PRINCIPLES OF DIGITAL IMAGE

1. Give the Dinosaur picture a drawing effect.

In this study, it is aimed to make the read.png image look like drawing by passing it through several filters. Below are the steps followed.

A) Converting the image to grayscale

I converted our image to Gray scale.

```
imgInput.Load("read.png");
imgInputGr = GrayscaleImage(imgInput);
imgInputGr.Save("Grayscale.png");
```

B) Inverting the image

Black and white image inversion, I converted the light areas to dark and dark areas to light.

```
for (int y = 0; y < imgNegative.GetHeight(); y++)
{
  for (int x = 0; x < imgNegative.GetWidth(); x++)
  {
    imgNegative(x, y) = 255 - imgInput(x, y);
  }
}
imgNegative.Save("imgNegative.png");
```

C) Box Filtering that blurs the inverted image

Using the average filter, I have blurred the image, is to replace each pixel value of an image with the average value including its neighbors and itself. This results in the disappearance of pixel values that are not representative of their surroundings. I used the kernel size as 3×3 square.

```
imgBlur = imgNegative;
auto getValues = [&](int x, int y) { return imgNegative(x > -1 && x < imgNegative.GetWidth() ? x : 1,
y > -1 && x < imgNegative.GetHeight() ? y : 1);};
for (int i = 0; i < 10; i++)
{
  for (int y = 0; y < imgBlur.GetHeight() - 1; y++)
  {
    for (int x = 0; x < imgBlur.GetWidth() - 1; x++)
    {
      imgBlur(x, y) = (getValues(x - 1, y - 1) + getValues(x, y - 1) + getValues(x + 1, y - 1) +
      getValues(x - 1, y) + getValues(x, y) + getValues(x + 1, y) +
      getValues(x - 1, y + 1) + getValues(x, y + 1) + getValues(x + 1, y + 1)) / 9;
    }
  }
  imgNegative = imgBlur;
}
imgBlur.Save("imgBlur.png");
```

D) Decomposition by pixel's intensity of the blurred image

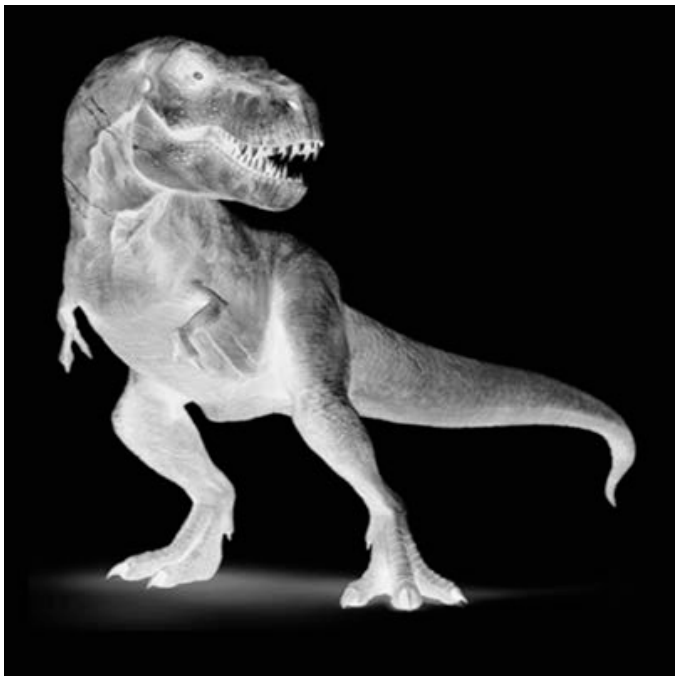
I changed the pixels with high values to white pixels, then I changed the pixels with density 0 to white (255) to change the background to white. Then, to highlight the lines that make up the outer surface of the picture, I synced the pixels with a certain value range to ImgFinal.

```
imgFinal = imgInput;
for (int y = 0; y < imgFinal.GetHeight(); y++)
{
  for (int x = 0; x < imgFinal.GetWidth(); x++)
  {
    if (imgBlur(x, y) >= 0 && imgBlur(x, y) < 10 )
    {
      imgFinal(x, y) = 255;
    }
    else if (imgBlur(x, y) >= 100 && imgBlur(x,y) < 255 )
    {
      imgFinal(x, y) = 255;
    }
    else
    {
      imgFinal(x, y) = imgBlur(x, y);
    }
  }
}
imgFinal.Save("imgFinal.png");
```

