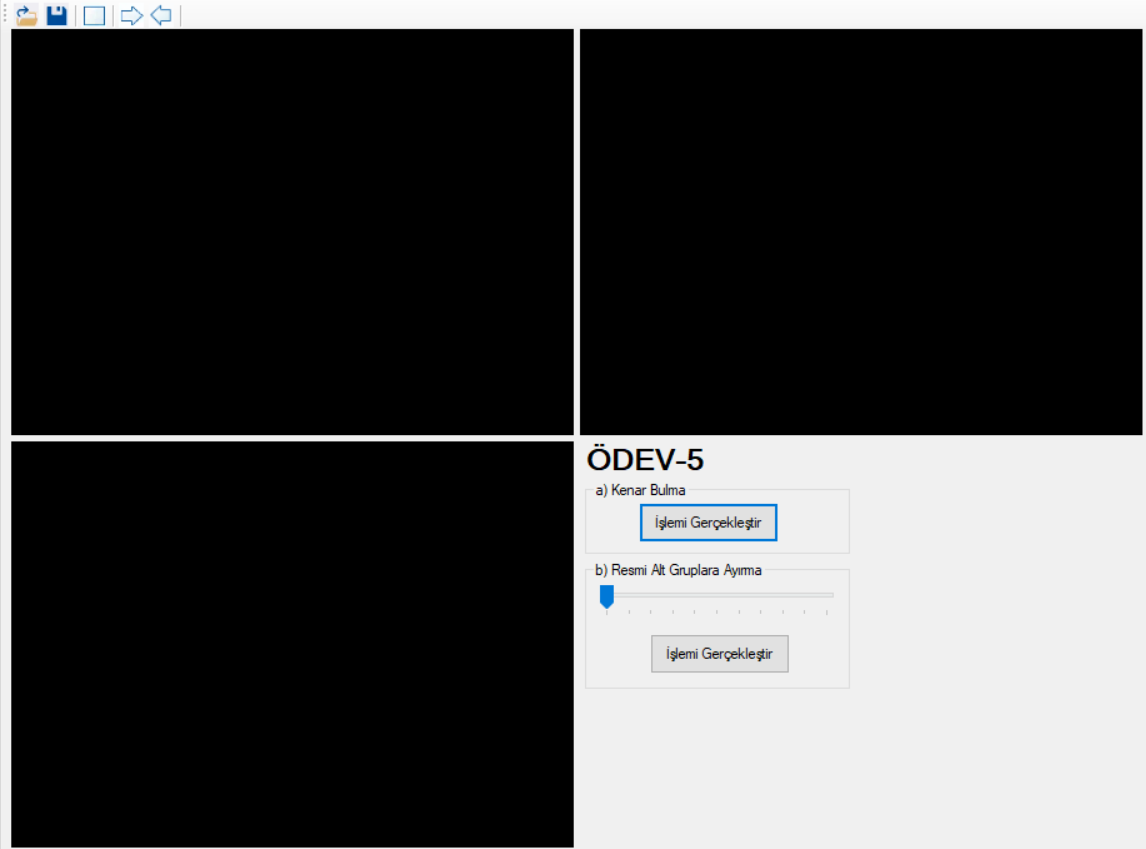
|  |  |  |
| --- | --- | --- |
| **AD SOYAD** | **CENGİZHAN TOPÇU** | **KARABÜK ÜNİVERSİTESİ MÜHENDİSLİK FAKÜLTESİ**  **MEKATRONİK MÜHENDİSLİĞİ** |
| **NUMARA** | **2017010225048** |
| **GÖRÜNTÜ İŞLEME ÖDEV-5** | |



