
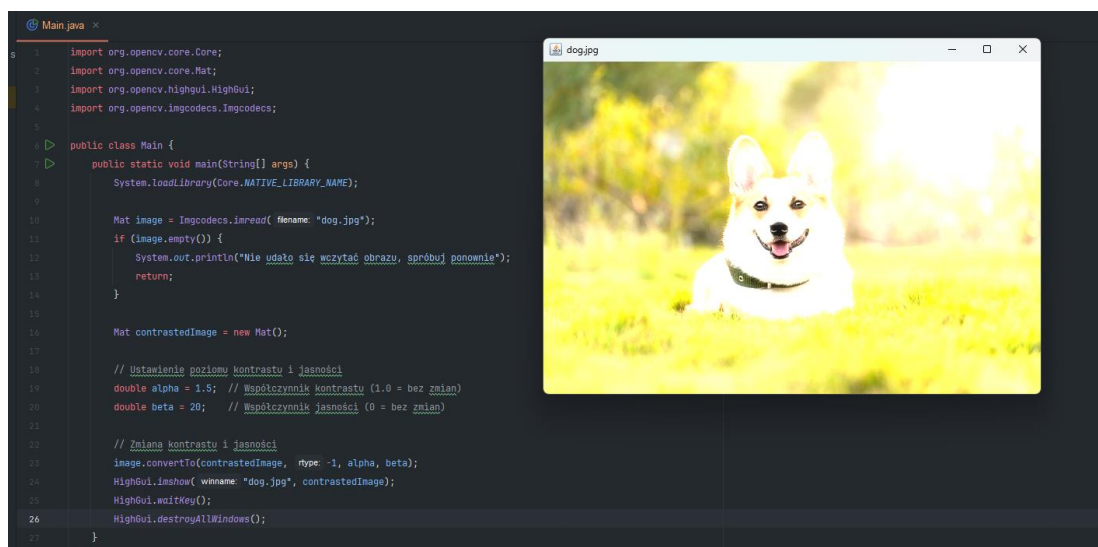
	Politechnika Bydgoska im. J. J. Śniadeckich Wydział Telekomunikacji, Informatyki i Elektrotechniki <b>Zakład Systemów Teleinformatycznych</b>			
<b>Przedmiot</b>	Przetwarzanie obrazów			
<b>Prowadzący</b>	mgr inż. Grzegorz Czczot			
<b>Temat</b>	Operacje geometryczne			
<b>Student</b>	Paweł Jońca			
<b>Nr lab.</b>	3	<b>Data wykonania</b>	28.10.2024r	
<b>Ocena</b>		<b>Data oddania spr.</b>	28.10.2024r	

## Zad 1

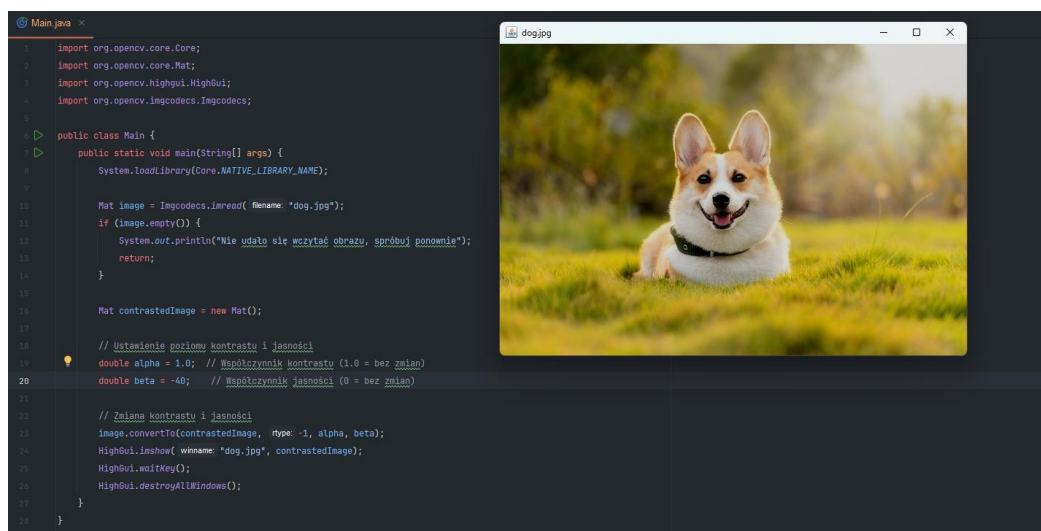


```

1  import org.opencv.core.Core;
2  import org.opencv.core.Mat;
3  import org.opencv.highgui.HighGui;
4  import org.opencv.imgcodecs.Imgcodecs;
5
6  public class Main {
7      public static void main(String[] args) {
8          System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
9
10         Mat image = Imgcodecs.imread( "name: \"dog.jpg\"");
11         if (image.empty()) {
12             System.out.println("Nie udało się wczytać obrazu, spróbuj ponownie");
13             return;
14         }
15
16         Mat contrastedImage = new Mat();
17
18         // Ustawienie poziomu kontrastu i jasności
19         double alpha = 1.5; // Współczynnik kontrastu (1.0 = bez zmian)
20         double beta = 20; // Współczynnik jasności (0 = bez zmian)
21
22         // Zmiana kontrastu i jasności
23         image.convertTo(contrastedImage, -1, alpha, beta);
24         HighGui.imshow( "winname: \"dog.jpg\", contrastedImage");
25         HighGui.waitKey();
26         HighGui.destroyAllWindows();
27     }
28 }

