Runigram.read

```
public static Color[][] read(String fileName) {
        In in = new In(fileName);
        in.readString();
        int numCols = in.readInt();
        int numRows = in.readInt();
        in.readInt();
        Color[][] image = new Color[numRows][numCols];
        for (int i = 0; i < numRows; i++) {
            for (int j = 0; j < numCols; j++) {
                image[i][j] = new Color(in.readInt(), in.readInt(),
                                        in.readInt());
            }
        }
        return image;
    }
```

Runigram.print

```
private static void print(Color[][] image) {
        for (int i = 0; i < image.length; i++) {
            for (int j = 0; j < image[i].length; j++) {
                print(image[i][j]);
            }
        System.out.println ();
        }
    }
```

Runigram.flippedHorizontally

```
public static Color[][] flippedHorizontally(Color[][] image) {
        Color[][] flipped = new Color[image.length][image[0].length];
        for (int i = 0; i < image.length; i++) {
            for (int j = 0; j < image[i].length; j++) {
                flipped[i][j] = image[i][image[i].length - j - 1];
            }
        }
        return flipped;
    }
```

Runigram,flippedVertically

```
public static Color[][] flippedVertically(Color[][] image){
        Color[][] flipped = new Color[image.length][image[0].length];
        for (int i = 0; i < image.length; i++) {
            for (int j = 0; j < image[i].length; j++) {
                flipped[i][j] = image[image.length - 1 - i][j];
            }
        }
        return flipped;
    }
```

Runigram.luminance

```
    public static Color luminance(Color pixel) {
        int lumValue = (int)(0.299 * pixel.getRed() + 0.587 *
                             pixel.getGreen() + 0.114 *
                             pixel.getBlue());
        Color lum = new Color(lumValue, lumValue, lumValue);
        return lum;
    }
```

Runigram.grayScalled

```
    public static Color[][] grayScaled(Color[][] image) {
        Color[][] gScale = new Color[image.length][image[0].length];
        for (int i = 0; i < image.length; i++) {
            for (int j = 0; j < image[i].length; j++) {
                gScale[i][j] = luminance(image[i][j]);
            }
        }
        return gScale;
    }
```

Runigram.scalled

```
    public static Color[][] scaled(Color[][] image, int width, int
                                   height) {
        int cH = image.length;
        int cW = image[0].length;
        Color[][] scaledImage = new Color[height][width];
        for (int i = 0; i < height; i++) {
            for (int j = 0; j < width; j++) {
                scaledImage[i][j] = image[i*cH/height][j*cW/width];
            }
        }
        return scaledImage;
    }
```

Runigram.blend

```
    public static Color blend(Color c1, Color c2, double alpha) {
        double red = alpha*c1.getRed() + (1-alpha)*c2.getRed();
        double green = alpha*c1.getGreen() + (1-alpha)*c2.getGreen();
        double blue = alpha*c1.getBlue() + (1-alpha)*c2.getBlue();
        Color blended = new Color((int) red, (int) green, (int) blue);
        return blended;
    }
```

Runigram.blend

```
    public static Color[][] blend(Color[][] image1, Color[][] image2,
                                  double alpha) {
        Color[][] blended = new
                            Color[image1.length][image1[0].length];
        for (int i = 0; i < blended.length; i++) {
            for (int j = 0; j < blended[i].length; j++) {
                blended[i][j] = blend(image1[i][j], image2[i][j],
                alpha);
            }
        }
        return blended;
    }
```

Runigram.morph

```
        public static void morph(Color[][] source, Color[][] target,
                                 int n) {
        Color[][] scaledTarget = new Color
                                [source.length][source[0].length];
        scaledTarget = scaled(target, source[0].length,
                              source.length);
        double alpha = 0;
        for (int i = 0; i < n; i++){
            alpha = (n - i)/n;
            display(blend(source, scaledTarget, alpha));
            StdDraw.pause(500);
        }
    }
```

Editor4

```
import java.awt.Color;

public class Editor4 {

    public static void main (String[] args) {
        String source = args[0];
        int n = Integer.parseInt(args[1]);
        Color[][] sourceImage = Runigram.read(source);
        Runigram.setCanvas(sourceImage);
        Runigram.morph(sourceImage, Runigram.grayScaled(sourceImage),
                       n);
    }
}
```