
	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 morfologiczne		
<b>Student</b>	Paweł Jońca		
<b>Nr lab.</b>	4	<b>Data wykonania</b>	31.10
<b>Ocena</b>		<b>Data oddania spr.</b>	31.10

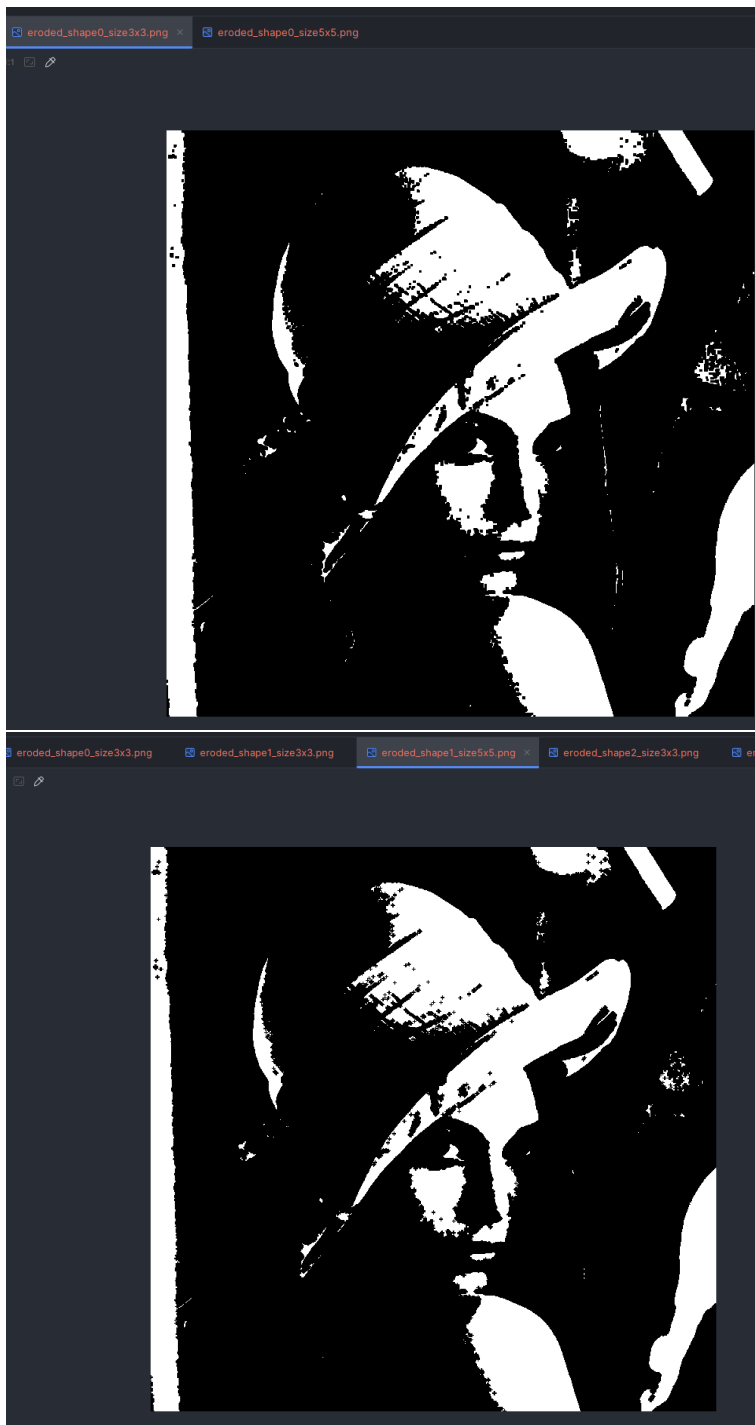
## Zad 1

```

Main.java x
1  import org.opencv.core.*;
2  import org.opencv.imgcodecs.Imgcodecs;
3  import org.opencv.imgproc.Imgproc;
4
5  public class Main {
6      static { System.loadLibrary(Core.NATIVE_LIBRARY_NAME); }
7
8      public static void main(String[] args) {
9          Mat img = Imgcodecs.imread("women.png", Imgcodecs.IMREAD_GRAYSCALE);
10
11         if (img.empty()) {
12             System.out.println("Failed to load image.");
13             return;
14         }
15
16         // Binarizacja obrazu
17         Mat binaryImg = new Mat();
18         Imgproc.threshold(img, binaryImg, 127, 255, Imgproc.THRESH_BINARY);
19
20         // Definicje rozmiarów i kształtów elementów strukturalnych
21         Size[] sizes = {new Size(3, 3), new Size(5, 5)};
22         int[] shapes = {Imgproc.MORPH_RECT, Imgproc.MORPH_ELLIPSE, Imgproc.MORPH_CROSS};
23
24         // Wykonaj erozję dla różnych elementów strukturalnych i zapisz wyniki
25         int index = 0;
26         for (Size size : sizes) {
27             for (int shape : shapes) {
28                 Mat element = Imgproc.getStructuringElement(shape, size);
29                 Mat erodedImg = new Mat();
30                 Imgproc.erode(binaryImg, erodedImg, element);
31
32                 // Zapisanie wyniku do pliku
33                 String outputFilename = String.format("eroded_shape%d_size%.0f%.0f.png", shape, size.width, size.height);
34                 Imgcodecs.imwrite(outputFilename, erodedImg);
35                 System.out.println("Zapisano: " + outputFilename);
36
37                 index++;
38             }
39         }
40     }
41 }

