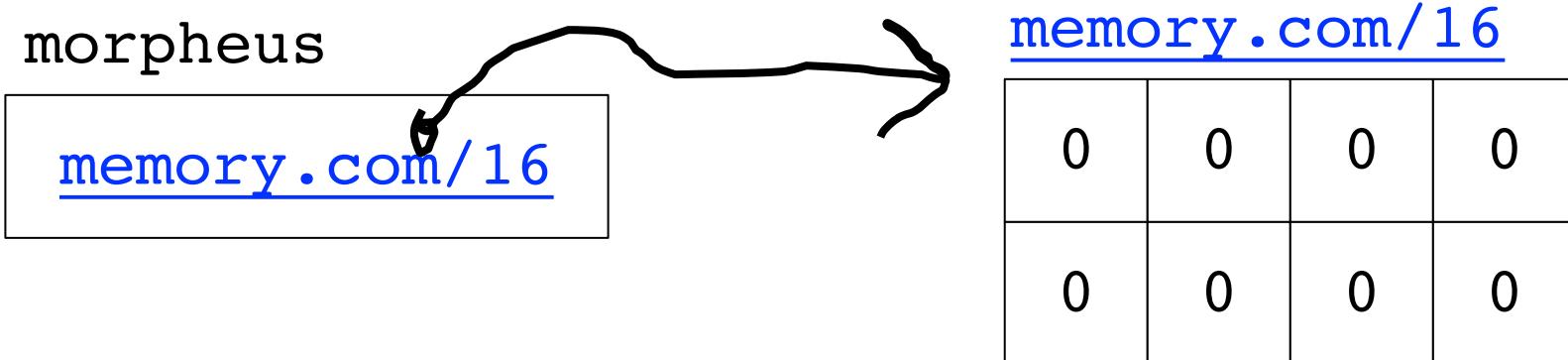
morpheus

0	0	0	0
0	0	0	0



# My First Matrix

```
int[][] morpheus = new int[2][4];
```

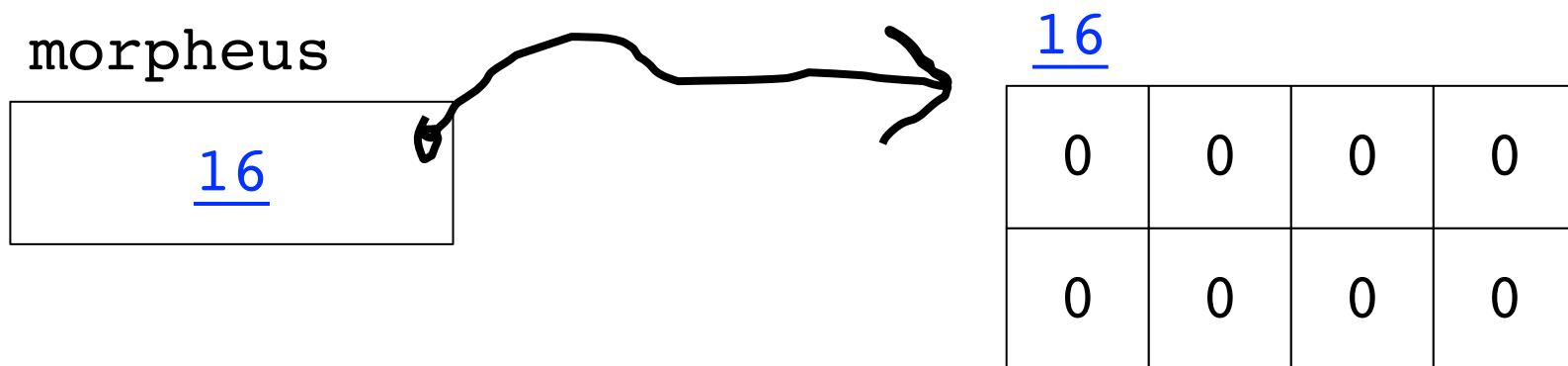


Aside: It's actually more like this.



# My First Matrix

```
int[][] morpheus = new int[2][4];
```

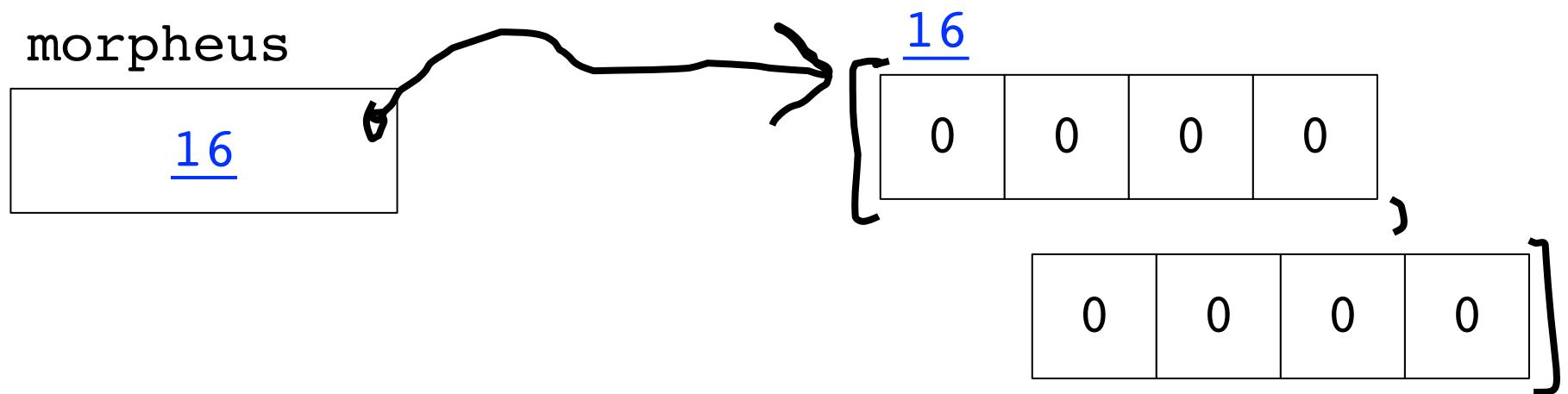


Aside: It's actually more like this.