**using** System**;**

**using** System**.**Collections**.**Generic**;**

**using** System**.**ComponentModel**;**

**using** System**.**Data**;**

**using** System**.**Drawing**;**

**using** System**.**IO**;**

**using** System**.**Linq**;**

**using** System**.**Text**;**

**using** System**.**Threading**.**Tasks**;**

**using** System**.**Windows**.**Forms**;**

**namespace** GoruntuIsleme\_DersNotlari\_Ödev5

**{**

**public** **partial** class Form1 **:** Form

**{**

**public** Form1**()**

**{**

InitializeComponent**();**

**}**

//Resim Yükle

**private** void BtnAc\_Click**(object** sender**,** EventArgs e**)**

**{**

**try**

**{**

openFileDialog1**.**DefaultExt **=** ".jpg"**;**

openFileDialog1**.**Filter **=** "Image Files(\*.BMP;\*.JPG;\*.GIF;\*.PNG)|\*.BMP;\*.JPG;\*.GIF;\*.PNG|All files (\*.\*)|\*.\*"**;**

openFileDialog1**.**ShowDialog**();**

String ResminYolu **=** openFileDialog1**.**FileName**;**

pictureBox1**.**Image **=** Image**.**FromFile**(**ResminYolu**);**

**}**

**catch**

**{**

MessageBox**.**Show**(**"HATA! Resim yüklenmedi"**);**

**}**

**}**

//Resim Kaydet

**private** void BtnKaydet\_Click**(object** sender**,** EventArgs e**)**

**{**

SaveFileDialog saveFileDialog1 **=** **new** SaveFileDialog**();**

saveFileDialog1**.**Filter **=** "Jpeg Resmi|\*.jpg|Bitmap Resmi|\*.bmp|Gif Resmi|\*.gif"**;**

saveFileDialog1**.**Title **=** "Resmi Kaydet"**;**

saveFileDialog1**.**ShowDialog**();**

**if** **(**saveFileDialog1**.**FileName **!=** ""**)** //Dosya adı boş değilse kaydedecek.

**{**

// FileStream nesnesi ile kayıtı gerçekleştirecek.

FileStream DosyaAkisi **=** **(**FileStream**)**saveFileDialog1**.**OpenFile**();**

**switch** **(**saveFileDialog1**.**FilterIndex**)**

**{**

**case** 1**:**

pictureBox2**.**Image**.**Save**(**DosyaAkisi**,** System**.**Drawing**.**Imaging**.**ImageFormat**.**Jpeg**);**

**break;**

**case** 2**:**

pictureBox2**.**Image**.**Save**(**DosyaAkisi**,** System**.**Drawing**.**Imaging**.**ImageFormat**.**Bmp**);**

**break;**

**case** 3**:**

pictureBox2**.**Image**.**Save**(**DosyaAkisi**,** System**.**Drawing**.**Imaging**.**ImageFormat**.**Gif**);**

**break;**

**}**

DosyaAkisi**.**Close**();**

**}**

**}**

//Resim Boyutu Ayarlama

**private** void btnBoyutAyarla\_Click**(object** sender**,** EventArgs e**)**

**{**

pictureBox1**.**SizeMode **=** PictureBoxSizeMode**.**Zoom**;**

pictureBox2**.**SizeMode **=** PictureBoxSizeMode**.**Zoom**;**

pictureBox3**.**SizeMode **=** PictureBoxSizeMode**.**Zoom**;**

**}**

//Resmi Sağa Aktarma

**private** void btnSagaAktar\_Click**(object** sender**,** EventArgs e**)**

**{**

//pictureBox2.Image = pictureBox1.Image;

Color OkunanRenk**,** DonusenRenk**;**

int R **=** 0**,** G **=** 0**,** B **=** 0**;**

Bitmap GirisResmi**,** CikisResmi**;**

GirisResmi **=** **new** Bitmap**(**pictureBox1**.**Image**);**

int ResimGenisligi **=** GirisResmi**.**Width**;** //GirisResmi global tanımlandı. İçerisine görüntü yüklendi.

int ResimYuksekligi **=** GirisResmi**.**Height**;**

CikisResmi **=** **new** Bitmap**(**ResimGenisligi**,** ResimYuksekligi**);** //Cikis resmini oluşturuyor. Boyutları giriş resmi ile aynı olur. Tanımlaması globalde yapıldı.

