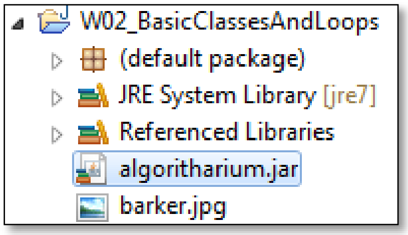
# Algoritharium 3

For this assignment, you will be completing the PaintableCanvas3.java file. There are several methods in that file that you will need to finish, but the assignment needs a little setup first.

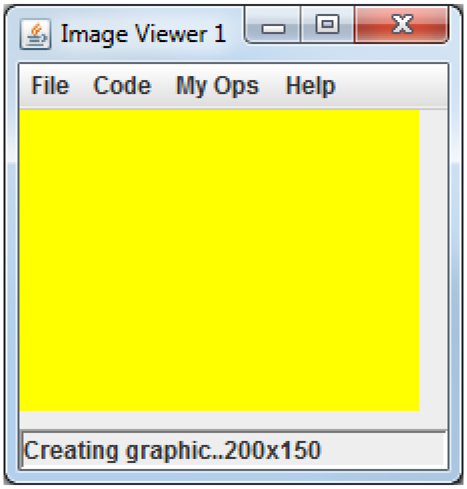
**Starting algoritharium.jar**

The algoritharium.jar file should still be where you used it in last week's eclipse folder. Find it and double click it to start the program.



**Load an image**

Although you can load in any image you want to work with, select File → New to load one that is all yellow.



Creating graphic..200x150

**Load the code**

Select Code → Load and open the PaintableCanvas3.class file found in your eclipse folder for this assignment. Note that eclipse must have already compiled this class for you, so be sure to open eclipse first.

# Assignment

This week you will not be coloring the picture, but rather looking through the picture for specific colors. We may not have had our talk about colors yet, but here is a brief introduction. Colors on the computer are made up of 3 components: Red, Green, and Blue. Generally, humans cannot distinguish more than 8 bits of color for each of these. So, each of these 3 components can have up to 8 bits. A bit of binary math with 8 bits gives 256 possible values. We always like to start at 0, so 0 is the lowest, and 255 is the highest.

The color red would be composed of 255 in the red, 0 in the green, and 0 in the blue, or

```
Color c = new Color(255,0,0);
```

Similarly, with green and blue.

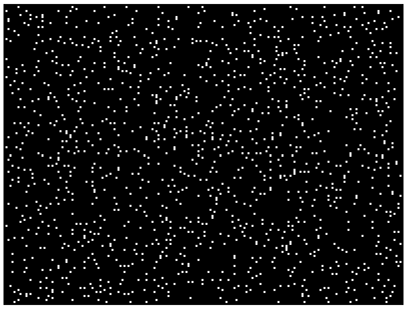
If you want to check the colors – like for this assignment – you would use the following code:

```
int red = c.getRed();
```

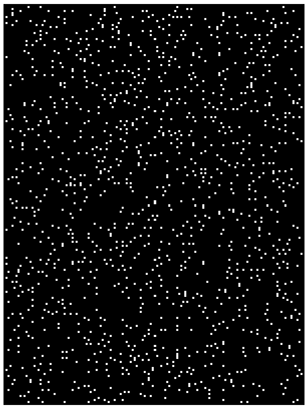
Similarly, for blue and green.

**One final note:** Having 255 in the red doesn't mean it will be pure red. Pure red is only achieved if there is no green and also no blue.

**howManyStars**
This method should look through the current picture and print how many pure white pixels there are. Pure white is 255 in all 3 color components



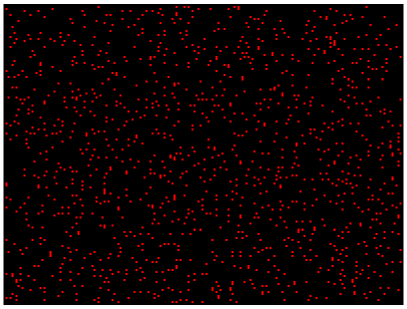
This one has 1304



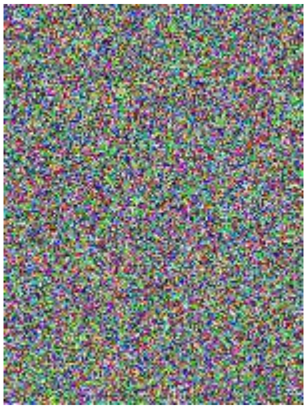
This one has 1285

**howManyPureRedPixels**

This method should look through the current picture and print how many pure red pixels there are. Pure red is 255 in the red component and 0 in the other two components



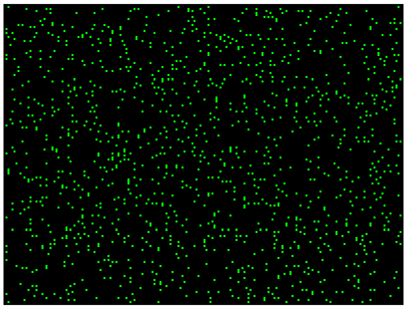
This one has 1282 red pixels



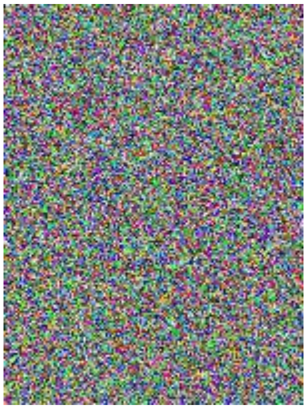
This one has 259

**howManyPureGreenPixels**

This method should look through the current picture and print how many pure green pixels there are. Pure green is 255 in the green component and 0 in the other two components



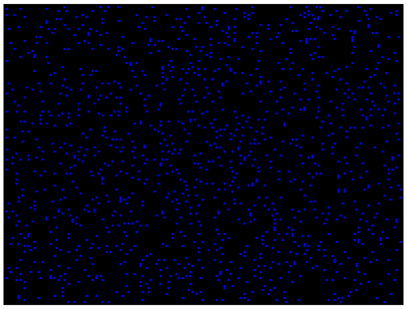
This one has 1277 green pixels



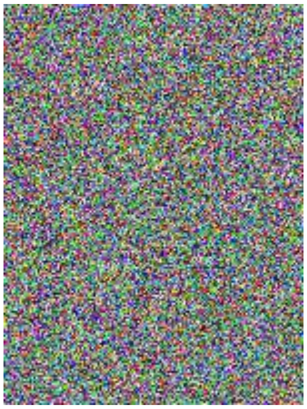
This one has 327 green pixels

**howManyPureBluePixels**

This method should look through the current picture and print how many pure blue pixels there are. Pure blue is 255 in the blue component and 0 in the other two components



This one has 1301, but is difficult to see



This one has 281

**howManyWithSpecificNumbers**

Now make this method work to find the color that has a red value of 58, a green value of 169, and a blue value of 31.

??????????

Put in your guess for all 5 methods in your PaintableCanvas3.java document as described in the next section.

**Submission:**

Put your answers for this assignment at the top of your PaintableCanvas3.java file commented as shown below:

//My guess using the unknown.jpg file for the number of White pixels is: 156

//My guess using the unknown.jpg file for the number of Red pixels is: 282

//My guess using the unknown.jpg file for the number of Green pixels is: 323

//My guess using the unknown.jpg file for the number of Blue pixels is: 308

//My guess using the unknown.jpg file for the specific color choice mentioned above: 877

(PaintableCanvas3 java code here)