# My First Matrix

```
int[][] morpheus = new int[2][4];
```



If we are going to be brutally honest



# My First Matrix

```
int[][] morpheus = new int[2][4];
```

morpheus

0	0	0	0
0	0	0	0



# My First Matrix

```
int[][] morpheus = new int[2][4];
```

morpheus

0	0	0	0
0	0	0	0

Task: Make this cell hold the value 1



```
int[][][] morpheus = new int[2][4];
```

	Col 0	Col 1	Col 2	Col 3
Row 0	0	0	0	1
Row 1	0	0	0	0

Task: Make this cell hold the value 1



# My First Matrix

```
morpheus[0][3] = 1;
```

Row 0

morpheus

Col 3

0	0	0	0
0	0	0	0



# My First Matrix

```
morpheus[0][3] = 1;
```

morpheus

0	0	0	1
0	0	0	0



When “indexing” into a matrix,  
row comes first, then column.

```
myMatrix[      row      ] [      col      ]
```



When “indexing” into a matrix,  
row comes first, then column.



R is for stanford<sup>D</sup>

myMatrix[



C is for Cal



When “indexing” into a matrix,  
row comes first, then column.



Matrix: The revolutions

# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```

*Before the method call:*

matrix

0	0	4	0
0	17	0	0
0	0	0	6



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```

*After the method call:*

matrix

1	1	1	1
1	1	1	1
1	1	1	1



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}  
  
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}  
  
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

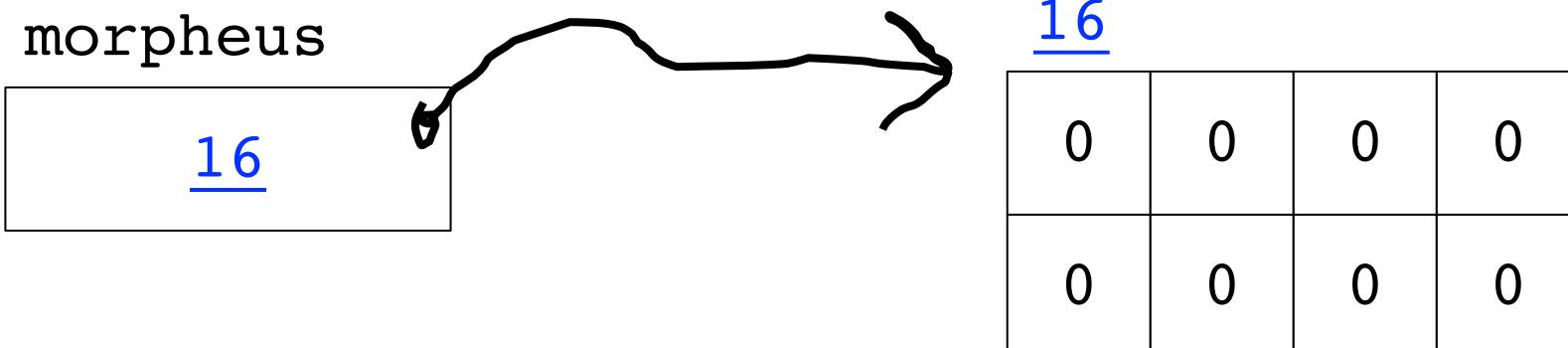
```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}  
  
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```

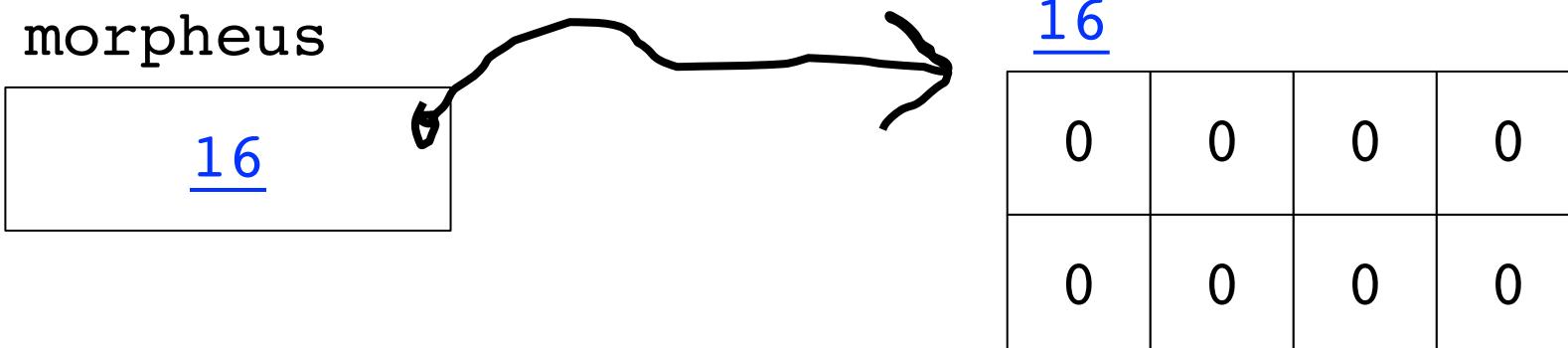
```
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```