**for** **(**int x **=** 0**;** x **<** ResimGenisligi**;** x**++)**

**{**

**for** **(**int y **=** 0**;** y **<** ResimYuksekligi**;** y**++)**

**{**

OkunanRenk **=** GirisResmi**.**GetPixel**(**x**,** y**);**

R **=** OkunanRenk**.**R**;**

G **=** OkunanRenk**.**G**;**

B **=** OkunanRenk**.**B**;**

//if (x>100 && x<200)

DonusenRenk **=** Color**.**FromArgb**(**R**,** G**,** B**);**

CikisResmi**.**SetPixel**(**x**,** y**,** DonusenRenk**);**

**}**

**}**

pictureBox2**.**Image **=** CikisResmi**;**

**}**

//Resmi Sola Aktarma

**private** void btnSolaAktar\_Click**(object** sender**,** EventArgs e**)**

**{**

Color OkunanRenk**,** DonusenRenk**;**

int R **=** 0**,** G **=** 0**,** B **=** 0**;**

Bitmap GirisResmi**,** CikisResmi**;**

GirisResmi **=** **new** Bitmap**(**pictureBox2**.**Image**);**

int ResimGenisligi **=** GirisResmi**.**Width**;** //GirisResmi global tanımlandı. İçerisine görüntü yüklendi.

int ResimYuksekligi **=** GirisResmi**.**Height**;**

CikisResmi **=** **new** Bitmap**(**ResimGenisligi**,** ResimYuksekligi**);** //Cikis resmini oluşturuyor. Boyutları giriş resmi ile aynı olur. Tanımlaması globalde yapıldı.

**for** **(**int x **=** 0**;** x **<** ResimGenisligi**;** x**++)**

**{**

**for** **(**int y **=** 0**;** y **<** ResimYuksekligi**;** y**++)**

**{**

OkunanRenk **=** GirisResmi**.**GetPixel**(**x**,** y**);**

R **=** OkunanRenk**.**R**;**

G **=** OkunanRenk**.**G**;**

B **=** OkunanRenk**.**B**;**

DonusenRenk **=** Color**.**FromArgb**(**R**,** G**,** B**);**

CikisResmi**.**SetPixel**(**x**,** y**,** DonusenRenk**);**

**}**

**}**

pictureBox1**.**Image **=** CikisResmi**;**

**}**

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**private** void button1\_Click**(object** sender**,** EventArgs e**)**

**{**

Bitmap GirisResmi**,** CikisResmi**;**

GirisResmi **=** **new** Bitmap**(**pictureBox1**.**Image**);**

int ResimGenisligi **=** GirisResmi**.**Width**;**

int ResimYuksekligi **=** GirisResmi**.**Height**;**

CikisResmi **=** **new** Bitmap**(**ResimGenisligi**,** ResimYuksekligi**);**

int SablonBoyutu **=** 3**;**

//int ElemanSayisi = SablonBoyutu \* SablonBoyutu;

int x**,** y**;**

Color Renk**;**

int P1**,** P2**,** P3**,** P4**,** P5**,** P6**,** P7**,** P8**,** P9**;**

**for** **(**x **=** **(**SablonBoyutu **-** 1**)** **/** 2**;** x **<** ResimGenisligi **-** **(**SablonBoyutu **-** 1**)** **/** 2**;** x**++)** //Resmi taramaya şablonun yarısı kadar dış kenarlardan içeride başlayacak ve bitirecek.