```

2 przykładowe zdjęcia z 6

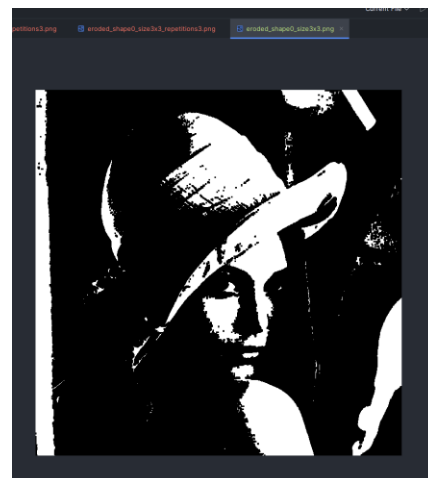
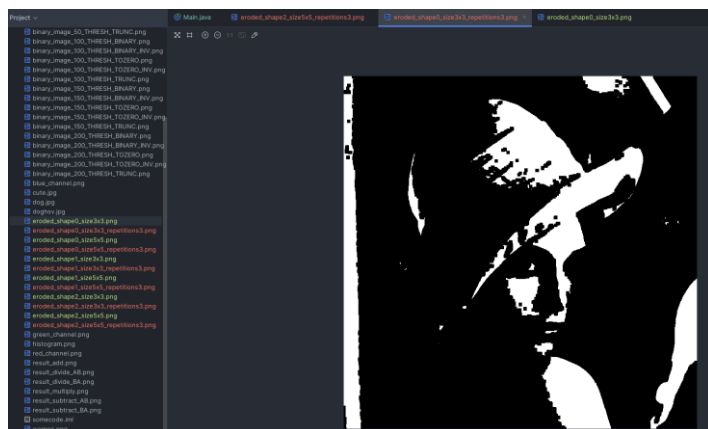


Zad 2.

```

1  import org.opencv.core.*;
2  import org.opencv.imgcodecs.Imgcodecs;
3  import org.opencv.imgproc.Imgproc;
4
5  public class Main {
6      static { System.loadLibrary(Core.NATIVE_LIBRARY_NAME); }
7
8      public static void main(String[] args) {
9          Mat img = Imgcodecs.imread("women.png", Imgcodecs.IMREAD_GRAYSCALE);
10
11          if (img.empty()) {
12              System.out.println("Failed to load image.");
13              return;
14          }
15          // Binaryzacja obrazu
16          Mat binaryImg = new Mat();
17          Imgproc.threshold(img, binaryImg, 127, 255, Imgproc.THRESH_BINARY);
18
19          // Definicje rozmiarów i kształtów elementów strukturalnych
20          Size[] sizes = {new Size(3, 3), new Size(5, 5)};
21          int[] shapes = {Imgproc.MORPH_RECT, Imgproc.MORPH_ELLIPSE, Imgproc.MORPH_CROSS};
22
23          // Liczba powtórzeń erozji
24          int numErosions = 3;
25          // Wykonaj wielokrotną erozję dla różnych elementów strukturalnych i zapisz wyniki
26          for (Size size : sizes) {
27              for (int shape : shapes) {
28                  Mat element = Imgproc.getStructuringElement(shape, size);
29                  Mat erodedImg = binaryImg.clone();
30                  // Powtórzona erozja
31                  for (int i = 0; i < numErosions; i++) {
32                      Imgproc.erode(erodedImg, erodedImg, element);
33                  }
34                  // Zapisanie wyniku do pliku
35                  String outputFilename = String.format("eroded_shape%d_size%0fx%0f_repetitions%d.png", shape, size.width, size.height, numErosions);
36                  Imgcodecs.imwrite(outputFilename, erodedImg);
37                  System.out.println("Zapisano: " + outputFilename);
38              }
39          }
40      }
41  }

```



Dla porównania. Wielokrotna erozja, powoduje, że obiekty na obrazie zmniejszają się. Drobne szczegóły mogą całkowicie zniknąć.

