```
import java.util.*;
import java.awt.image.BufferedImage;
import java.io.*;
import java.awt.Color;

/**************************************************************
Effects contains methods for editing pictures in the
steganography GUI.

@author Pranav Ramanan
@version 06/05/2014
**************************************************************/
public class Effects
{
  /**************************************************************
  Creates new image that is opposite color of old image
  **************************************************************/
   public static BufferedImage Inverse(BufferedImage img, int h, int w)
   {

      BufferedImage newImage = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);

      for(int y=0; y<h; y++)
      {
         for(int x=0; x<w; x++)
         {
            int rgb=img.getRGB(x, y);
            int r = (rgb)&0xFF;
            int g = (rgb>>8)&0xFF;
            int b = (rgb>>16)&0xFF;

            int invr= 255-r;
            int invg= 255-g;
            int invb= 255-b;

            int invColor = invr;
            invColor = (invColor << 8) + invg;
            invColor = (invColor << 8) + invb;
            newImage.setRGB(x, y, invColor);
         }
      }
      return newImage;
   }

  /**************************************************************
  Takes the image and slightly changes the color of the
  pixels to give it an opaque look.
  **************************************************************/
   public static BufferedImage Fade(BufferedImage img, int h, int w,int fade,String RGB)
   {

      BufferedImage newImage = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);
      for(int y=0; y<h; y++)
      {
```

```
        for(int x=0; x<w; x++)
        {
           int rgb=img.getRGB(x, y);
           Color c = new Color(rgb);
           Color tintColor = c;
            if(RGB.toUpperCase().equals("R"))
           {
               int boostRed = c.getRed()-fade;
               if(boostRed<0) boostRed=0;
               if(boostRed>255) boostRed=225;
              tintColor = new Color(boostRed,c.getGreen(),c.getBlue());
           }
           else if(RGB.toUpperCase().equals("G"))
           {
               int boostGreen = c.getGreen()-fade;
               if(boostGreen<0) boostGreen=0;
               if(boostGreen>225) boostGreen=225;
              tintColor = new Color(c.getRed(),boostGreen,c.getBlue());
           }
           else if(RGB.toUpperCase().equals("B"))
           {
               int boostBlue = c.getBlue()-fade;
               if(boostBlue<0) boostBlue=0;
               if(boostBlue>225) boostBlue=225;
              tintColor = new Color(c.getRed(),c.getGreen(),boostBlue);
           }
           else if(RGB.toUpperCase().equals("ALL"))
           {
               int boostRed = c.getRed()-fade;
               if(boostRed<0) boostRed=0;
               int boostGreen = c.getGreen()-fade;
               if(boostGreen<0) boostGreen=0;
               int boostBlue = c.getBlue()-fade;
               if(boostBlue<0) boostBlue=0;
               tintColor = new Color(boostRed,boostGreen,boostBlue);
           }
           int newColor = tintColor.getRGB();
           newImage.setRGB(x, y, newColor);
        }
    }
    return newImage;
 }

/*************************************************************
Takes the image and adds a color to each of the pixels
to tint the image.
*************************************************************/
 public static BufferedImage Tint(BufferedImage img, int h, int w, int tint, String RGB)
 {

    BufferedImage newImage = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);

    for(int y=0; y<h; y++)
    {
```

```
        for(int x=0; x<w; x++)
        {
           int rgb=img.getRGB(x, y);
           Color c = new Color(rgb);
           Color tintColor = c;
           if(RGB.toUpperCase().equals("R"))
           {
               int boostRed = c.getRed()+tint;
               if(boostRed>255) boostRed=225;
               if(boostRed<0) boostRed=0;
              tintColor = new Color(boostRed,c.getGreen(),c.getBlue());
           }
           else if(RGB.toUpperCase().equals("G"))
           {
               int boostGreen = c.getGreen()+tint;
               if(boostGreen>225) boostGreen=225;
               if(boostGreen<0) boostGreen=0;
              tintColor = new Color(c.getRed(),boostGreen,c.getBlue());
           }
           else if(RGB.toUpperCase().equals("B"))
           {
               int boostBlue = c.getBlue()+tint;
               if(boostBlue>225) boostBlue=225;
               if(boostBlue<0) boostBlue=0;
              tintColor = new Color(c.getRed(),c.getGreen(),boostBlue);
           }
           else if(RGB.toUpperCase().equals("ALL"))
           {
               int boostRed = c.getRed()+tint;
               if(boostRed>255) boostRed=225;
               int boostGreen = c.getGreen()+tint;
               if(boostGreen>225) boostGreen=225;
               int boostBlue = c.getBlue()+tint;
               if(boostBlue>225) boostBlue=225;
               tintColor = new Color(boostRed,boostGreen,boostBlue);
           }
           int newColor = tintColor.getRGB();
           newImage.setRGB(x, y, newColor);
        }
     }
     return newImage;
  }

 /*************************************************************
 Creates image that is only black and white.