**{**

**for** **(**y **=** **(**SablonBoyutu **-** 1**)** **/** 2**;** y **<** ResimYuksekligi **-** **(**SablonBoyutu **-** 1**)** **/** 2**;** y**++)**

**{**

Renk **=** GirisResmi**.**GetPixel**(**x **-** 1**,** y **-** 1**);**

P1 **=** **(**Renk**.**R **+** Renk**.**G **+** Renk**.**B**)** **/** 3**;**

Renk **=** GirisResmi**.**GetPixel**(**x**,** y **-** 1**);**

P2 **=** **(**Renk**.**R **+** Renk**.**G **+** Renk**.**B**)** **/** 3**;**

Renk **=** GirisResmi**.**GetPixel**(**x **+** 1**,** y **-** 1**);**

P3 **=** **(**Renk**.**R **+** Renk**.**G **+** Renk**.**B**)** **/** 3**;**

Renk **=** GirisResmi**.**GetPixel**(**x **-** 1**,** y**);**

P4 **=** **(**Renk**.**R **+** Renk**.**G **+** Renk**.**B**)** **/** 3**;**

Renk **=** GirisResmi**.**GetPixel**(**x**,** y**);**

P5 **=** **(**Renk**.**R **+** Renk**.**G **+** Renk**.**B**)** **/** 3**;**

Renk **=** GirisResmi**.**GetPixel**(**x **+** 1**,** y**);**

P6 **=** **(**Renk**.**R **+** Renk**.**G **+** Renk**.**B**)** **/** 3**;**

Renk **=** GirisResmi**.**GetPixel**(**x **-** 1**,** y **+** 1**);**

P7 **=** **(**Renk**.**R **+** Renk**.**G **+** Renk**.**B**)** **/** 3**;**

Renk **=** GirisResmi**.**GetPixel**(**x**,** y **+** 1**);**

P8 **=** **(**Renk**.**R **+** Renk**.**G **+** Renk**.**B**)** **/** 3**;**

Renk **=** GirisResmi**.**GetPixel**(**x **+** 1**,** y **+** 1**);**

P9 **=** **(**Renk**.**R **+** Renk**.**G **+** Renk**.**B**)** **/** 3**;**

//Hesaplamayı yapan Sobel Temsili matrisi ve formülü.

int Gx **=** Math**.**Abs**(-**P1 **+** P3 **-** 2 **\*** P4 **+** 2 **\*** P6 **-** P7 **+** P9**);** //Dikey çizgiler

int Gy **=** Math**.**Abs**(**P1 **+** 2 **\*** P2 **+** P3 **-** P7 **-** 2 **\*** P8 **-** P9**);** //Yatay Çizgiler

//Renkler sınırların dışına çıktıysa, sınır değer alınacak. Negatif olamaz, formüllerde mutlak değer vardır.

**if** **(**Gx **>** 100**)** Gx **=** 255**;**

**else** Gx **=** 0**;**

**if** **(**Gy **>** 100**)** Gy **=** 255**;**

**else** Gy **=** 0**;**

int Gxy**=** Gx **+** Gy**;**

**if** **(**Gxy **>** 255**)**

Gxy **=** 255**;**

int TetaRadyan **=** 0**;**

**if** **(**Gy **!=** 0**)** TetaRadyan **=** Convert**.**ToInt32**(**Math**.**Atan**(**Gx **/** Gy**));**

**else** **if** **(**Gy **!=** 0**)** TetaRadyan **=** Convert**.**ToInt32**(**Math**.**Atan**(**Gx**));**

int TetaDerece **=** Convert**.**ToInt32**((**TetaRadyan **\*** 360**)** **/** **(**2 **\*** Math**.**PI**));**

**if** **(**TetaDerece **>=**0 **&&** TetaDerece **<**20**)**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**FromArgb**(**0**,** 0**,** 0**));**

**if** **(**TetaDerece **>=** 20 **&&** TetaDerece **<** 40**)**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**FromArgb**(**255**,** 255**,** 0**));**

**if** **(**TetaDerece **>=** 40 **&&** TetaDerece **<** 60**)**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**FromArgb**(**0**,** 0**,** 255**));**

**if** **(**TetaDerece **>=** 60 **&&** TetaDerece **<** 80**)**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**FromArgb**(**255**,** 255**,** 0**));**

**if** **(**TetaDerece **>=** 80 **&&** TetaDerece **<** 100**)**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**FromArgb**(**125**,** 0**,** 255**));**