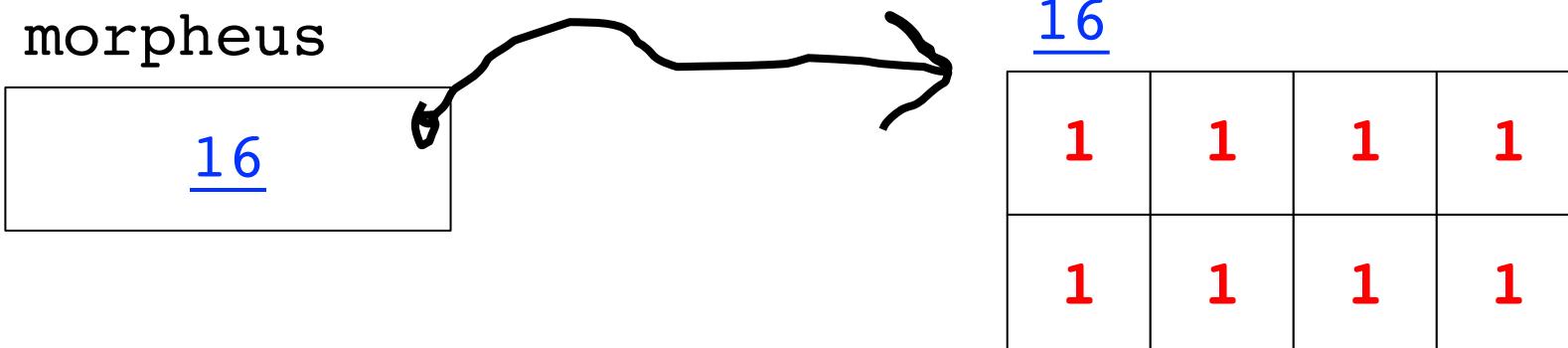
```
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```

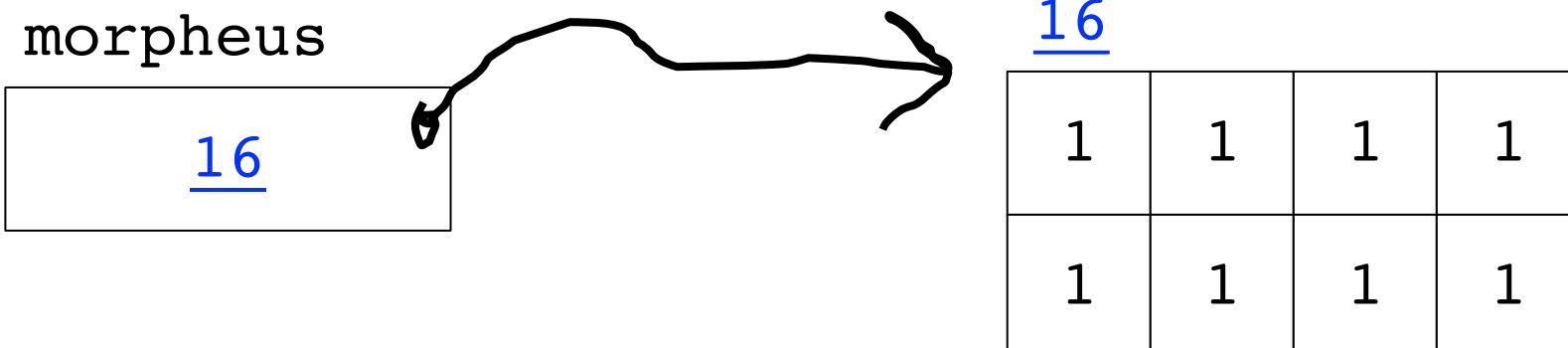
```
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```