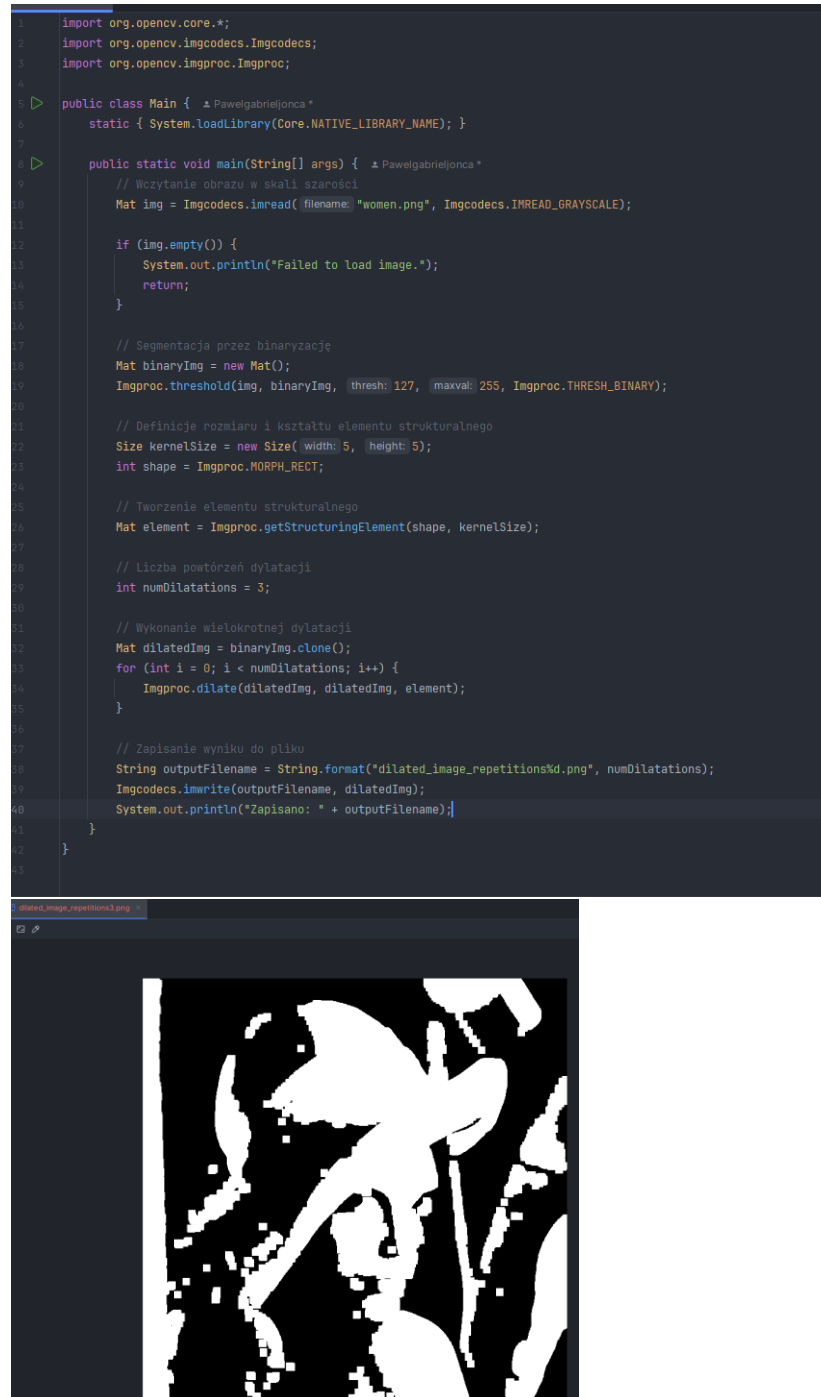
### Zad 3

```
Main.java x  dilated_image.png
1  import org.opencv.core.*;
2  import org.opencv.imgcodecs.Imgcodecs;
3  import org.opencv.imgproc.Imgproc;
4
5  public class Main {  Pawelgabrieljonca *
6      static { System.loadLibrary(Core.NATIVE_LIBRARY_NAME); }
7
8      public static void main(String[] args) {  Pawelgabrieljonca *
9          Mat img = Imgcodecs.imread( filename: "women.png", Imgcodecs.IMREAD_GRAYSCALE);
10
11          if (img.empty()) {
12              System.out.println("Failed to load image.");
13              return;
14          }
15
16          // Segmentacja poprzez binaryzację obrazu (progowanie)
17          Mat binaryImg = new Mat();
18          Imgproc.threshold(img, binaryImg, thresh: 127, maxval: 255, Imgproc.THRESH_BINARY);
19
20          // Definicje rozmiarów i kształtów elementów strukturalnych
21          Size kernelSize = new Size( width: 5, height: 5);
22          int shape = Imgproc.MORPH_RECT;
23
24          // Tworzenie elementu strukturalnego
25          Mat element = Imgproc.getStructuringElement(shape, kernelSize);
26
27          // Wykonanie dylatacji
28          Mat dilatedImg = new Mat();
29          Imgproc.dilate(binaryImg, dilatedImg, element);
30
31          // Zapisanie wyniku do pliku
32          String outputFilename = "dilated_image.png";
33          Imgcodecs.imwrite(outputFilename, dilatedImg);
34          System.out.println("Zapisano: " + outputFilename);
35      }
36  }
37
```



