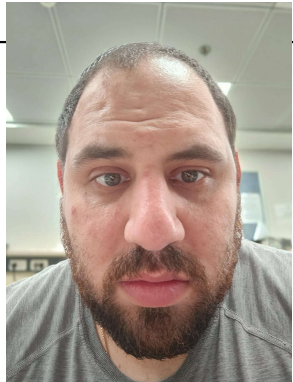




# Course: Image Processing 31651

## Assignment #12

### Synthetic Image Creation (Part 2) – version 2

|            | ID (4 last digits) | Shorten Name | Photo of the student  |  |
|------------|--------------------|--------------|---|--|
| Student #1 | 1950               | shienfeld    |    |  |
| Student #2 | 2210               | pony         |  |  |
| Student #3 | 7939               | akimov       |   |  |

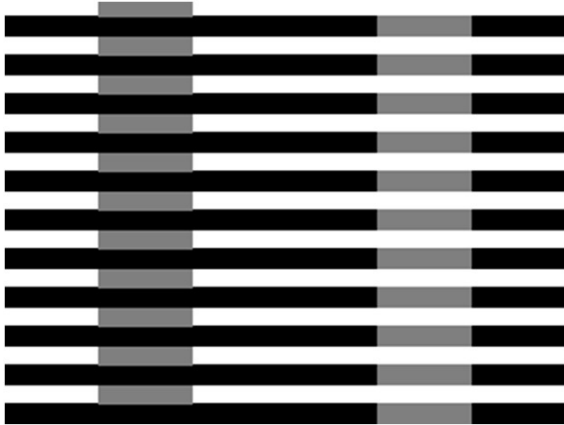
## Assignment #12: Create image demonstrating White' illusion

In the file ImProcInPlainC.h defined numerical values of:  
NUMBER\_OF\_ROWS and NUMBER\_OF\_COLUMNS

| # | Image Description  |
|---|--|
| 1 | <p>By using function "AddGrayRectangle" ONLY from Assignment 11,<br/>write function "void createWhitesIllusion( unsigned char img[][NUMBER_OF_COLUMNS]<br/>int numberOfBlackHorizontalStrips,<br/>unsigned char grayLevelOfGrayBars )" demonstrating White's Illusion as described on Figure 1 at:<br/><a href="https://en.wikipedia.org/wiki/White%27s_illusion">https://en.wikipedia.org/wiki/White%27s_illusion</a></p> |
| 2 | <p>By using above function with different parameters (numberOfBlackHorizontalStrips, grayLevelOfGrayBars ) ,<br/>create at least 3 gray BMP files "grayImage12n.bmp" (n=1,2,3)</p>   |
|   | <p><b>For report use the following template.</b><br/><b>Done: 12.1 12.2 12.3 12.4 12.5</b></p> <p><b>REMOVE ALL RED when the items are ready</b><br/><b>Important hint: Fill report pages while working.</b><br/><b>Do not complain "not enough time" in case you did not follow THIS rule.</b></p>  |

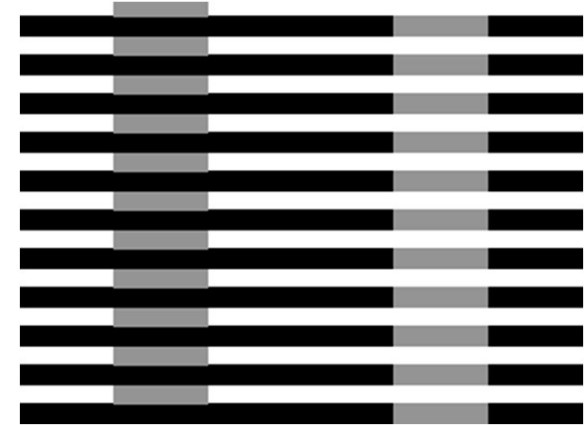
## 12.1 - our resulted images “grayImage12n.bmp” with add short comment– part 1