```

## Zad 2

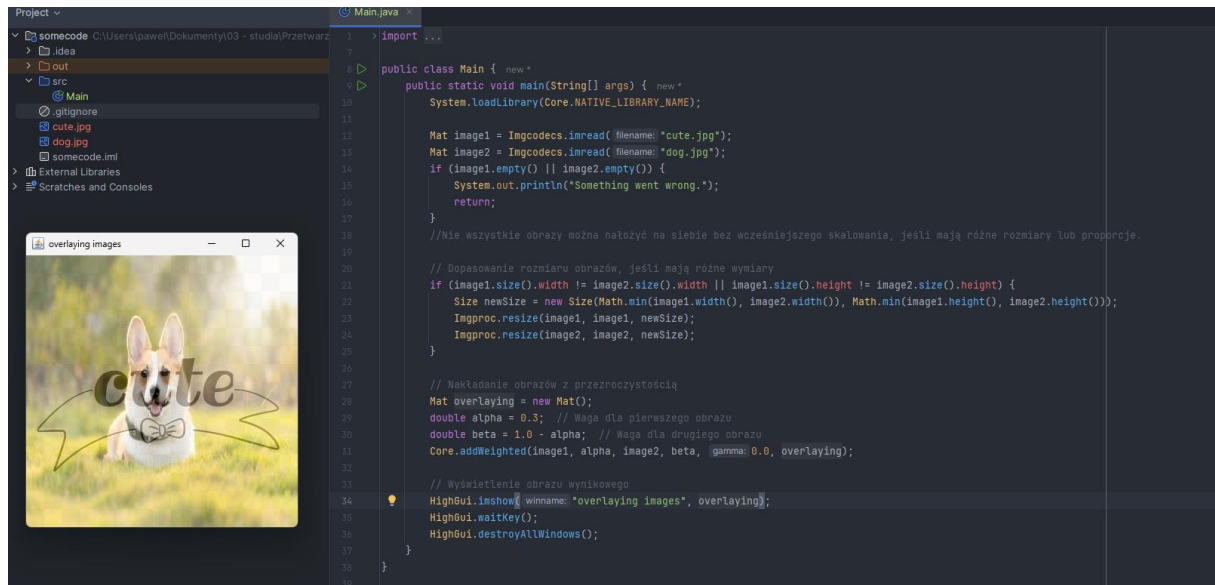


```

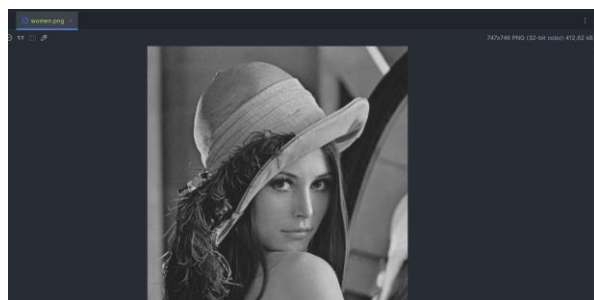
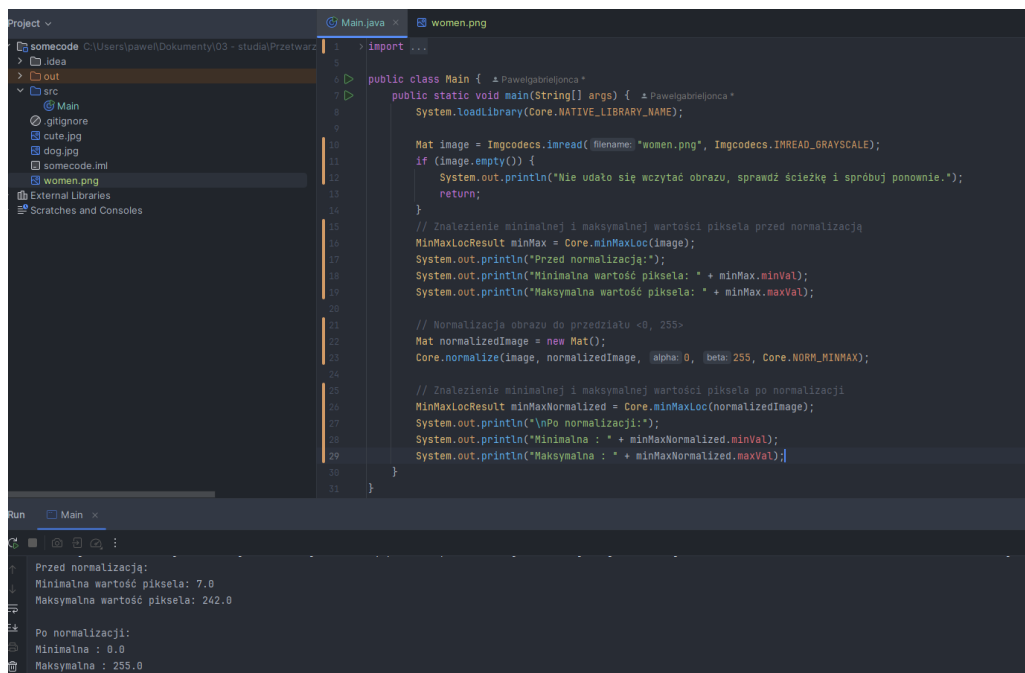
1  import org.opencv.core.Core;
2  import org.opencv.core.Mat;
3  import org.opencv.highgui.HighGui;
4  import org.opencv.imgcodecs.Imgcodecs;
5
6  public class Main {
7      public static void main(String[] args) {
8          System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
9
10         Mat image = Imgcodecs.imread( "name: \"dog.jpg\"");
11         if (image.empty()) {
12             System.out.println("Nie udało się wczytać obrazu, spróbuj ponownie");
13             return;
14         }
15
16         Mat contrastedImage = new Mat();
17
18         // Ustawienie poziomu kontrastu i jasności
19         double alpha = 1.0; // Współczynnik kontrastu (1.0 = bez zmian)
20         double beta = -40; // Współczynnik jasności (0 = bez zmian)
21
22         // Zmiana kontrastu i jasności
23         image.convertTo(contrastedImage, -1, alpha, beta);
24         HighGui.imshow( "winname: \"dog.jpg\", contrastedImage");
25         HighGui.waitKey();
26         HighGui.destroyAllWindows();
27     }
28 }

