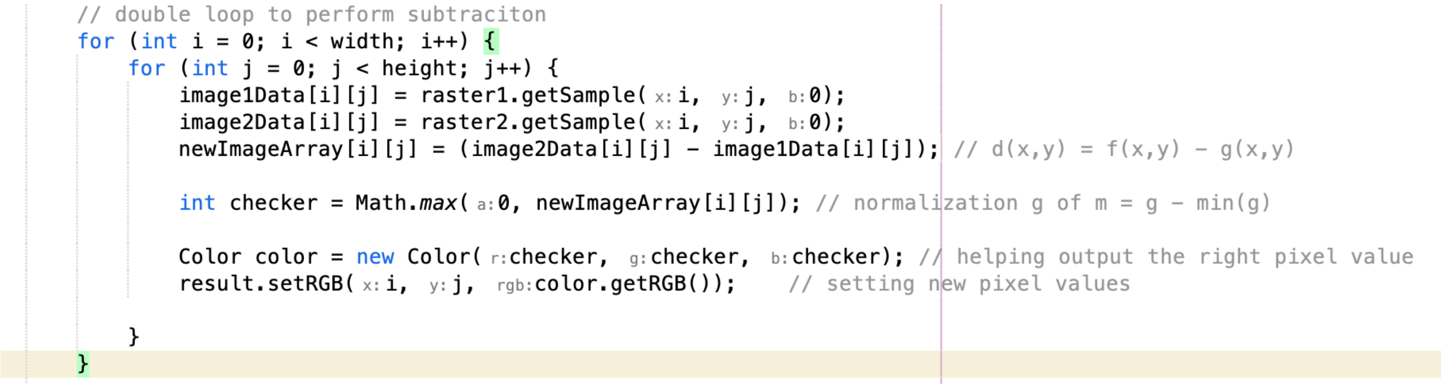
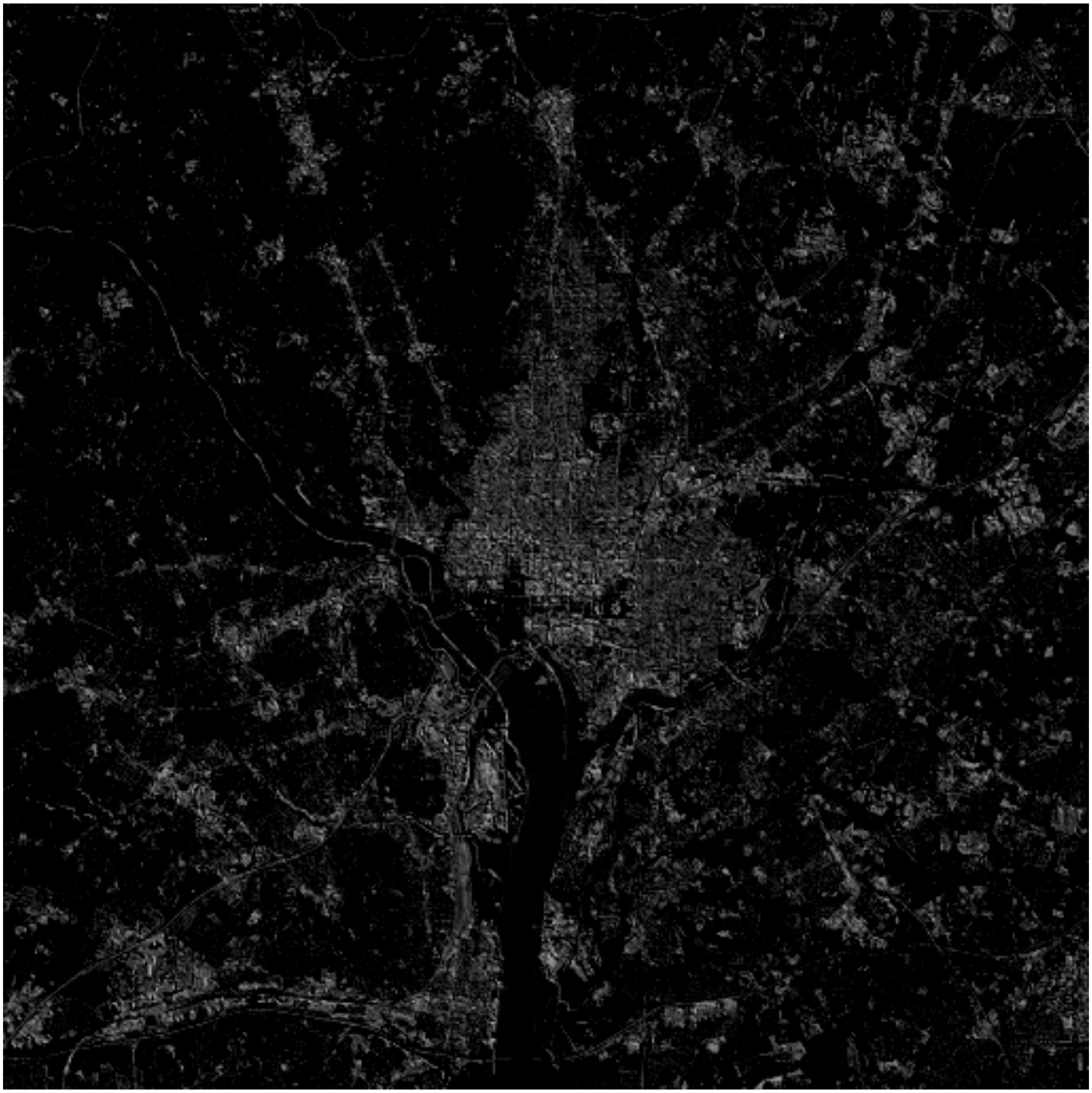
PROJECT CODED IN APACHE NETBEANS 16