## Zad 4

Operacja dylatacji powoduje rozszerzenie jasnych obszarów na obrazie, co może pomóc w wypełnieniu luk w obiektach, połączeniu pobliskich obszarów, a także zmniejszeniu wpływu szumów i małych czarnych elementów w białych obszarach.



Wielokrotna dylatacja prowadzi do dalszego rozszerzenia jasnych obszarów obrazu.

## Zad 5

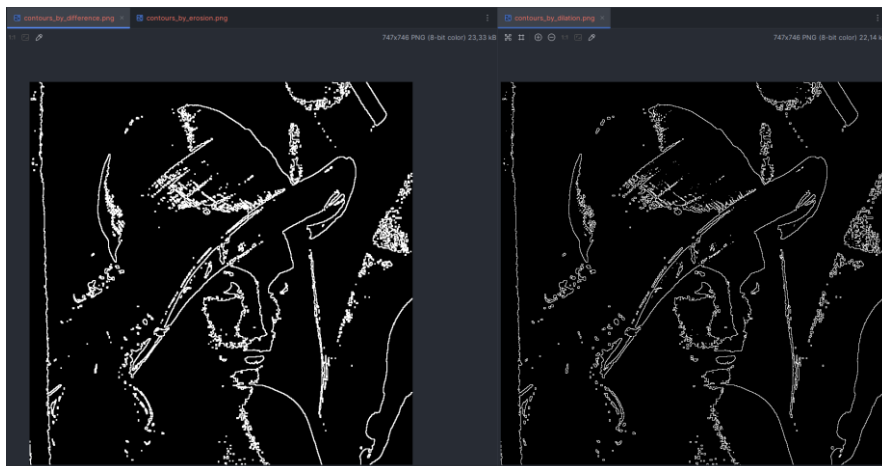


Efekt otwarcia : usunięcie drobnych obiektów. Otwarcie jest szczególnie skuteczne w usuwaniu małych, jasnych obiektów (szumów) na obrazie, pozostawiając większe struktury

Efekt domknięcia : wypełnienie małych dziur. Domknięcie zamyka niewielkie, ciemne dziury wewnątrz białych obiektów, co skutkuje ich „zagęszczeniem”