```

### Zad 3

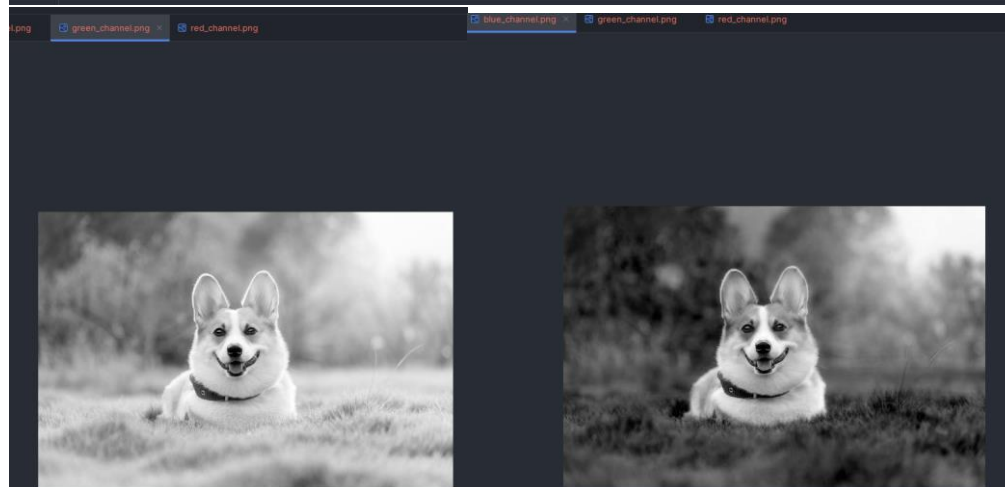


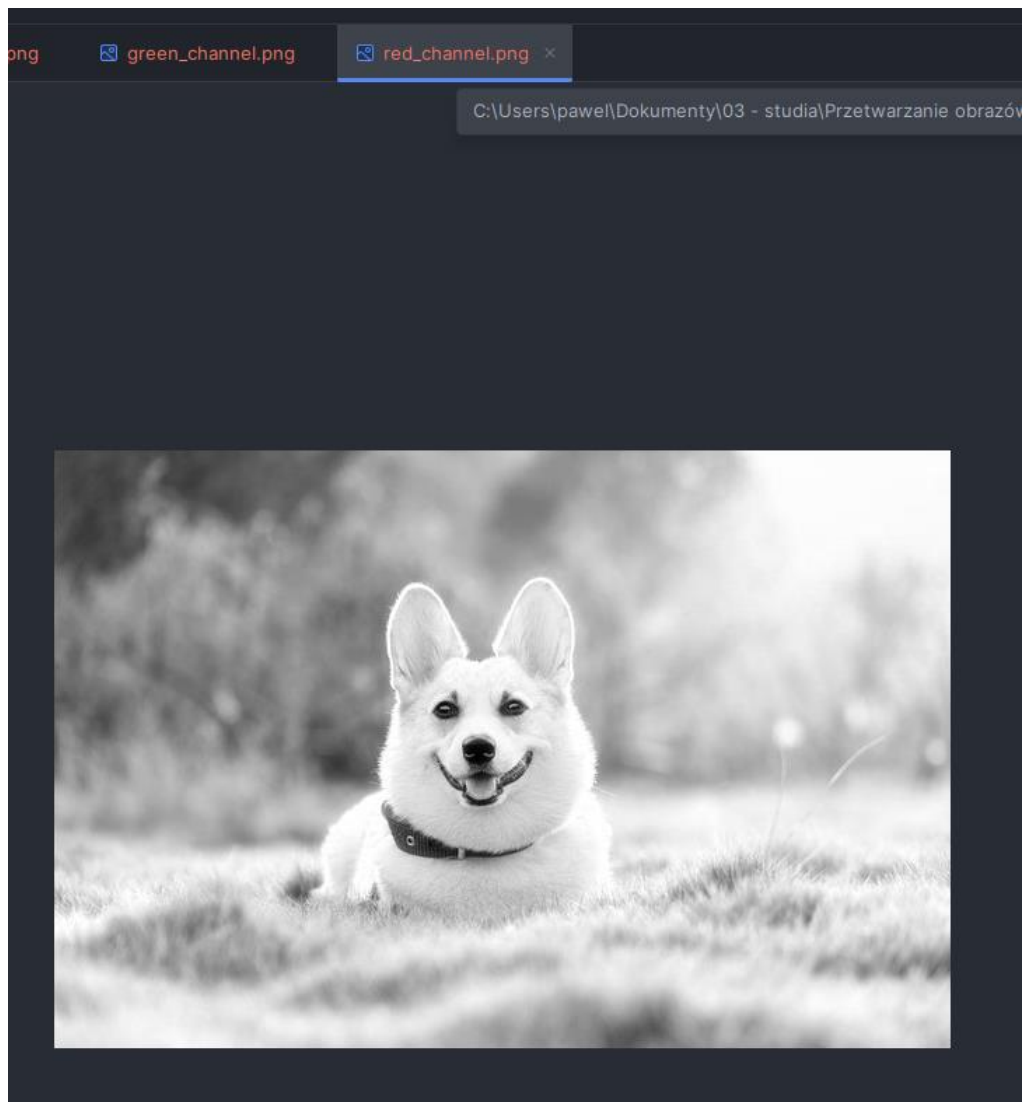
### Zad 4



## Zad 5

```
1 import org.opencv.core.*;
2 import org.opencv.imgcodecs.Imgcodecs;
3 import org.opencv.core.Core;
4
5 public class Main {
6     public static void main(String[] args) {
7         System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
8         Mat image = Imgcodecs.imread("dog.jpg");
9         if (image.empty()) {
10             System.out.println("Failed to load dog.png.");
11             return;
12         }
13
14         // Rozdzielenie obrazu na trzy kanały: B, G, R
15         java.util.List<Mat> channels = new java.util.ArrayList<>();
16         Core.split(image, channels);
17
18         // Tworzenie matryc dla poszczególnych składowych
19         Mat blueChannel = new Mat();
20         Mat greenChannel = new Mat();
21         Mat redChannel = new Mat();
22
23         // Inicjalizacja matryc o takich samych wymiarach jak oryginalny obraz, ale z zerowymi wartościami dla dwóch kanałów
24         blueChannel = channels.get(0);
25         greenChannel = channels.get(1);
26         redChannel = channels.get(2);
27
28         Imgcodecs.imwrite("blue_channel.png", blueChannel);
29         Imgcodecs.imwrite("green_channel.png", greenChannel);
30         Imgcodecs.imwrite("red_channel.png", redChannel);
31
32         System.out.println("Obrazy zapisane.");
33     }
34 }
35 }
```





Na obrazie który składa się tylko z czerwonych pikseli część elementów zanika ponieważ piksele, które nie zawierają czerwonego znikają, bo ich wartość niebieska i zielona jest ustawiona na zero. Dlatego na każdym z obrazów widać tylko te elementy które były związane z daną składową kolorystyczną.

## Zad 6



Różnice: Przestrzeń HSV oddziela informacje o kolorze, nasyceniu i jasności, co sprawia, że obraz wygląda bardziej nietypowo i nie jest intuicyjnie zrozumiały bez odpowiedniego przetwarzania. Zapisany obraz się różni kolorystycznie co widać na pierwszy rzut oka.