**if** **(**TetaDerece **>=** 100 **&&** TetaDerece **<** 120**)**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**FromArgb**(**255**,** 0**,** 125**));**

**if** **(**TetaDerece **>=** 120 **&&** TetaDerece **<** 140**)**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**FromArgb**(**255**,** 0**,** 0**));**

**if** **(**TetaDerece **>=** 140 **&&** TetaDerece **<** 160**)**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**FromArgb**(**255**,** 255**,** 0**));**

**if** **(**TetaDerece **>=** 160 **&&** TetaDerece **<** 180**)**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**FromArgb**(**0**,** 255**,** 0**));**

**}**

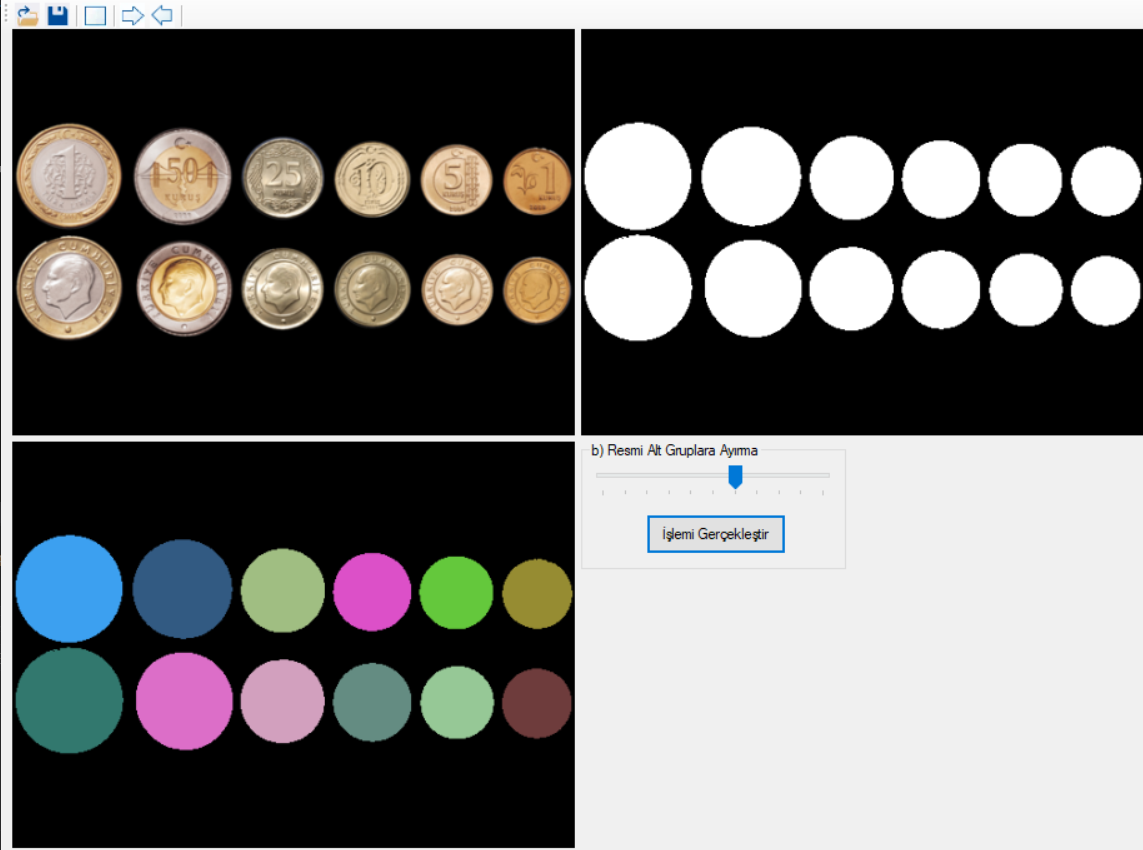
**}**

pictureBox2**.**Image **=** CikisResmi**;**

**}**

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu



//Resmi önce gri tona dönüştürüyor

**public** Bitmap ResmiGriTonaDonustur**(**Bitmap GirisResmi**)**

**{**

int R **=** 0**,** G **=** 0**,** B **=** 0**;**

Color OkunanRenk**,** DonusenRenk**;**

//Bitmap GirisResmi = new Bitmap(pictureBox1.Image);

int ResimGenisligi **=** GirisResmi**.**Width**;** //GirisResmi global tanımlandı. Fonksiyonla gelmedi.