grayImage121.bmp



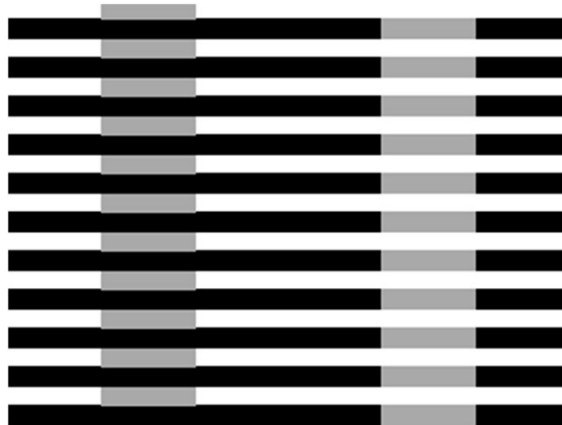
gray level of 127

grayImage122.bmp



gray level of 148

grayImage123.bmp



gray level of 169

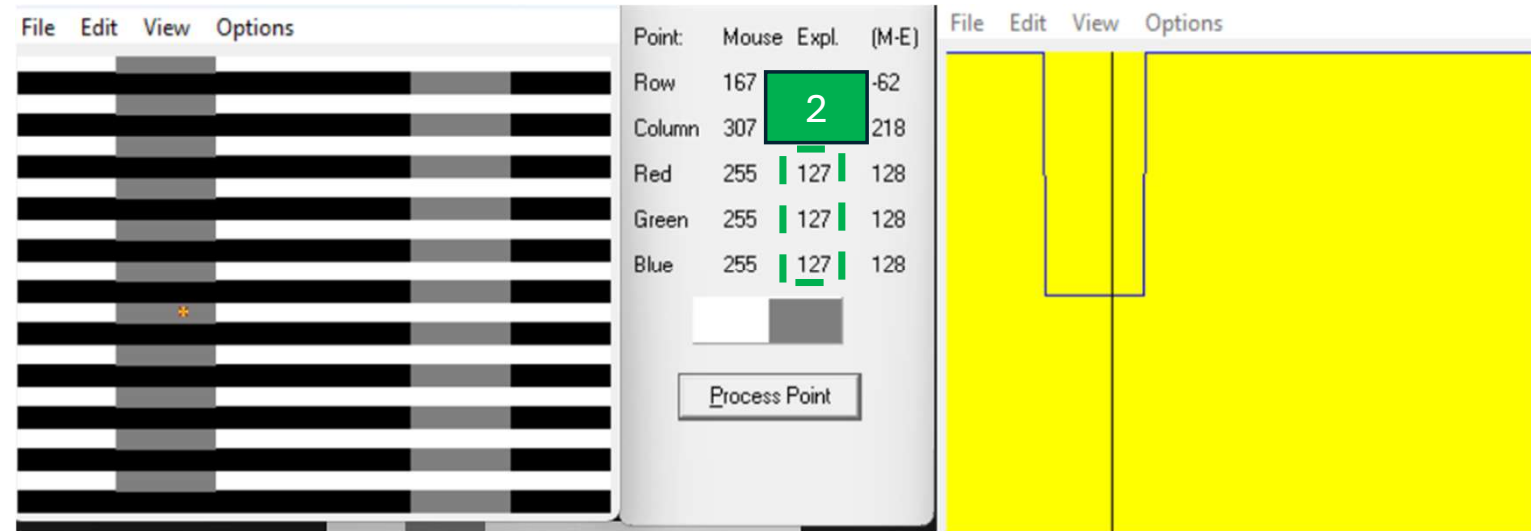