HSV to przestrzeń kolorów, która reprezentuje kolory w sposób bardziej intuicyjny dla ludzkiego postrzegania. H(Hue) odcień, określający kolor np. czerwony, niebieski, zielony wyrażany w stopniach 0- 360

S(Saturation) Nasycenie, które opisuje intensywność koloru. Wartość 0 oznacza brak koloru (szarość) a 255 pełne nasycenie

V(Value) Wartość, która opisuje jasność koloru 0 to czerń a 255 to maksymalna jasność



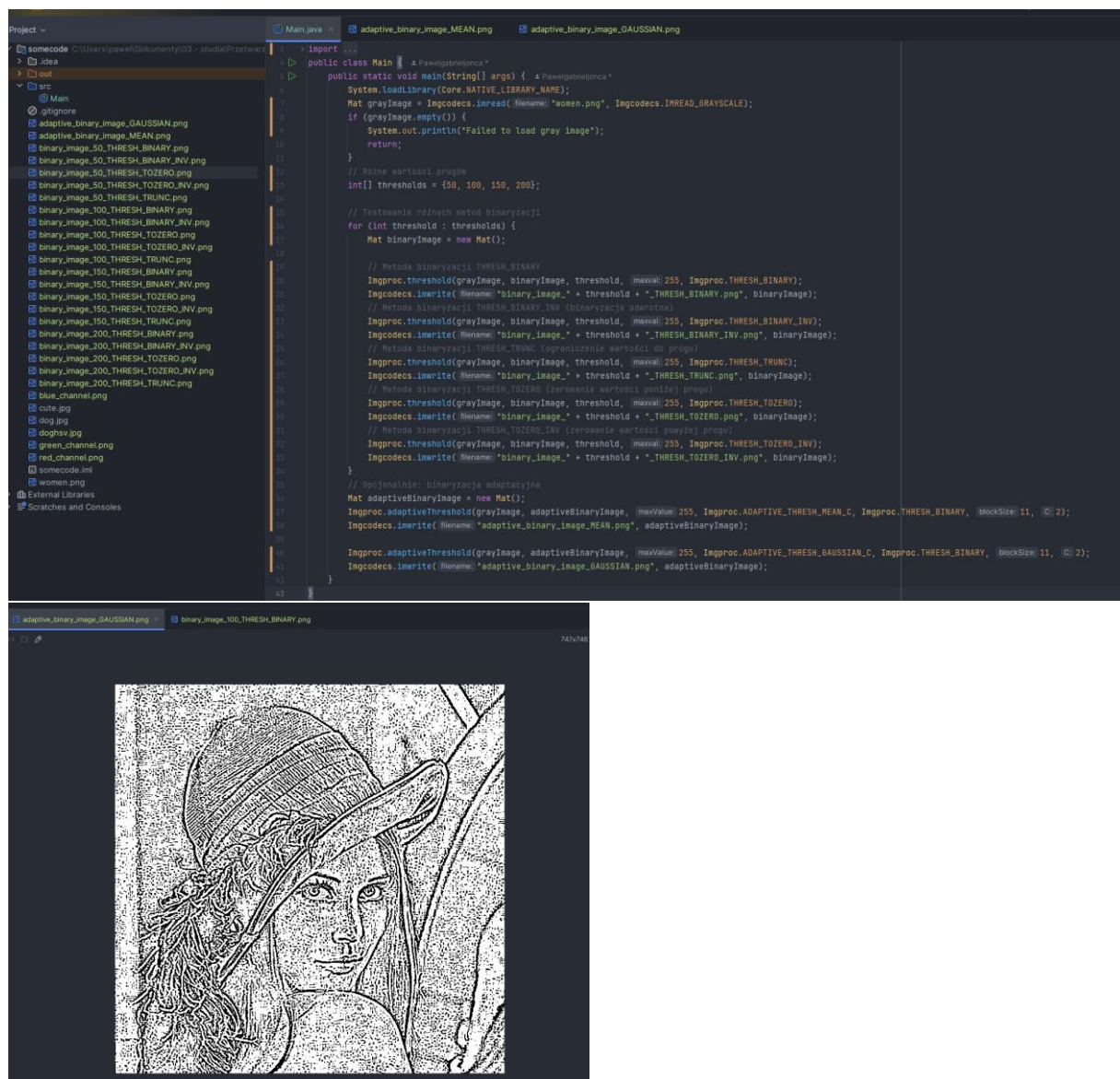
Konwersje między przestrzeniami RGB i HSV w OpenCV jest możliwa dzięki odpowiednim algorytmom matematycznym, które przeliczają wartości kanałów RGB na odpowiednie komponenty HSV.