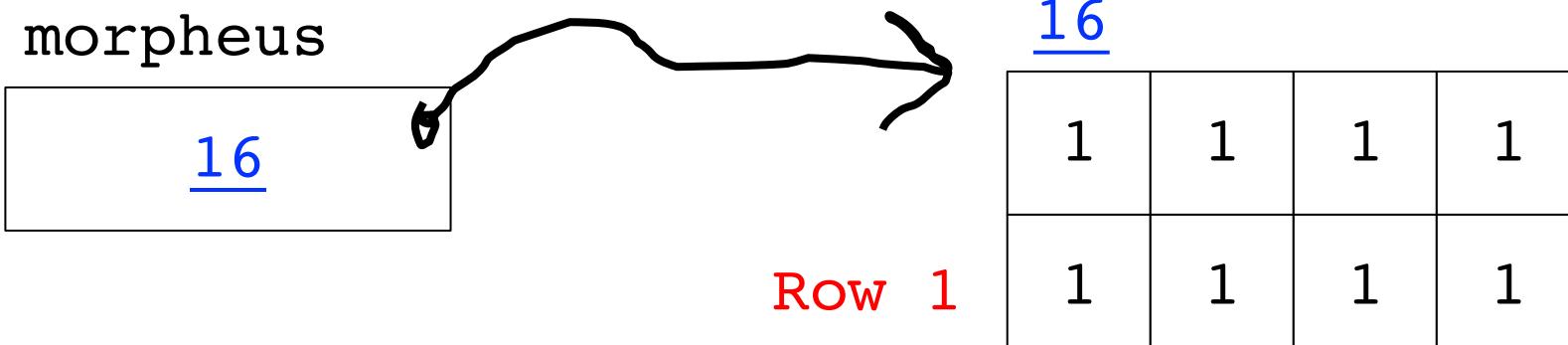
```
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```

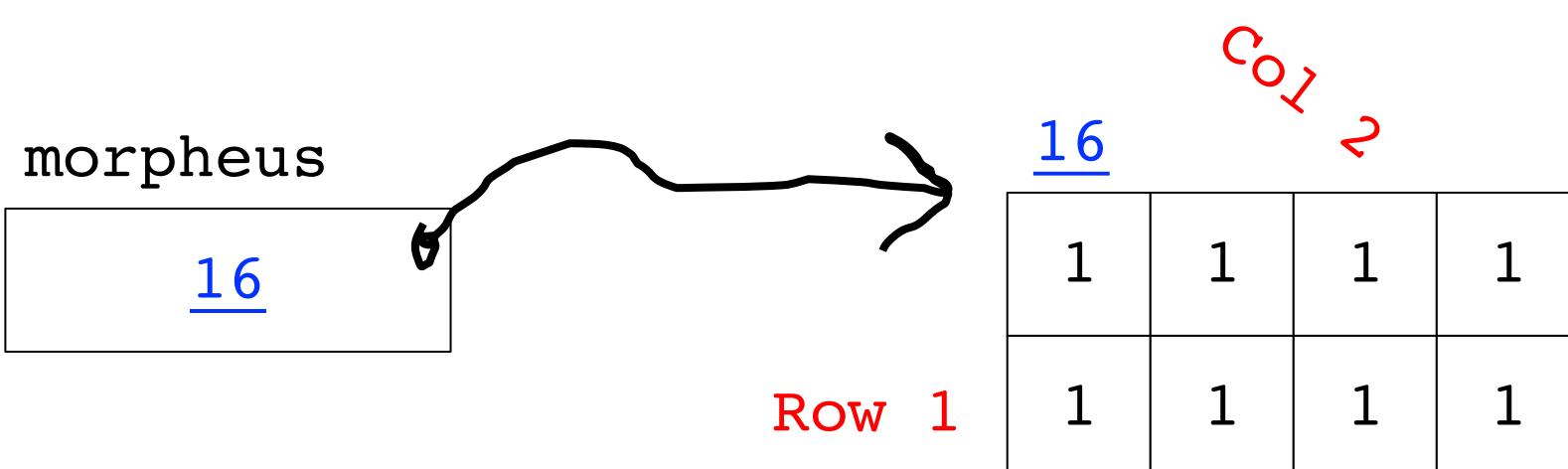
```
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```

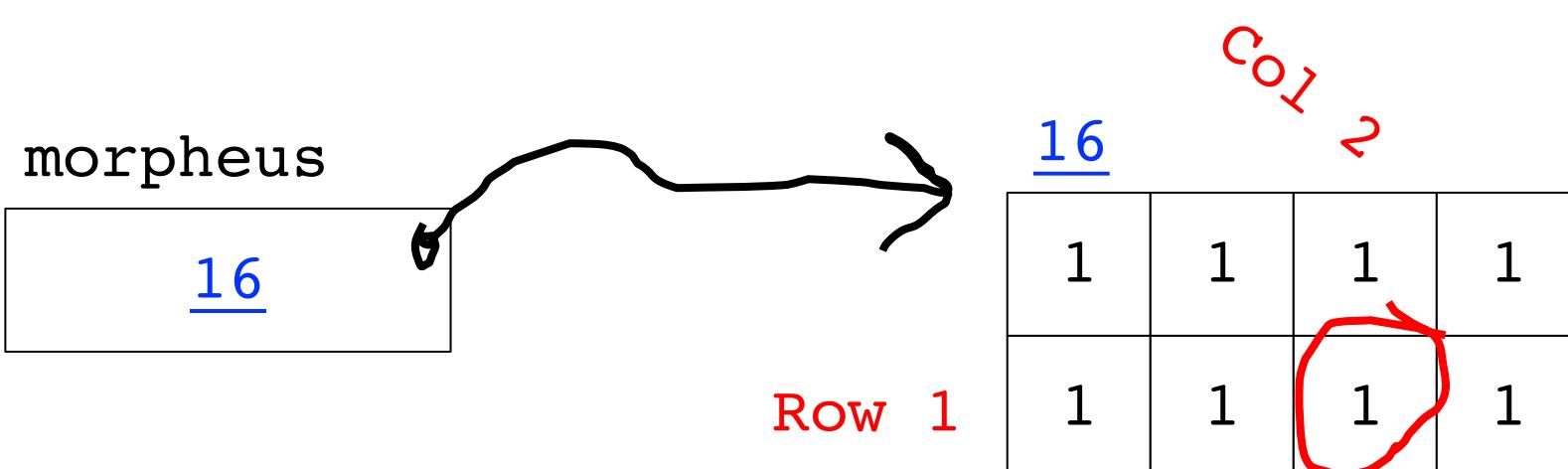
```
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```

```
public void run(){  
    int[][][] morpheus = new int[4][2];  
    setValuesToOne(morpheus);  
    println(morpheus[1][2]);  
}
```



# Set Values to One