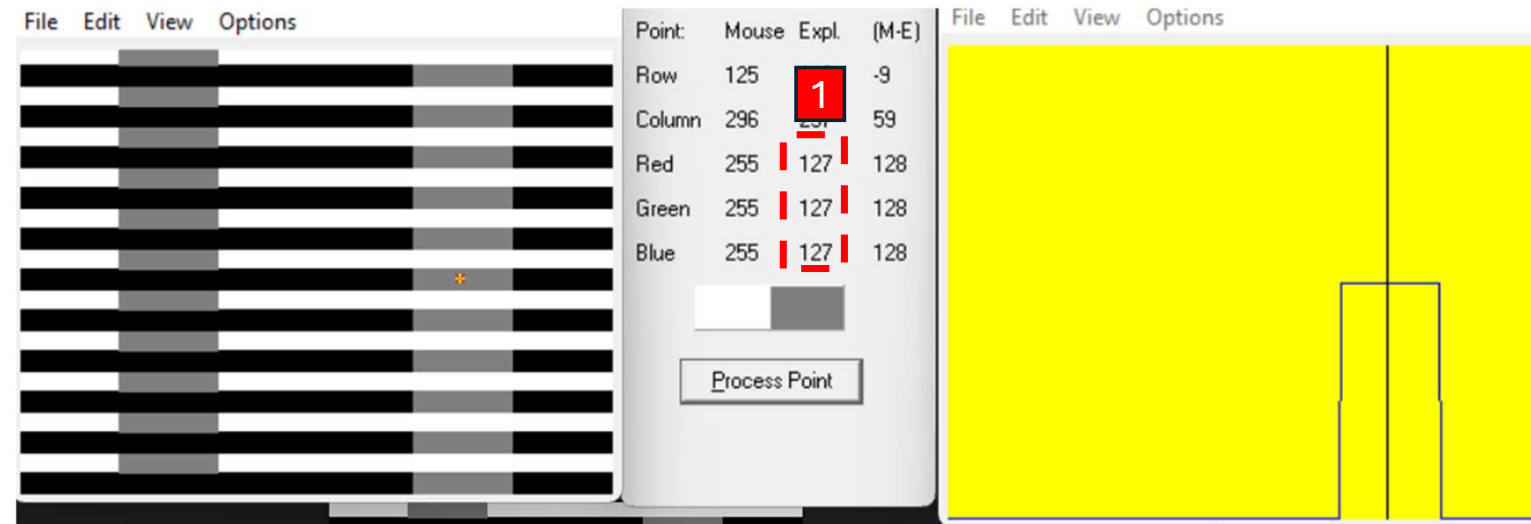
We decided to change only the gray level between the 3 examples, but the number of the black horizontal stripes is a parameter which can be changed anytime easily