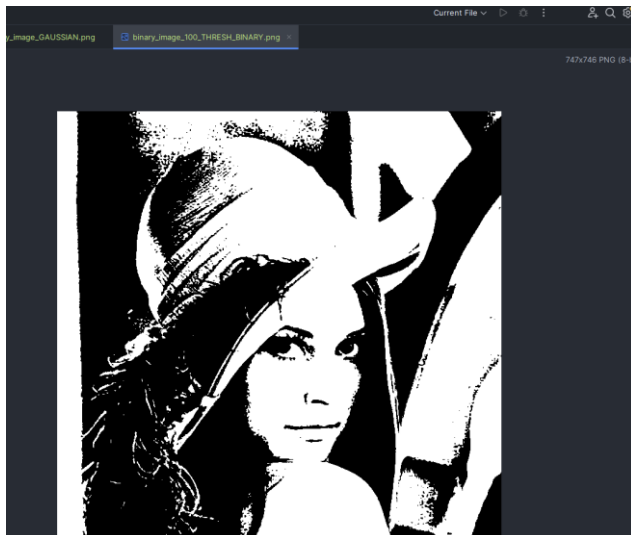
W OpenCV konwersji tej dokonujemy za pomocą funkcji:

`Imgproc.cvtColor(mat, mat, Imgproc.COLOR_BGR2HSV)` – z RGB (BGR) do HSV.

`Imgproc.cvtColor(mat, mat, Imgproc.COLOR_HSV2BGR)` – z HSV do RGB (BGR)

## Zad 7

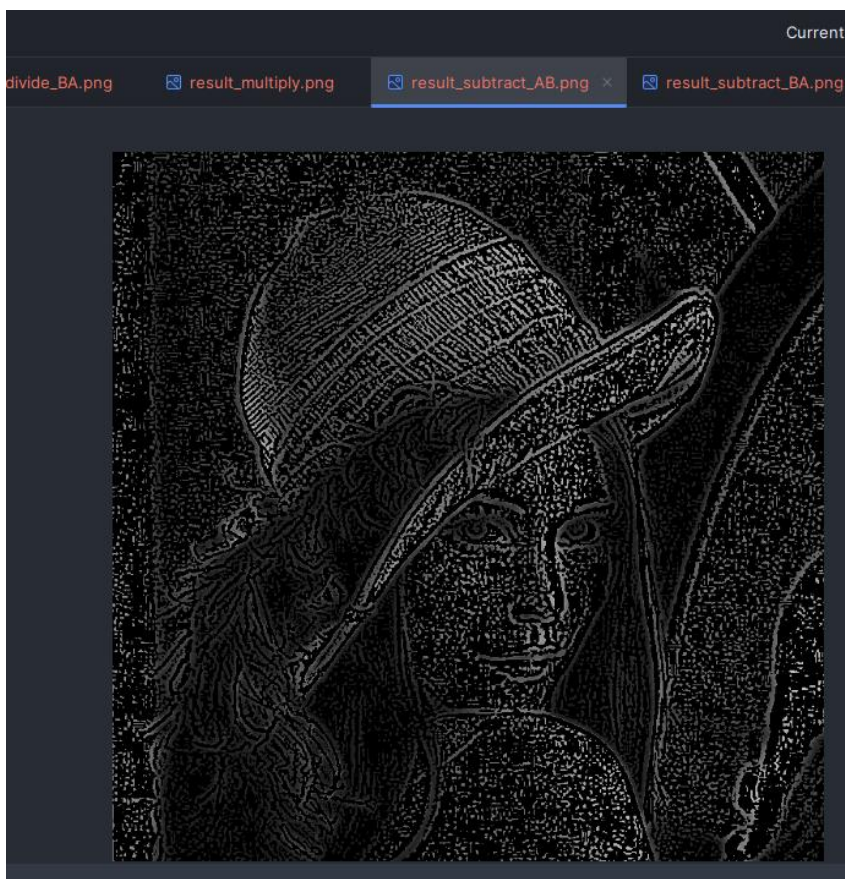




Dwa zdjęcia dla przykładu, że wszystko działa, bo jest ich więcej.