```
private void setValuesToOne(int[][][] matrix) {  
    // your code here...  
}
```



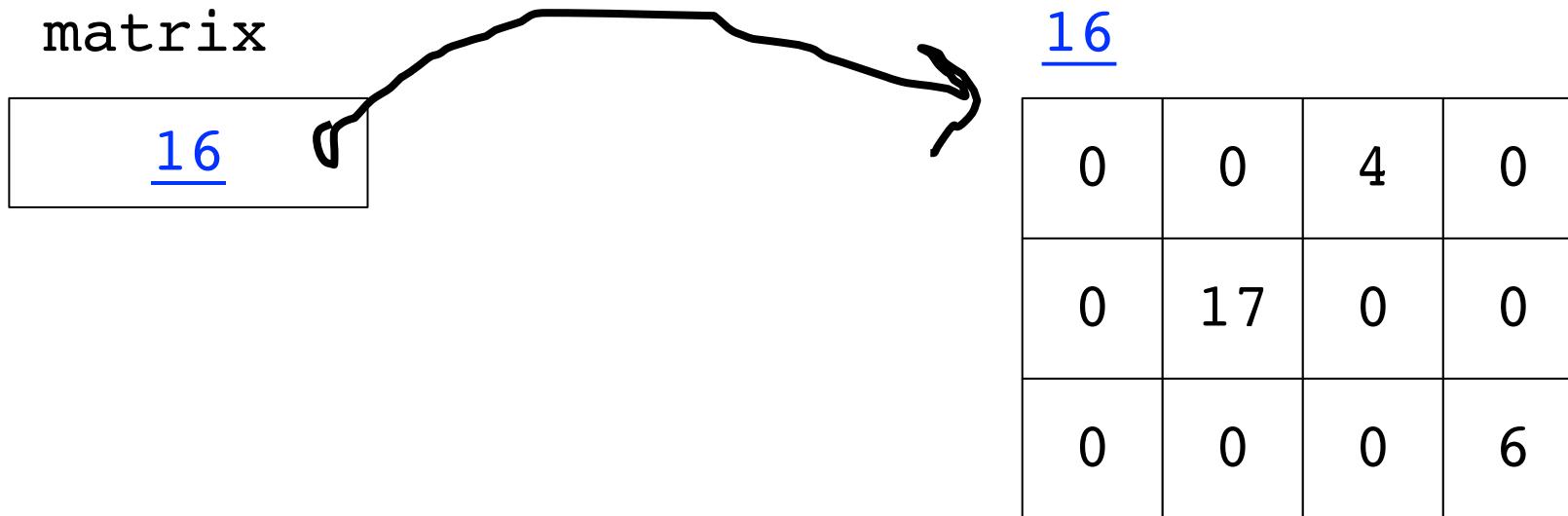
# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    // your code here...  
}
```



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    // your code here...  
}
```



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    // your code here...  
}
```

matrix

0	0	4	0
0	17	0	0
0	0	0	6



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    for(each row r) {  
        for(each col c) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```

matrix

0	0	4	0
0	17	0	0
0	0	0	6



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    for(int r = 0; r < 3; r++) {  
        for(int c = 0; c < 4; c++) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```

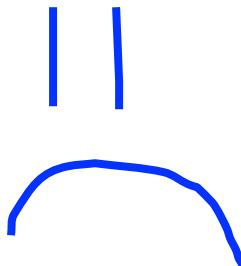
matrix

0	0	4	0
0	17	0	0
0	0	0	6



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    for(int r = 0; r < 3; r++) {  
        for(int c = 0; c < 4; c++) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```



matrix

0	0	4	0	4	0
0	17	0	0	0	0



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    for(int r = 0; r < matrix.length; r++) {  
        for(int c = 0; c < matrix[0].length; c++) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```

Works for  
this matrix

matrix

0	0	4	0	4	0
0	17	0	0	0	0



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    for(int r = 0; r < matrix.length; r++) {  
        for(int c = 0; c < matrix[0].length; c++) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```

Also works for  
this matrix

matrix

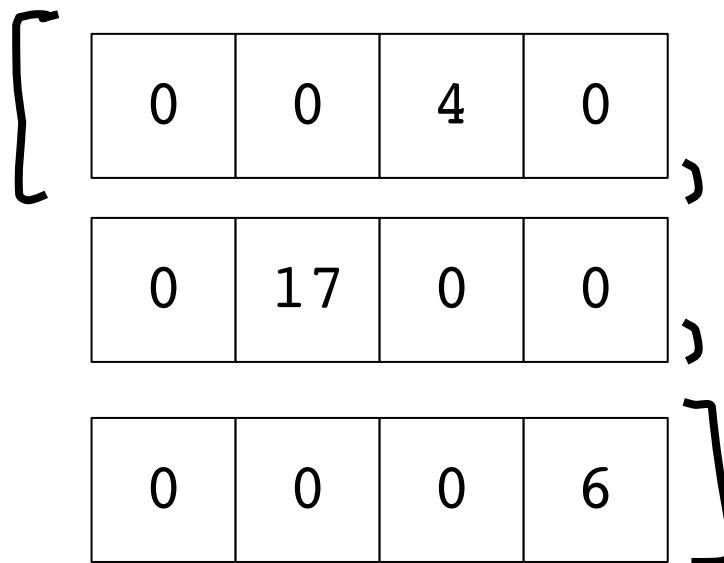
0	0	4	0
0	17	0	0
0	0	0	6



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    for(int r = 0; r < matrix.length; r++) {  
        for(int c = 0; c < matrix[0].length; c++) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```

matrix



# Set Values to One

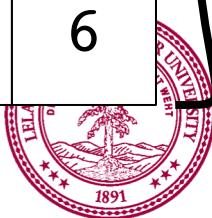
```
private void setValuesToOne(int[][] matrix) {  
    for(int r = 0; r < matrix.length; r++) {  
        for(int c = 0; c < matrix[0].length; c++) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```

matrix

0	0	4	0
---	---	---	---

0	17	0	0
---	----	---	---

0	0	0	6
---	---	---	---



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    for(int r = 0; r < matrix.length; r++) {  
        for(int c = 0; c < matrix[0].length; c++) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```

matrix

0	0	4	0
---	---	---	---

0

0	17	0	0
---	----	---	---

1

0	0	0	6
---	---	---	---

2



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    for(int r = 0; r < matrix.length; r++) {  
        for(int c = 0; c < matrix[0].length; c++) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```

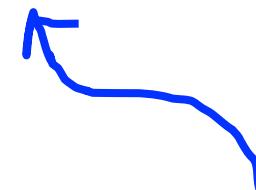
matrix

0	0	4	0
0	17	0	0
0	0	0	6



# Set Values to One