int ResimYuksekligi **=** GirisResmi**.**Height**;**

//CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi, System.Drawing.Imaging.PixelFormat.Format8bppIndexed); //8 bir formatında gri-ton resim oluşturmak için.

Bitmap CikisResmi **=** **new** Bitmap**(**ResimGenisligi**,** ResimYuksekligi**);** //Cikis resmini oluşturuyor. Boyutları giriş resmi ile aynı olur.

int i **=** 0**,** j **=** 0**;** //Çıkış resminin x ve y si olacak.

**for** **(**int x **=** 0**;** x **<** ResimGenisligi**;** x**++)**

**{**

j **=** 0**;**

**for** **(**int y **=** 0**;** y **<** ResimYuksekligi**;** y**++)**

**{**

OkunanRenk **=** GirisResmi**.**GetPixel**(**x**,** y**);**

//int GriDegeri = (int)(OkunanRenk.R + OkunanRenk.G + OkunanRenk.B)/3; //Ortalama Gri-ton formülü

//int GriDegeri = Convert.ToInt16(OkunanRenk.R \* 0.299 + OkunanRenk.G \* 0.587 + OkunanRenk.B \* 0.114); //Gri-ton formülü

int GriDegeri **=** Convert**.**ToInt16**(**OkunanRenk**.**R **\*** 0.21 **+** OkunanRenk**.**G **\*** 0.71 **+** OkunanRenk**.**B **\*** 0.071**);** //Gri-ton formülü

R **=** GriDegeri**;**

G **=** GriDegeri**;**

B **=** GriDegeri**;**

DonusenRenk **=** Color**.**FromArgb**(**R**,** G**,** B**);**

CikisResmi**.**SetPixel**(**i**,** j**,** DonusenRenk**);**

j**++;**

**}**

i**++;**

**}**

**return** CikisResmi**;**

**}**

//Resmi 128 ile eşikleme siyah beyaz yapıyor

**public** Bitmap ResmiEsiklemeYap**(**Bitmap GirisResmi**)**

**{**

int R **=** 0**,** G **=** 0**,** B **=** 0**;**

Color OkunanRenk**;**

//Bitmap GirisResmi = new Bitmap(pictureBox1.Image);

int ResimGenisligi **=** GirisResmi**.**Width**;** //GirisResmi global tanımlandı.

int ResimYuksekligi **=** GirisResmi**.**Height**;**

Bitmap CikisResmi **=** **new** Bitmap**(**ResimGenisligi**,** ResimYuksekligi**);** //Cikis resmini oluşturuyor. Boyutları giriş resmi ile aynı olur.

//Eşik değerini trackBar'dan alacaktır.

int EsikDegeri **=** trackBar1**.**Value**;**

int i **=** 0**,** j **=** 0**;** //Çıkış resminin x ve y si olacak.