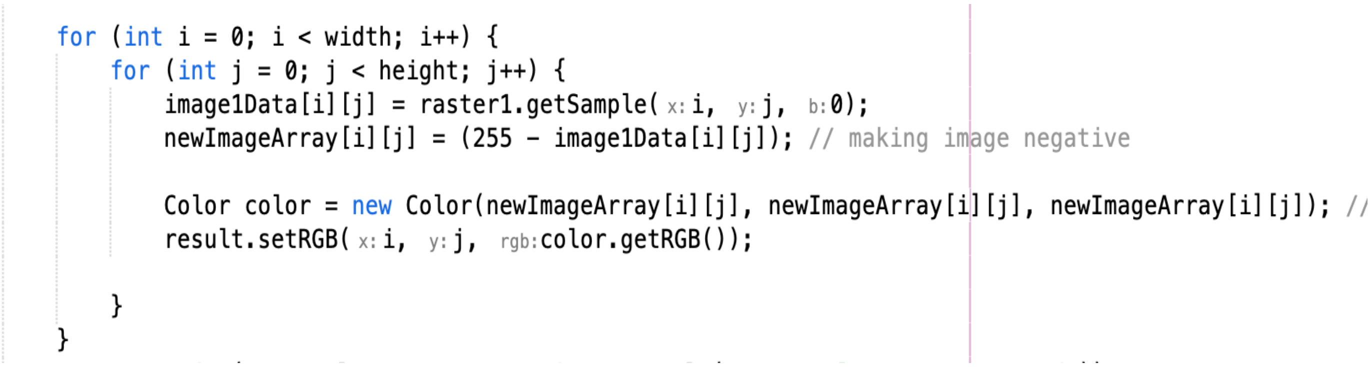
1. Subtraction



```
// double loop to perform subtraciton
for (int i = 0; i < width; i++) {
    for (int j = 0; j < height; j++) {
        image1Data[i][j] = raster1.getSample( x: i,  y: j,  b: 0);
        image2Data[i][j] = raster2.getSample( x: i,  y: j,  b: 0);
        newImageArray[i][j] = (image2Data[i][j] - image1Data[i][j]); // d(x,y) = f(x,y) - g(x,y)

        int checker = Math.max( a: 0, newImageArray[i][j]); // normalization g of m = g - min(g)

        Color color = new Color( r: checker,  g: checker,  b: checker); // helping output the right pixel value
        result.setRGB( x: i,  y: j,  rgb: color.getRGB());    // setting new pixel values

    }
}
```

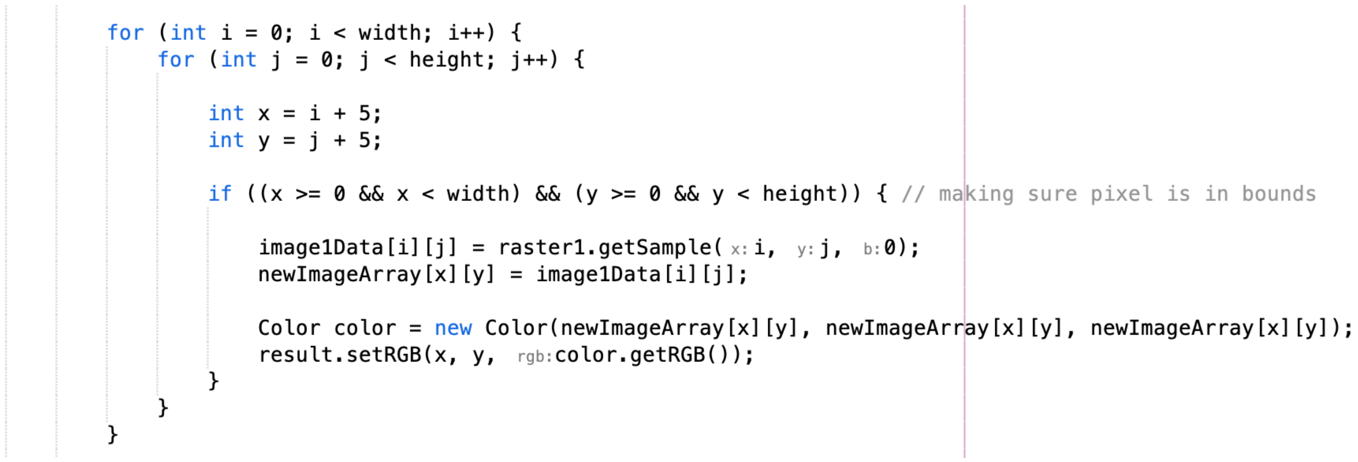


2. Negative image



```
for (int i = 0; i < width; i++) {
    for (int j = 0; j < height; j++) {
        image1Data[i][j] = raster1.getSample( x: i,  y: j,  b: 0);
        newImageArray[i][j] = (255 - image1Data[i][j]); // making image negative

        Color color = new Color(newImageArray[i][j], newImageArray[i][j], newImageArray[i][j]); //
        result.setRGB( x: i,  y: j,  rgb: color.getRGB());

    }
}
```

3. Translation



```
        for (int i = 0; i < width; i++) {
            for (int j = 0; j < height; j++) {

                int x = i + 5;
                int y = j + 5;

                if ((x >= 0 && x < width) && (y >= 0 && y < height)) { // making sure pixel is in bounds

                    image1Data[i][j] = raster1.getSample( x: i,  y: j,  b: 0);
                    newImageArray[x][y] = image1Data[i][j];

                    Color color = new Color(newImageArray[x][y], newImageArray[x][y], newImageArray[x][y]);
                    result.setRGB(x, y,  rgb: color.getRGB());
                }
            }
        }
```