Zad 8

Aby było możliwe wykonanie operacji na obrazach muszą one mieć ten sam rozmiar oraz mieć tę samą liczbę kanałów np. być w kolorze RGB lub w skali szarości



```
1 > import ...
5 public class Main {
6     public static void main(String[] args) {
7         System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
8         Mat imageA = Imgcodecs.imread("women.png");
9         Mat imageB = Imgcodecs.imread("adaptive_binary_image_GAUSSIAN.png");
10        if (imageA.empty() || imageB.empty()) {
11            System.out.println("Failed to load images");
12            return;
13        }
14        // Dopasowanie rozmiaru obrazów, jeśli mają różne wymiary
15        if (imageA.size().width != imageB.size().width || imageA.size().height != imageB.size().height) {
16            Size newSize = new Size(Math.min(imageA.width(), imageB.width()), Math.min(imageA.height(), imageB.height()));
17            Imgproc.resize(imageA, imageA, newSize);
18            Imgproc.resize(imageB, imageB, newSize);
19        }
20        // Operacja A + B
21        Mat resultAdd = new Mat();
22        Core.add(imageA, imageB, resultAdd);
23        Imgcodecs.imwrite("result_add.png", resultAdd);
24        // Operacja A - B
25        Mat resultSubtractAB = new Mat();
26        Core.subtract(imageA, imageB, resultSubtractAB);
27        Imgcodecs.imwrite("result_subtract_AB.png", resultSubtractAB);
28        // Operacja B - A
29        Mat resultSubtractBA = new Mat();
30        Core.subtract(imageB, imageA, resultSubtractBA);
31        Imgcodecs.imwrite("result_subtract_BA.png", resultSubtractBA);
32        // Operacja A * B
33        Mat resultMultiply = new Mat();
34        Core.multiply(imageA, imageB, resultMultiply);
35        Imgcodecs.imwrite("result_multiply.png", resultMultiply);
36        // Operacja A / B
37        Mat resultDivideAB = new Mat();
38        Core.divide(imageA, imageB, resultDivideAB);
39        Imgcodecs.imwrite("result_divide_AB.png", resultDivideAB);
40        // Operacja B / A
41        Mat resultDivideBA = new Mat();
42        Core.divide(imageB, imageA, resultDivideBA);
43        Imgcodecs.imwrite("result_divide_BA.png", resultDivideBA);
44    }
45 }
46
```