```
private void setValuesToOne(int[][] matrix) {  
    for(int r = 0; r < numRows(matrix); r++) {  
        for(int c = 0; c < numCols(matrix); c++) {  
            matrix[r][c] = 1;  
        }  
    }  
}
```



These aren't defined.  
But I highly recommend  
them :-)

matrix

0	0	4	0
0	17	0	0
0	0	0	6



# How to get number of rows

```
private int numRows(int[][] matrix) {  
    return matrix.length;  
}
```



# How to get number of cols

```
private int numCols(int[][] matrix) {  
    return matrix[0].length;  
}
```



# 2D Arrays on one slide

## 1. Make a Matrix

```
double[][] mahMatrix = new double[nRows][nCols];
```

## 2. Set and get values from a matrix using bracket notation

```
mahMatrix[4][2] = 99.0;  
println(mahMatrix[0][0]);
```

## 3. Get the number of rows and columns of a matrix (pro-tip: define method)

```
int nRows = mahMatrix.length;  
int nCols = mahMatrix[0].length
```

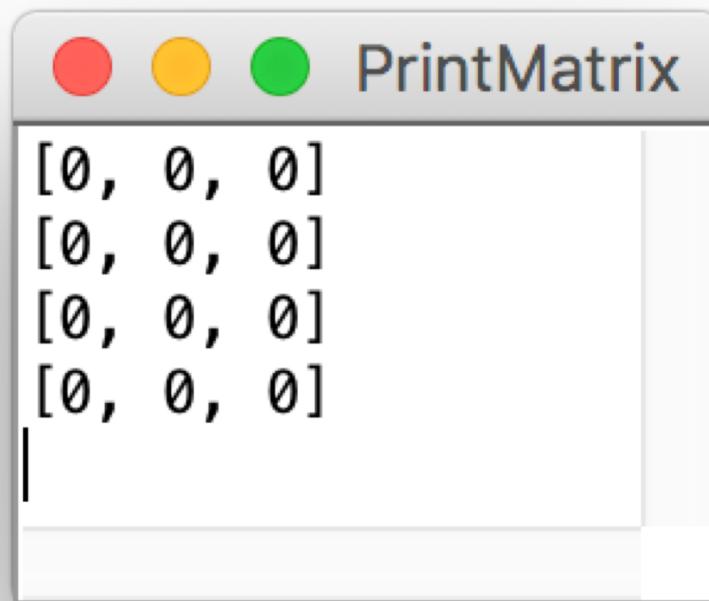
## 4. Use a double for loop to iterate over the whole matrix

```
for(int r = 0; r < mahMatrix.length; r++) {  
    for(int c = 0; c < mahMatrix[0].length; c++) {  
        //party pixel: mahMatrix[r][c]  
    }  
}
```



Your turn: print matrix

```
private void printMatrix(int[][] matrix){  
    // TODO: fill this in  
}
```



# A deadly lack of style...

```
private void setValuesToTwo(int[][] matrix) {  
    for(int i = 0; i < matrix[0].length; i++) {  
        for(int j = 0; j < matrix.length; j++) {  
            matrix[i][j] = 2;  
        }  
    }  
}
```



# A deadly lack of style...

```
private void setValuesToTwo(int[][][] matrix) {  
    for(int i = 0; i < matrix[0].length; i++) {  
        for(int j = 0; j < matrix.length; j++) {  
            matrix[i][j] = 2;  
        }  
    }  
}
```

# A deadly lack of style...

```
private void setValuesToTwo(int[][] matrix) {  
    for(int i = 0; i < matrix[0].length; i++) {  
        for(int j = 0; j < matrix.length; j++) {  
            matrix[i][j] = 2;  
        }  
    }  
}
```



# A deadly lack of style...

```
private void setValuesToTwo(int[][] matrix) {  
    for(int i = 0; i < matrix[0].length; i++) {  
        for(int j = 0; j < matrix.length; j++) {  
            matrix[i][j] = 2;  
        }  
    }  
}
```



# A deadly lack of style...

```
private void setValuesToTwo(int[][] matrix) {  
    for(int r = 0; r < matrix[0].length; r++) {  
        for(int j = 0; j < matrix.length; j++) {  
            matrix[r][j] = 2;  
        }  
    }  
}
```



# A deadly lack of style...

```
private void setValuesToTwo(int[][] matrix) {  
    for(int r = 0; r < matrix[0].length; r++) {  
        for(int c = 0; c < matrix.length; c++) {  
            matrix[r][c] = 2;  
        }  
    }  
}
```



# A deadly lack of style...

```
private void setValuesToTwo(int[][] matrix) {  
    for(int r = 0; r < numRows(matrix); r++) {  
        for(int c = 0; c < numCols(matrix); c++) {  
            matrix[r][c] = 2;  
        }  
    }  
}
```



# Images are Matrices!



# Images are Matrices!