## 12.2 - Relevant Profiles – part 1/3

4

grayImage121.bmp

We present two points and their profiles. We can see that the first point has gray level of 127 and the second is also 127. Although they look different to our human eye - they have the same gray level. (this is the illusion...)

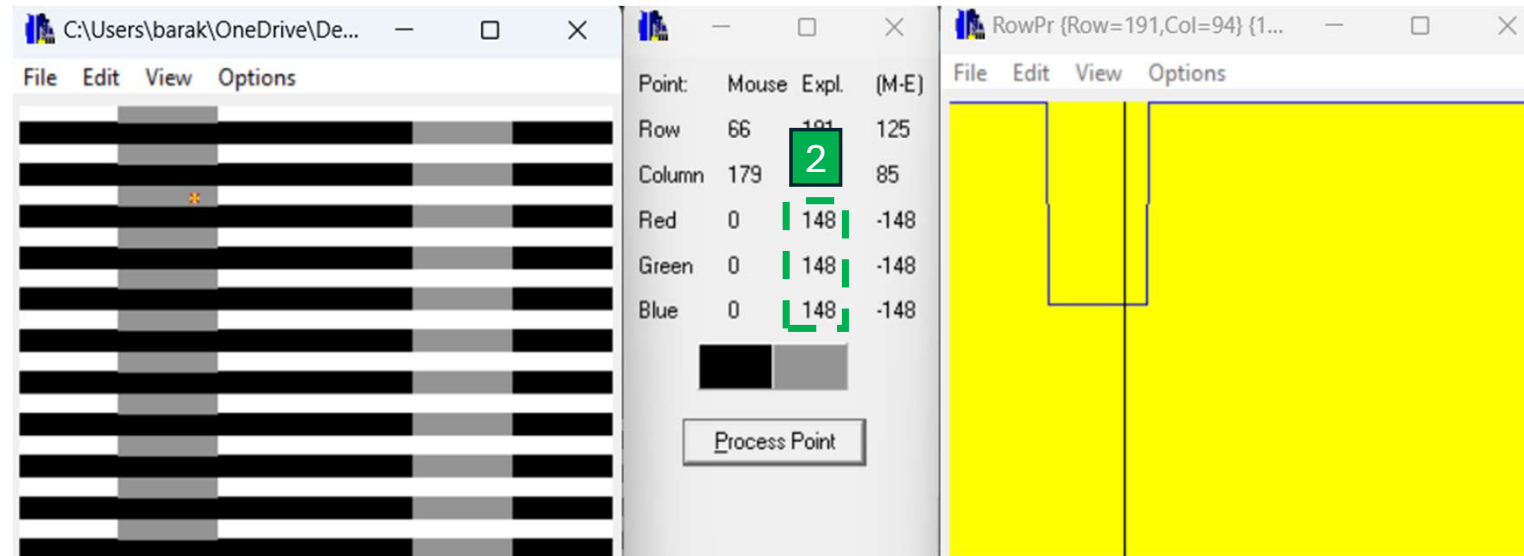
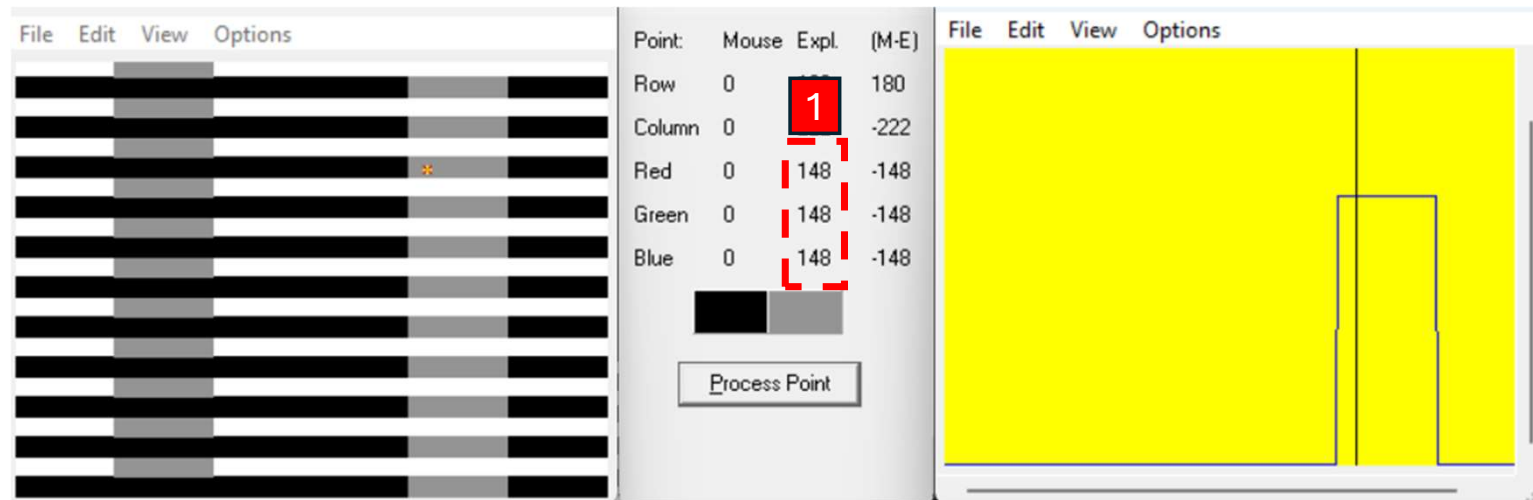


## 12.2 - Relevant Profiles – part 2/3

5

grayImage122.bmp

We present two points and their profiles. We can see that the first point has gray level of 148 and the second is also 148. Although they look different to our human eye - they have the same gray level. (this is the illusion...)