 *************************************************************/
     public static BufferedImage BlackWhite(BufferedImage img, int h, int w)
  {
     BufferedImage newImage = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);
     int avg=0;
    for(int y=0; y<h; y++)
     {
        for(int x=0; x<w; x++)
        {
```

```
            int rgb=img.getRGB(x, y);
            int r = (rgb)&0xFF;
            int g = (rgb>>8)&0xFF;
            int b = (rgb>>16)&0xFF;
            avg+=(r+g+b)/3;
        }
    }
    avg=avg/(h*w);
    for(int y=0; y<h; y++)
    {
        for(int x=0; x<w; x++)
        {


            int rgb=img.getRGB(x, y);
            int r = (rgb)&0xFF;
            int g = (rgb>>8)&0xFF;
            int b = (rgb>>16)&0xFF;

            if (((r+g+b)/3)>avg)
            {
                r=225;
                g=225;
                b=225;
            }
            else
            {
                r=0;
                g=0;
                b=0;
            }

            int newColor = r;
            newColor = (newColor << 8) + g;
            newColor = (newColor << 8) + b;

            newImage.setRGB(x, y, newColor);
        }
    }
    return newImage;
}
    /*************************************************************
    Creates image that is only black and white.
    *************************************************************/
    public static BufferedImage Grayscale(BufferedImage img, int h, int w)
{

    BufferedImage newImage = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);

    for(int y=0; y<h; y++)
    {
        for(int x=0; x<w; x++)
        {
```

```
            int rgb=img.getRGB(x, y);
            int r = (rgb)&0xFF;
            int g = (rgb>>8)&0xFF;
            int b = (rgb>>16)&0xFF;

            int combColor = (r+b+g)/3;

            int newColor = combColor;
            newColor = (newColor << 8) + combColor;
            newColor = (newColor << 8) + combColor;

            newImage.setRGB(x, y, newColor);
         }
      }
      return newImage;
   }
  /*************************************************************
  Takes the image removes all the red, green, or blue.
  *************************************************************/
   public static BufferedImage Remove(BufferedImage img, int h, int w, String RGB)
   {

      BufferedImage newImage = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);

      for(int y=0; y<h; y++)
      {
         for(int x=0; x<w; x++)
         {

            int rgb=img.getRGB(x, y);
            Color c = new Color(rgb);
            Color tintColor = c;
            if(RGB.toUpperCase().equals("R"))
            {
               tintColor = new Color(000,c.getGreen(),c.getBlue());
            }
            else if(RGB.toUpperCase().equals("G"))
            {
               tintColor = new Color(c.getRed(),000,c.getBlue());
            }
            else if(RGB.toUpperCase().equals("B"))
            {
               tintColor = new Color(c.getRed(),c.getGreen(),000);
            }
            int newColor = tintColor.getRGB();
            newImage.setRGB(x, y, newColor);
         }
      }
      return newImage;
   }

  /*************************************************************
  Creates new image that is opposite color of old image
```

```
*************************************************************/
 public static BufferedImage Inverse(BufferedImage img, int h, int w, String BD)
 {

    BufferedImage newImage = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);

    for(int y=0; y<h; y++)
    {
       for(int x=0; x<w; x++)
       {


          int rgb=img.getRGB(x, y);
          int r = (rgb)&0xFF;
          int g = (rgb>>8)&0xFF;
          int b = (rgb>>16)&0xFF;

          if (BD=="B")
          {
             int brightR=r+10;
             int brightG=g+10;
             int brightB=b+10;

             int Bcolor = brightR;
             Bcolor = (Bcolor << 8) + brightG;
             Bcolor = (Bcolor << 8) + brightB;

             newImage.setRGB(h, w, Bcolor);
          }

          else
          {
             int darkR=r-10;
             int darkG=g-10;
             int darkB=b-10;

             int Bcolor = darkR;
             Bcolor = (Bcolor << 8) + darkG;
             Bcolor = (Bcolor << 8) + darkB;

             newImage.setRGB(x, y, Bcolor);
          }


       }
    }
    return newImage;
 }

/*************************************************************
Creates new image with wildly changed colors.
*************************************************************/
 public static BufferedImage Colorize(BufferedImage img, int h, int w)
 {
```

```
      BufferedImage newImage = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);
      for(int y=0; y<h; y++)
      {
         for(int x=0; x<w; x++)
         {
            int rgb=img.getRGB(x, y);
            int r = (rgb)&0xFF;
            int g = (rgb>>8)&0xFF;
            int b = (rgb>>16)&0xFF;

            int newR= ((0-r)*-1)/2+r;
            int newG= ((0-g)*-1)/2+g;
            int newB= ((0-b)*-1)/2+b;

            int fadeColor = newR;
            fadeColor = (fadeColor << 8) + newG;
            fadeColor = (fadeColor << 8) + newB;

            newImage.setRGB(x, y, fadeColor);

         }
      }
      return newImage;
   }
}
```