```
GImage img = new GImage("snowman.jpg");  
int[][] pixels = img.getPixelArray();
```

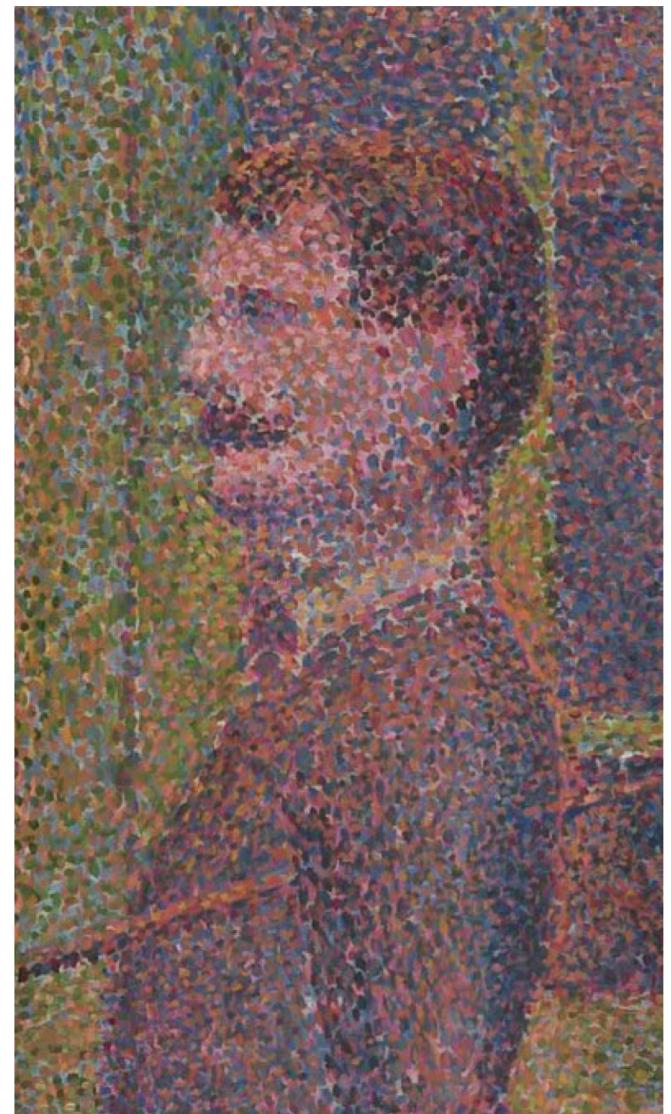
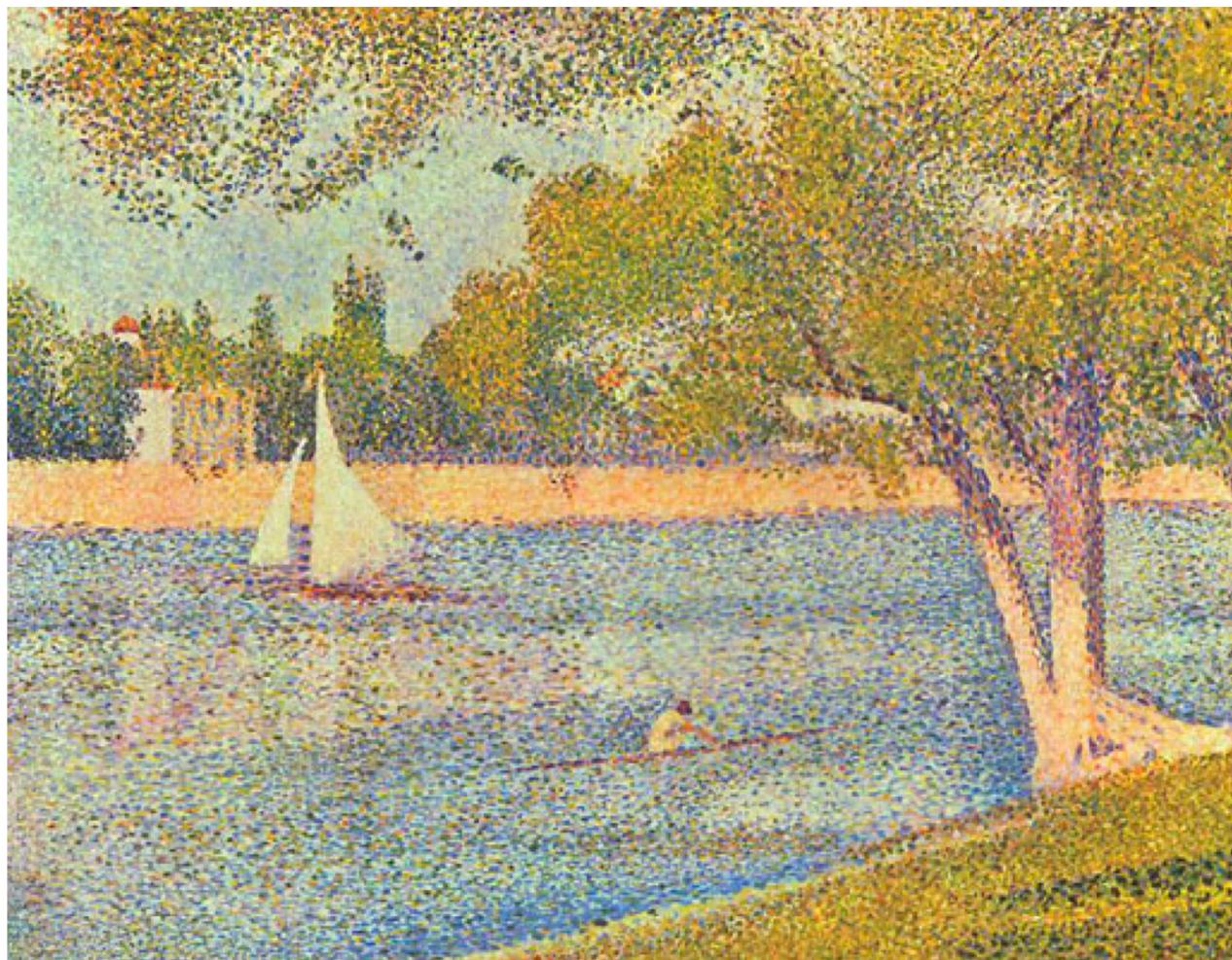
Creates an image  
from a matrix of  
pixels

Gives you the image  
as a matrix of ints  
(which you can edit)