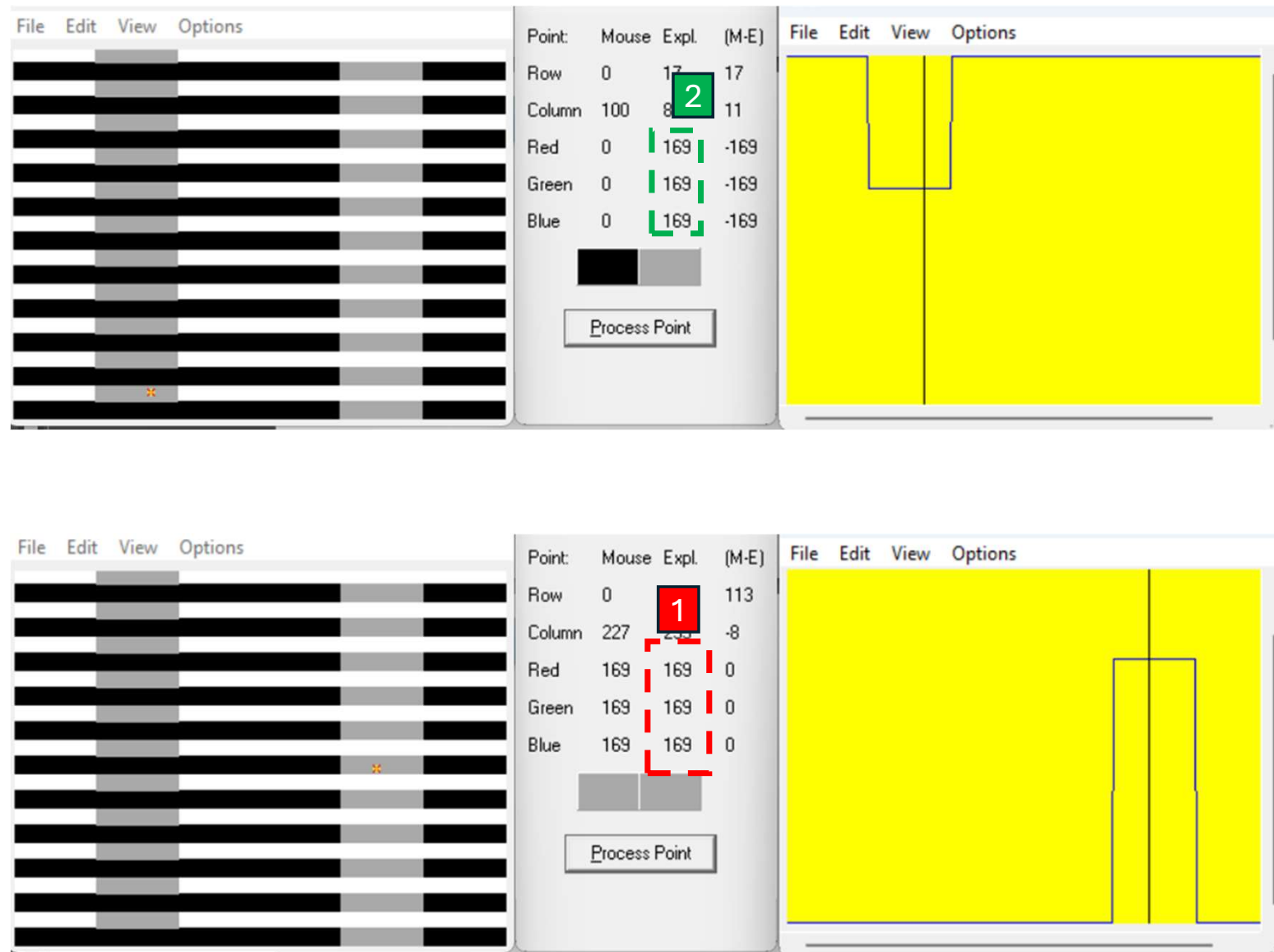


## 12.2 - Relevant Profiles – part 3/3

6

grayImage123.bmp

We present two points and their profiles. We can see that the first point has gray level of 169 and the second is also 169. Although they look different to our human eye - they have the same gray level . (this is the illusion...)