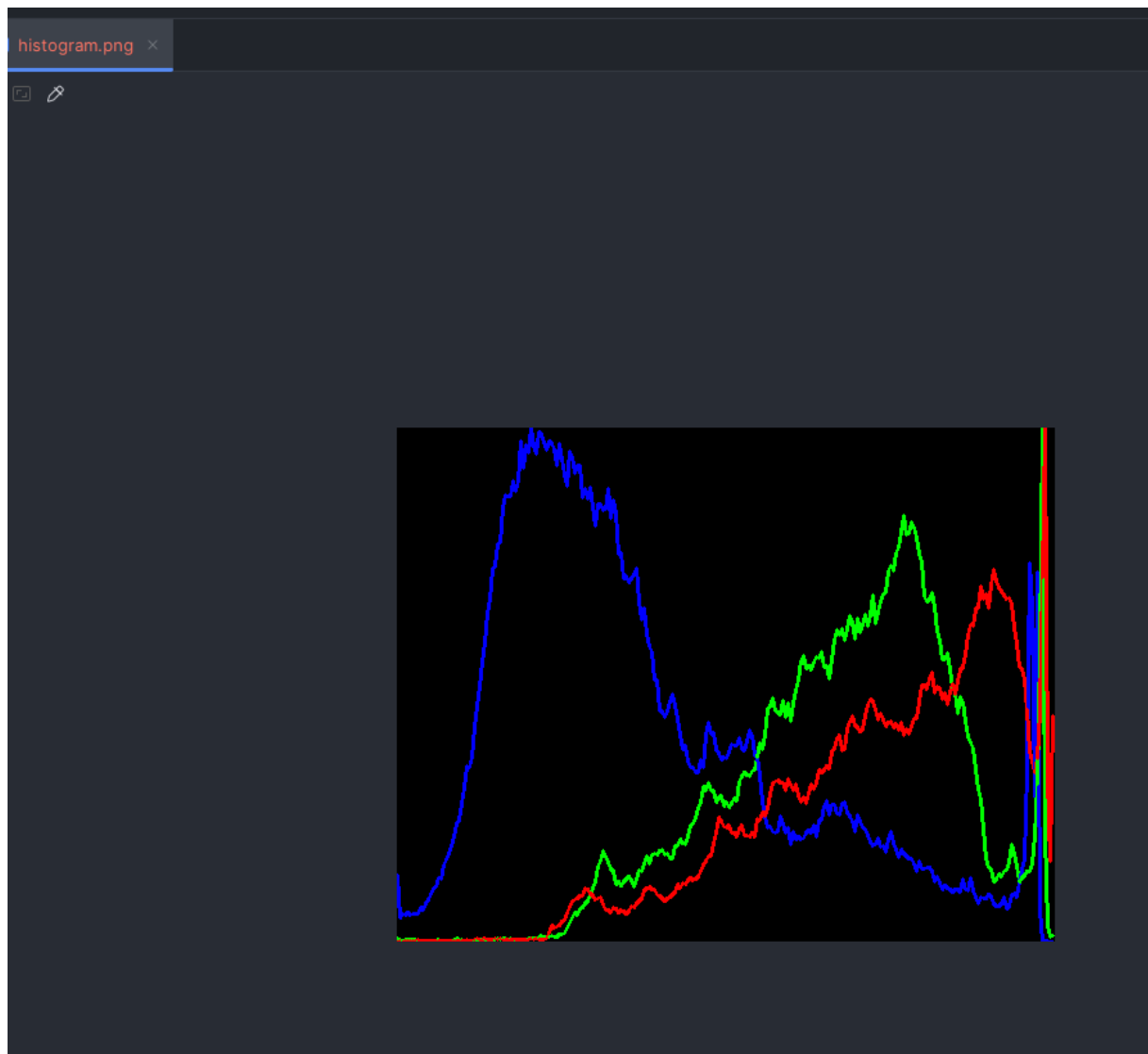
Zad 9



```

1  > import ...
2  public class Main { 1 Pawelgabrieljonca *
3  public static void main(String[] args) { 1 Pawelgabrieljonca *
4      System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
5      Mat image = Imgcodecs.imread(filename: "dog.jpg");
6      if (image.empty()) {
7          System.out.println("Failed to load image");
8          return;
9      }
10     List<Mat> bgrPlanes = new ArrayList<>(); // Rozdziel obraz na trzy kanały: B, G, R
11     Core.split(image, bgrPlanes);
12     int histSize = 256; // 256 przedziałów intensywności pikseli
13     MatOfInt histSizeMat = new MatOfInt(histSize);
14     MatOfFloat histRange = new MatOfFloat(1.0f, 0, 256); // Zakres intensywności (0 do 255)
15     boolean accumulate = false;
16     // Histogramy dla każdego z kanałów
17     Mat bHist = new Mat();
18     Mat gHist = new Mat();
19     Mat rHist = new Mat();
20     // Oblicz histogram dla kanałów B, G, R
21     Imgproc.calcHist(Collections.singletonList(bgrPlanes.get(0)), new MatOfInt(histSizeMat), new Mat(), bHist, histSizeMat, histRange, accumulate);
22     Imgproc.calcHist(Collections.singletonList(bgrPlanes.get(1)), new MatOfInt(histSizeMat), new Mat(), gHist, histSizeMat, histRange, accumulate);
23     Imgproc.calcHist(Collections.singletonList(bgrPlanes.get(2)), new MatOfInt(histSizeMat), new Mat(), rHist, histSizeMat, histRange, accumulate);
24     // Ustawienia wykresu histogramu
25     int histW = 512; // Szerokość wykresu
26     int histH = 400; // Wysokość wykresu
27     int binWidth = (int) Math.round((double) histW / histSize);
28     Mat histImage = new Mat(histH, histW, CvType.CV_8UC3, new Scalar(0, 0, 0));
29     // Normalizacja histogramu do rozmiarów wykresu