```
GImage copy = new GImage(pixels);
```



Part two: Surat meets Instagram



Seurat: French post impressionist painter

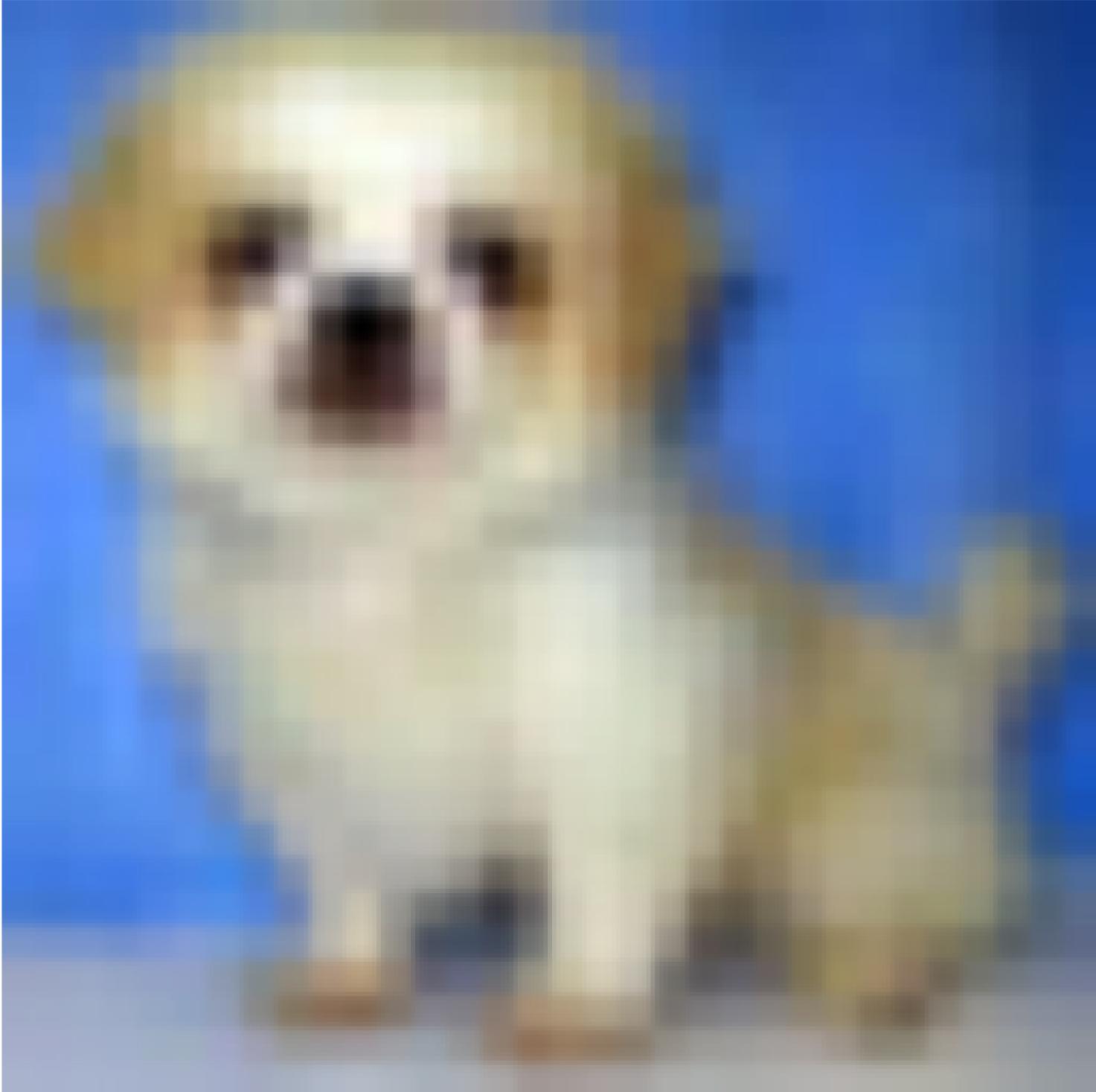


# Pointillism Filter

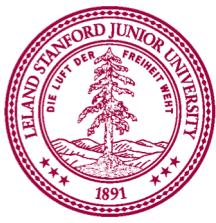
Repeat many times:

1. Pick a random pixel from an image.
2. Find the pixel's color
3. "Paint" a rather large brush stroke at a corresponding location, with the color



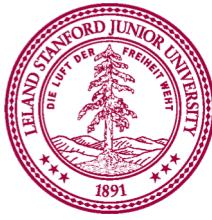


Palo Alto, CA 94304, Stanford University





Piech, CS106A, Stanford University



**c = 36**

**r = 24**



**c = 36**

**r = 24**



c = 21

r = 38



c = 21

r = 38

