МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
КАФЕДРА «ЭВМ и системы»

«Увеличение и уменьшение цифровых изображений»

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2
по дисциплине «Цифровая обработка сигналов»

Листов **6**

Выполнил студент группы Э-56
Никитюк В. А.

Проверил Дубицкий А. В.

Брест 2020

Цель: Изучить методы увеличения и уменьшения цифровых изображений и применить полученные знания на практике.

Ход работы:

Задание для первого варианта: написать программу способную производить увеличение/уменьшение исходного изображения в нецелое число раз методом ближайшего соседа. Программа должна сохранять полученное изображение в виде файла формата BMP. С помощью программы уменьшить исходное изображение в 1,5; 2; 5,3; 15 раз. Полученные изображения затем восстановить до исходного размера и сравнить результаты с исходным изображением.

Код программы:

Class Main:

```
package com.company;

import java.io.File;

public class Main {

    public static void main(String[] args) {
        if (args.length < 4)
            return;
        else {
            File input = new File(args[0]);
            if (input.exists()) {
                File output = new File(args[3]);
                if(args[2].equals("1")){
                    Neighbor.compress(input,
Double(args[1]), output);
                    Neighbor.resize(output,
new  Double(args[1]).doubleValue(),
                    new File(args[3]+"RESTORE.bmp"));
                }
                if(args[2].equals("0")){
```

```
                    Neighbor.resize(input,
new  Double(args[1]), output);
                    Neighbor.compress(output,
new  Double(args[1]), new File(args[3]+"RESTORE.bmp"));
                }
            } else {
                System.out.println("Enter correct image path");
            }
        }
    }
}
```

Class Nieghbor:

```
package com.company;

import javax.imageio.ImageIO;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

public class Neighbor {

    public static void compress(File image, double koef,
    File output){
        BufferedImage sourcePicture = null;
        try {
            sourcePicture = ImageIO.read(image);
        }catch (IOException e){
            System.out.println(e.getMessage());
        }
        Integer outputWidth =
    (int) (sourcePicture.getWidth() / koef);
        Integer outputHeight =
    (int) (sourcePicture.getHeight() / koef);
```

```
    BufferedImage outputPicture =
new BufferedImage(outputWidth, outputHeight,
sourcePicture.getType());
    double y=0, x=0;
    for(int i = 0; i < outputHeight; i++){
        for (int j = 0; j < outputWidth; j++){
            outputPicture.setRGB(j,i,sourcePicture.getRGB((int
            x+=koef;
        }
        y+=koef;
        x=0;
    }
    try {
        ImageIO.write(outputPicture,"bmp",output);
    } catch (IOException e) {
        e.printStackTrace();
    }
}

public static void resize(File image, double koef,
File output) {
    BufferedImage sourcePicture = null;
    try {
        sourcePicture = ImageIO.read(image);
    }catch (IOException e){
        System.out.println(e.getMessage());
    }
    Integer outputWidth =
    (int) (sourcePicture.getWidth() * koef);
    Integer outputHeight =
    (int) (sourcePicture.getHeight() * koef);
    double y=0, x=0;
    BufferedImage outputPicture =
```

```
            new BufferedImage(outputWidth, outputHeight, sourcePicture
            for (int i = 0; i < outputHeight; i++) {
                for (int j = 0; j < outputWidth; j++) {
                    outputPicture.setRGB(j, i, sourcePicture.getRGB((int
                    x+=(1.0/koef);
                }
                y+=(1.0/koef);
                x=0;
            }
            try {
                ImageIO.write(outputPicture, "bmp", output);
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

}
```

Рисунок 1 — Исходное изображение

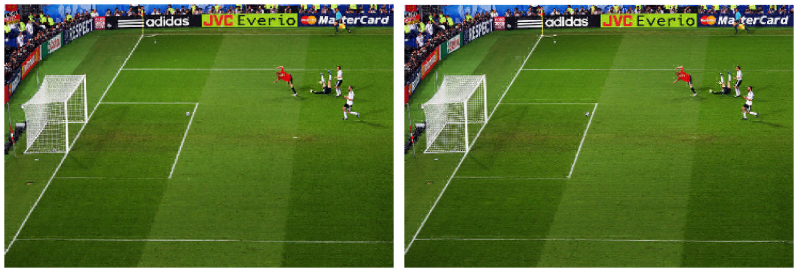

Рисунок 2 — Сжатие 1,5 и 2

Рисунок 3 — Сжатие 5,3 и 15

Вывод: изучили методы увеличения и уменьшения цифровых изображений и применили полученные знания на практике, в изображении увеличенном/уменьшенном с помощью метода ближайшего соседа изображние пикселизируется при увелечении и получается сильно зернистым при уменьшении. Для каждого пикселя конечного изображения выбирается 1 пиксель исходного, наиболее близкий к его положению с учётом масштабирования