Create a directory named **lab18** in your **labs** directory and place all code for this lab in your **lab18** directory.

Preface

A color can be defined by a red component, a green component and a blue component with each component having a value between 0 and 255. For example, the color red can be defined as (255, 0, 0) where the red component is the highest possible value (255) and the green and blue components are zero.

RGB.java
Write an entity class named **RGB** that models a color. The class should contain the following.

1. Three private fields that hold the color components.
2. A constructor that takes three values as arguments and initializes the corresponding fields.
3. Getters and setters for each of the fields.
4. A method named **toString** that overrides the **Object** class' **toString** method. The method should return a string containing the three components separated by commas. For example, 255,0,0.
5. A method named **equals** that overrides the **Object** class' **equals** method. Two instances of the **RGB** class are considered equal if all of their respective components are equal.

image.txt
Assume a file named **image.txt** exists and contains the rgb color values for each pixel of an image. Each pixel color is written on a separate line. The values of a color's components are written with commas between them.

Sample Input File

```
255,0,0
128,128,7
13,64,16
34,120,241
```

Note: The above sample input file is a sample. **image.txt** can contain any arbitrary number of pixel colors, not just 4 as shown above.

ColorApp.java
Write a class named **ColorApp** that satisfies the following.

1. The class contains a method named **printColors**. The method has 2 parameters. The first parameter is named **array** and holds a reference to an array of **RGB** objects. The method should print to the screen the color components of each RGB element in the array.

2. Write a method named **main** that does the following:
   - Allocate an array named **buffer** that can hold 256 **RGB** references.
   - Read the data in **image.txt** and store the data as **RGB** objects in the array.
   - Call **printColors** to print to the screen the components of all of the colors that are stored in the array.