**for** **(**int x **=** 0**;** x **<** ResimGenisligi**;** x**++)**

**{**

j **=** 0**;**

**for** **(**int y **=** 0**;** y **<** ResimYuksekligi**;** y**++)**

**{**

OkunanRenk **=** GirisResmi**.**GetPixel**(**x**,** y**);**

**if** **(**OkunanRenk**.**R **>=** EsikDegeri**)**

R **=** 255**;**

**else**

R **=** 0**;**

**if** **(**OkunanRenk**.**G **>=** EsikDegeri**)**

G **=** 255**;**

**else**

G **=** 0**;**

**if** **(**OkunanRenk**.**B **>=** EsikDegeri**)**

B **=** 255**;**

**else**

B **=** 0**;**

Color DonusenRenk **=** Color**.**FromArgb**(**R**,** G**,** B**);**

CikisResmi**.**SetPixel**(**i**,** j**,** DonusenRenk**);**

j**++;**

**}**

i**++;**

**}**

**return** CikisResmi**;**

**}**

**private** void button2\_Click**(object** sender**,** EventArgs e**)**

**{**

Bitmap GirisResmi**,** CikisResmi**;**

int KomsularinEnKucukEtiketDegeri **=** 0**;**

GirisResmi **=** **new** Bitmap**(**pictureBox1**.**Image**);**

int ResimGenisligi **=** GirisResmi**.**Width**;**

int ResimYuksekligi **=** GirisResmi**.**Height**;**

int PikselSayisi **=** ResimGenisligi **\*** ResimYuksekligi**;**

GirisResmi **=** ResmiGriTonaDonustur**(**GirisResmi**);** //Resmi önce gri tona dönüştürüyor.

GirisResmi **=** ResmiEsiklemeYap**(**GirisResmi**);** //Resmi 128 ile eşikleme siyah beyaz yapıyor.

pictureBox2**.**Image **=** GirisResmi**;** //Resmin son halini gösteriyor.

CikisResmi **=** **new** Bitmap**(**ResimGenisligi**,** ResimYuksekligi**);**

int x**,** y**,** i**,** j**,** EtiketNo **=** 0**;**

int**[,]** EtiketNumarasi **=** **new** int**[**ResimGenisligi**,** ResimYuksekligi**];** //Resmin her pikselinin etiket numarası tutulacak.

//Tüm piksellerin Etiket numarasını başlangıçta 0 olarak atayacak. Siyah ve beyaz farketmez. Zaten ileride beyaz olanlara numara verilecek.

**for** **(**x **=** 0**;** x **<** ResimGenisligi**;** x**++)**

**{**

**for** **(**y **=** 0**;** y **<** ResimYuksekligi**;** y**++)**

**{**

EtiketNumarasi**[**x**,** y**]** **=** 0**;**

**}**

**}**

int IlkDeger **=** 0**,** SonDeger **=** 0**;**

bool DegisimVar **=** **false;** //Etiket numaralarında değişim olmayana kadar dönmesi için sonsuz döngüyü kontrol edecek.

**do** //etiket numaralarında değişim kalmayana kadar dönecek.

**{**

DegisimVar **=** **false;**

//------------------------- Resmi tarıyor ----------------------------

**for** **(**y **=** 1**;** y **<** ResimYuksekligi **-** 1**;** y**++)** //Resmin 1 piksel içerisinden başlayıp, bitirecek. Çünkü çekirdek şablon en dış kenardan başlamalı.

**{**

**for** **(**x **=** 1**;** x **<** ResimGenisligi **-** 1**;** x**++)**

**{**

//Resim siyah beyaz olduğu için tek kanala bakmak yeterli olacak. Sıradaki piksel beyaz ise işlem yap. Beyaz olduğu 255 yerine 128 kullanarak yapıldı.

**if** **(**GirisResmi**.**GetPixel**(**x**,** y**).**R **>** 128**)**

**{**

//işlem öncesi ele alınan pikselin etiket değerini okuyacak. İşlemler bittikten sonra bu değer değişirse, sonsuz döngü için işlem yapılmış demektir.

IlkDeger **=** EtiketNumarasi**[**x**,** y**];**

//Komşular arasında en küçük etiket numarasını bulacak.

KomsularinEnKucukEtiketDegeri **=** 0**;**

**for** **(**j **=** **-**1**;** j **<=** 1**;** j**++)** //Çekirdek şablon 3x3 lük bir matris. Dolayısı ile x,y nin -1 den başlayıp +1 ne kadar yer kaplar.