30     Core.normalize(bHist, bHist, 1.0f, histImage.rows(), Core.NORM_MINMAX);
31     Core.normalize(gHist, gHist, 1.0f, histImage.rows(), Core.NORM_MINMAX);
32     Core.normalize(rHist, rHist, 1.0f, histImage.rows(), Core.NORM_MINMAX);
33     // Rysowanie histogramu dla każdego kanału
34     for (int i = 1; i < histSize; i++) {
35         // Niebieski kanał
36         Imgproc.line(
37             histImage,
38             new Point(x: binWidth * (i - 1), y: histH - Math.round(bHist.get(row: i - 1, col: 0)[0])),
39             new Point(x: binWidth * i, y: histH - Math.round(bHist.get(i, col: 0)[0])),
40             new Scalar(255, 0, 0), thickness: 2, lineType: 8, shift: 0);
41         // Zielony kanał
42         Imgproc.line(
43             histImage,
44             new Point(x: binWidth * (i - 1), y: histH - Math.round(gHist.get(row: i - 1, col: 0)[0])),
45             new Point(x: binWidth * i, y: histH - Math.round(gHist.get(i, col: 0)[0])),
46             new Scalar(0, 255, 0), thickness: 2, lineType: 8, shift: 0);
47         // Czerwony kanał
48         Imgproc.line(
49             histImage,
50             new Point(x: binWidth * (i - 1), y: histH - Math.round(rHist.get(row: i - 1, col: 0)[0])),
51             new Point(x: binWidth * i, y: histH - Math.round(rHist.get(i, col: 0)[0])),
52             new Scalar(0, 0, 255), thickness: 2, lineType: 8, shift: 0);
53     }
54     Imgcodecs.imwrite(filename: "histogram.png", histImage);
55     System.out.println("Histogram zapisany jako histogram.png");
56 }

```



## Wnioski

Zadania pokazały nam możliwość jakie oferuje biblioteka openCV. Mogliśmy poznać jak za pomocą kodu można zmieniać kolory zdjęć, nakładać je na siebie oraz poznać nową przestrzeń HSV. To wszystko na praktycznych przykładach dzięki czemu wiedzieliśmy efekt naszej pracy.