INPUT IMAGE	OUTPUT IMAGE
	

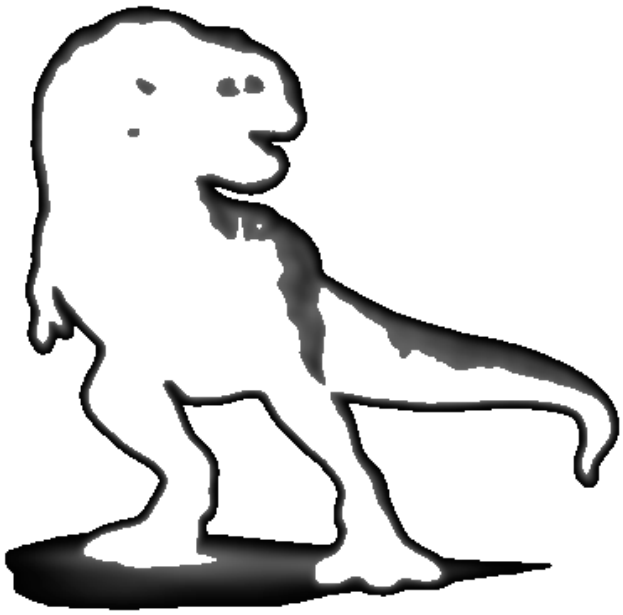
A. GRAY SCALE

B. INVERTING IMAGE



C. BLUR IMAGE

D. FINAL IMAGE




read.png

CODE OF PROGRAM

```
#include "Image.h"
#include <math.h>
#include <string>
#include <iostream>
#include <vector>
using namespace std;
int main() {
  GrayscaleImage imgInput, imgInputGr, imgNegative, imgBlur, imgFinal;
  imgInput.Load("read.png");
  imgInputGr = GrayscaleImage(imgInput);
  imgInputGr.Save("Grayscale.png");
  imgNegative = imgInput;
  for (int y = 0; y < imgNegative.GetHeight(); y++)
  {
    for (int x = 0; x < imgNegative.GetWidth(); x++)
    {
      imgNegative(x, y) = 255 - imgInput(x, y);
    }
  }
  imgNegative.Save("imgNegative.png");
  imgBlur = imgNegative;
  auto getValues = [&](int x, int y) { return imgNegative(x > -1 && x < imgNegative.GetWidth() ?
x : 1, y > -1 && x < imgNegative.GetHeight() ? y : 1);};
  for (int i = 0; i < 10; i++)
  {
    for (int y = 0; y < imgBlur.GetHeight() - 1; y++)
    {
      for (int x = 0; x < imgBlur.GetWidth() - 1; x++)
      {
        imgBlur(x, y) = (getValues(x - 1, y - 1) + getValues(x, y - 1) + getValues(x + 1, y - 1)
+
        getValues(x - 1, y) + getValues(x, y) + getValues(x + 1, y) +
        getValues(x - 1, y + 1) + getValues(x, y + 1) + getValues(x + 1, y + 1)) / 9;
      }
    }
    imgNegative = imgBlur;
  }
  imgBlur.Save("imgBlur.png");
  imgFinal = imgInput;
  for (int y = 0; y < imgFinal.GetHeight(); y++)
  {
    for (int x = 0; x < imgFinal.GetWidth(); x++)
    {
      if (imgBlur(x, y) >= 0 && imgBlur(x, y) < 10 )
      {
        imgFinal(x, y) = 255;
      }
      else if (imgBlur(x, y) >= 100 && imgBlur(x,y) < 255 )
      {
        imgFinal(x, y) = 255;
      }
      else
      {
        imgFinal(x, y) = imgBlur(x, y);
      }
    }
  }
  imgFinal.Save("imgFinal.png");
  return 0;
}
```