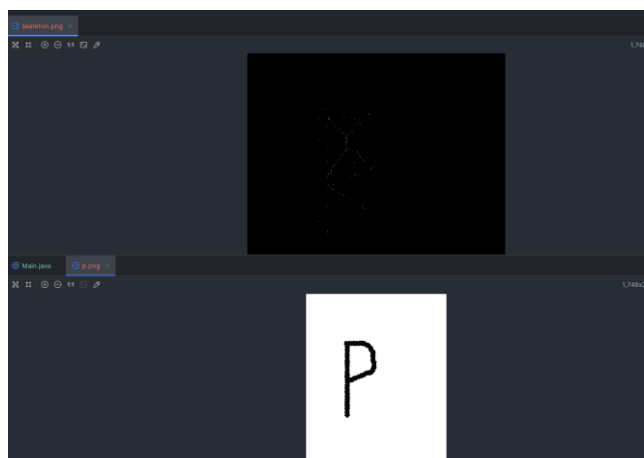
## Zad 6

```
1 > import ...
4 public class Main { /* Pawelgabrieljonca */
5     static { System.loadLibrary(Core.NATIVE_LIBRARY_NAME); }
6     public static void main(String[] args) { /* Pawelgabrieljonca */
7         Mat img = Imgcodecs.imread( filename: "women.png", Imgcodecs.IMREAD_GRAYSCALE);
8         if (img.empty()) {
9             System.out.println("Failed to load image.");
10            return;
11        }
12        // Segmentacja poprzez binaryzację
13        Mat binaryImg = new Mat();
14        Imgproc.threshold(img, binaryImg, thresh: 127, maxval: 255, Imgproc.THRESH_BINARY);
15
16        // Definicje rozmiaru i kształtu elementu strukturalnego
17        Size kernelSize = new Size( width: 3, height: 3);
18        int shape = Imgproc.MORPH_RECT;
19        Mat element = Imgproc.getStructuringElement(shape, kernelSize);
20
21        // Dylatacja obrazu
22        Mat dilatedImg = new Mat();
23        Imgproc.dilate(binaryImg, dilatedImg, element);
24
25        // Erozja obrazu
26        Mat erodedImg = new Mat();
27        Imgproc.erode(binaryImg, erodedImg, element);
28
29        // Metoda 1: Kontury przez dylatację
30        Mat contoursByDilation = new Mat();
31        Core.subtract(dilatedImg, binaryImg, contoursByDilation);
32        Imgcodecs.imwrite( filename: "contours_by_dilation.png", contoursByDilation);
33        System.out.println("Zapisano: contours_by_dilation.png");
34
35        // Metoda 2: Kontury przez erozję
36        Mat contoursByErosion = new Mat();
37        Core.subtract(binaryImg, erodedImg, contoursByErosion);
38        Imgcodecs.imwrite( filename: "contours_by_erosion.png", contoursByErosion);
39        System.out.println("Zapisano: contours_by_erosion.png");
40
41        // Metoda 3: Kontury przez różnicę między dylatacją a erozją
42        Mat contoursByDifference = new Mat();
43        Core.subtract(dilatedImg, erodedImg, contoursByDifference);
44        Imgcodecs.imwrite( filename: "contours_by_difference.png", contoursByDifference);
45        System.out.println("Zapisano: contours_by_difference.png");
46    }
47 }
```



## Zad 7

```
1 > import ...
4 public class Main { /* Pawelgabrieljonca */
5     static { System.loadLibrary(Core.NATIVE_LIBRARY_NAME); }
6
7     public static void main(String[] args) { /* Pawelgabrieljonca */
8         Mat img = Imgcodecs.imread(filename: "p.png", Imgcodecs.IMREAD_GRAYSCALE);
9
10        if (img.empty()) {
11            System.out.println("Failed to load image.");
12            return;
13        }
14        // Binarizacja obrazu
15        Mat binaryImg = new Mat();
16        Imgproc.threshold(img, binaryImg, thresh: 127, maxval: 255, Imgproc.THRESH_BINARY);
17
18        // Szkieletowanie przy pomocy iteracyjnej erozji i dyatacji
19        Mat skeleton = Mat.zeros(binaryImg.size(), CvType.CV_8UC1);
20        Mat temp = new Mat();
21        Mat eroded = new Mat();
22        Mat element = Imgproc.getStructuringElement(Imgproc.MORPH_CROSS, new Size(width: 3, height: 3));
23        boolean done;
24        do {
25            // Wykonanie erozji
26            Imgproc.erode(binaryImg, eroded, element);
27
28            // Wykonanie dyatacji na erodowanym obrazie
29            Imgproc.dilate(eroded, temp, element);
30
31            // Odejmowanie, aby uzyskać różnicę
32            Core.subtract(binaryImg, temp, temp);
33
34            // Sumowanie wyniku z dotychczasowym szkieletem
35            Core.bitwise_or(skeleton, temp, skeleton);
36
37            // Ustawienie erodowanego obrazu jako nowego obrazu wejściowego
38            eroded.copyTo(binaryImg);
39            // Sprawdzenie, czy obraz został całkowicie erodowany
40            done = Core.countNonZero(binaryImg) == 0;
41        } while (!done);
42        // Zapisanie obrazu szkieletu do pliku
43        String outputFilename = "skeleton.png";
44        Imgcodecs.imwrite(outputFilename, skeleton);
45        System.out.println("Zapisano: " + outputFilename);
46    }
47 }
```



Wnioski:



Wykonując wszystkie zadania mogłem poznać dalsze funkcje jakie oferuje biblioteka OpenCV. Zadania pokazały operacje morfologiczne czyli erozja i dylatacja również otwarcie i domknięcie. Ekstrakcja konturów oraz Szkieletowanie. Dzięki tym operacją można osiągnąć różne cele, takie jak oczyszczanie z szumów, segmentacji oraz uzyskania cienkich reprezentacji obiektów.