## 12.3 - Code of the function “createWhitesIllusion” – part 1/3

7

```
33 void createWhitesIllusion(unsigned char img[][NUMBER_OF_COLUMNS], int numberOfBlackHorizontalStrips,  
34 unsigned char grayLevelOfGrayBars) {  
35     s2dPoint leftside{ 0, 0 }, rightside{ NUMBER_OF_COLUMNS - 1, NUMBER_OF_ROWS - 1 }; // Corrected coordinates  
36     AddGrayRectangle(img, leftside, rightside, 0, 255); // Full screen white rectangle  
37     int stripHeight = (rightside.gety() - leftside.gety()) / numberOfBlackHorizontalStrips;  
38     if (rightside.getx() < leftside.getx())  
39         leftside.change(rightside);  
40     int i = leftside.gety();  
41     int bound_right = rightside.gety();  
42     for (; i < bound_right; i += stripHeight) {  
43         s2dPoint p1, p2;  
44         int narrowStripWidth = abs(leftside.getx() - rightside.getx()) / 6;  
45         if ((i / stripHeight) % 2 == 0) {  
46             // Add black strip  
47             /* p1.setXY(leftside.getx(), i);  
48              p2.setXY(rightside.getx(), i + stripHeight);*/  
49             p1.setXY(leftside.getx(), i);  
50             p2.setXY(rightside.getx(), i + stripHeight);  
51             AddGrayRectangle(img, p1, p2, 0, 0);  
52             // Second narrow strip  
53             p1.setXY(4 * narrowStripWidth, i);  
54             p2.setXY(5 * narrowStripWidth, i + stripHeight);  
55             AddGrayRectangle(img, p1, p2, 0, grayLevelOfGrayBars);  
56         }  
57         else {  
58             p1.setXY(narrowStripWidth, i);  
59             p2.setXY(2 * narrowStripWidth, i + stripHeight);  
60             AddGrayRectangle(img, p1, p2, 0, grayLevelOfGrayBars);  
61         }  
62     }  
63     StoreGrayImageAsGrayBmpFile(img, "white_illusion.bmp");  
64 }
```

The function starts by filling the entire image with a white background. It then calculates the height of each black strip based on the total number of strips specified.

The function iterates over the image height, adding black strips for even indexed rows and narrow gray bars within both the black and white strips, adjusting their positions accordingly.

This is done using the AddGrayRectangle function, which ensures the correct placement and shading of the rectangles, creating an illusion where the gray bars appear differently depending on their background.

## 12.4 - Code of the main function

8

```
12  int main() {
13      Second_assignment();
14      return 0;
15  }
16  void Second_assignment()
17  {
18      const int numIllusion = 3;
19      unsigned char grayLevels[numIllusion];
20      int numberOfBlackHorizontalStrips = 20;
21      for (int i = 0; i < numIllusion; i++) {
22
23          grayLevels[i] = static_cast<unsigned char>((static_cast<float>(i) / numIllusion) * 255) / 4 + 127.5;
24
25          createWhitesIllusion(img2
26              , numberOfBlackHorizontalStrips, grayLevels[i]);
27          char filename[30];
28          sprintf(filename, "grayImage12%d.bmp", i+1);
29          StoreGrayImageAsGrayBmpFile(img2, filename);
30
31      }
32  }
```

The number of the black horizontal stripes – it is possible to convert the constant to an array such that every image will be different both in gray level and in the number of black stripes. We decided not to deal with it right now

The main function initializes parameters for creating multiple versions of White's Illusion on an image.

It defines the number of illusions to be created, sets transparency and gray level values for each, and iterates to generate these illusions using the **createWhitesIllusion** function.

After creating each illusion, it saves the resulting image as a BMP file with a unique filename. This process is repeated for each set of transparency and gray level values to produce and store multiple images demonstrating the illusion.



## 12.5 - What did we learn ?

understand how grayscale images are represented and manipulated, using values from 0 (black) to 255 (white).

Learn how to blend different transparency levels and gray levels to create visual effects.

Apply geometric concepts to determine the positions and dimensions of shapes within an image.

Understand techniques for generating dynamic filenames to save multiple output files.