```java
// Tobias Woegerbuaer; Matrikelnr: 23347320
// Simon Muschik; Matrikelnr: 23336058

import java.sql.SQLOutput;

public class Color {

    private int rgb;            //2. Attribut rgb

    public static final Color BLACK = new Color(0);                            //12.haeufig verwendete Farben
    public static final Color GRAY = new Color(16777215);
    public static final Color GREEN = new Color(65280);
    public static final Color WHITE = new Color(16777215);
    public static final Color RED = new Color(16711680);
    public static final Color BLUE = new Color(255);


    public Color(int rgb) {             //3.1 Konstruktor, speichert uebergebenen wWert in rgb
        this.rgb = rgb;
    }

    public Color(int red, int green, int blue) {                //3.2

        int[] colorsRGB = {red, green, blue};                  //Array fuer RGB Zahlen
        String[] colorNames = {"red", "green", "blue"};         //Array Namen der Farben

        for (int i = 0; i < 3; i++) {                              //Faengt >255 ab + Fehlermeldung + Korrektur
            if (colorsRGB[i] > 255) {
                System.err.println("Error! " + colorNames[i] + " to high! Number " + colorsRGB[i]
                        + " got corrected to 255");
                colorsRGB[i] = 255;

            } else if (colorsRGB[i] < 0) {                          //Faengt <0 ab + Fehlermeldung + Korrektur
                System.err.println("Error! " + colorNames[i] + " must be >=0! Number " + colorsRGB[i]
                        + " got corrected to 0");
                colorsRGB[i] = 0;
            }
        }
        red = colorsRGB[0] << 16;
        green = colorsRGB[1] << 8;
        blue = colorsRGB[2];
        rgb = (red | green | blue);               //Speicherung von rot, gruen, blau in rgb
    }

    public Color() {                                //3.3 Konstrunktor, rgb wird schwarz zugewiesen
        rgb = 0;
    }

    public int getRgb() {                   //4. get-Methode Rgb
        return rgb;
    }

    public int getRed() {              //6. rot aus Rgb auslesen
        int red = rgb >> 16;
        return red;
    }

    public int getGreen() {             //(6)gruen aus Rgb auslesen
        int green = (rgb >> 8) & 255;
        return green;
    }

    public int getBlue() {                 //(6) blau aus Rgb auslesen
        int blue = rgb & 255;
        return blue;
    }

    public String getHex() {               //7.Umwandlung von rgb ins Hexadezimalsysthem als String
        String rgbHex = Integer.toHexString(rgb).toUpperCase();

        while (rgbHex.length() < 6) {
            rgbHex = "0" + rgbHex;
        }
        return "#" + rgbHex;
    }

    public Color(String rgbHex) {                             //9. hex - # in 10er Systhem
        rgbHex = rgbHex.substring(1);
        rgb = Integer.parseInt(rgbHex, 16);
    }

    @Override
    public String toString() {                                  //10. toString-Methode ueberschreiben
        return getHex();
    }

    public Color complementaryColor() {                  //11a. Komplementaerfarbe

        Color complementaryColor = new Color(Math.abs(getRed() - 255), Math.abs(getGreen() - 255), Math.abs(getBlue() - 255));
        return complementaryColor;
    }

    public Color mixColor(Color color) {                        //11b. Farben mischen

        int redNew = (color.getRed() + getRed()) / 2;             //Berechnung neue Einzelfarben
        int greenNew = (color.getGreen() + getGreen()) / 2;
        int blueNew = (color.getBlue() + getBlue()) / 2;

        Color mixedColor = new Color(redNew, greenNew, blueNew);                //neue, gemischte Farbe

        return mixedColor;
    }


    public static void main(String[] args) {

        Color color1 = new Color("#663399");                //Verschiedene Farben zum testen
        Color color2 = new Color("#FFE4C4");
        Color color3 = new Color("#008080");
        Color color4 = new Color("#6633990");

        ColorVisualizer colorVisualizer = new ColorVisualizer(color2);          //Visualizer

        //Testen mixColor:
        ColorVisualizer mixColor = new ColorVisualizer(color2.mixColor(GREEN));

        //Testen complementaryColor:
        ColorVisualizer complementaryColor = new ColorVisualizer(color2.complementaryColor());

    }
}
```