**{**

**for** **(**i **=** **-**1**;** i **<=** 1**;** i**++)**

**{**

**if** **(**EtiketNumarasi**[**x **+** i**,** y **+** j**]** **!=** 0 **&&** KomsularinEnKucukEtiketDegeri **==** 0**)** //hücrenin etiketi varsa ve daha hiç en küçük atanmadı ise ilk okuduğu bu değeri en küçük olarak atayacak.

**{**

KomsularinEnKucukEtiketDegeri **=** EtiketNumarasi**[**x **+** i**,** y **+** j**];**

**}**

**else** **if** **(**EtiketNumarasi**[**x **+** i**,** y **+** j**]** **<** KomsularinEnKucukEtiketDegeri **&&** EtiketNumarasi**[**x **+** i**,** y **+** j**]** **!=** 0 **&&** KomsularinEnKucukEtiketDegeri **!=** 0**)** //En küçük değer ve okunan hücreye etiket atanmışsa, içindeki değer en küçük değerden küçük ise o zaman en küçük o hücrenin değeri olmalıdır.

**{**

KomsularinEnKucukEtiketDegeri **=** EtiketNumarasi**[**x **+** i**,** y **+** j**];**

**}**

**}**

**}**

**if** **(**KomsularinEnKucukEtiketDegeri **!=** 0**)** //Beyaz komşu buldu ve içlerinde en küçük etiket değerine sahip numara da var. O zaman orta piksele o numarayı ata.

**{**

EtiketNumarasi**[**x**,** y**]** **=** KomsularinEnKucukEtiketDegeri**;**

**}**

**else** **if** **(**KomsularinEnKucukEtiketDegeri **==** 0**)** //Komşuların hiç birinde etiket numarası yoksa o zaman yeni bir numara ata

**{**

EtiketNo **=** EtiketNo **+** 1**;**

EtiketNumarasi**[**x**,** y**]** **=** EtiketNo**;**

**}**

SonDeger **=** EtiketNumarasi**[**x**,** y**];** //İşlem öncesi ve işlem sonrası değerler aynı ise ve butün piksellerde hep aynı olursa artık değişim yok demektir.

**if** **(**IlkDeger **!=** SonDeger**)**

DegisimVar **=** **true;**

**}**

**}**

**}**

**}** **while** **(**DegisimVar **==** **true);**

// Etiket değerine bağlı resmi renklendirecek-----------------------

// Önce etiket numaralarını diziye çekecek.

int**[]** DiziEtiket **=** **new** int**[**PikselSayisi**];**

i **=** 0**;**

**for** **(**x **=** 1**;** x **<** ResimGenisligi **-** 1**;** x**++)**

**{**

**for** **(**y **=** 1**;** y **<** ResimYuksekligi **-** 1**;** y**++)**

**{**

i**++;**

DiziEtiket**[**i**]** **=** EtiketNumarasi**[**x**,** y**];**

**}**

**}**

//Dizideki etiket numaralarını sıralıyor. Hazır fonksiyon kullanıyor.

Array**.**Sort**(**DiziEtiket**);**

//Tekrar eden etiket numaraarını çıkarıyor. Hazır fonksiyon kullanıyor.

int**[]** TekrarsizEtiketNumaralari **=** DiziEtiket**.**Distinct**().**ToArray**();**

int**[]** RenkDizisi **=** **new** int**[**TekrarsizEtiketNumaralari**.**Length**];**

**for** **(**j **=** 0**;** j **<** TekrarsizEtiketNumaralari**.**Length**;** j**++)**

**{**

RenkDizisi**[**j**]** **=** TekrarsizEtiketNumaralari**[**j**];** //sıradaki ilk renge, ait olacağı etiketin kaç numara olacağını atıyor.

**}**

int RenkSayisi **=** RenkDizisi**.**Length**;**

Color**[]** Renkler **=** **new** Color**[**RenkSayisi**];**

Random Rastgele **=** **new** Random**();**

int Kirmizi**,** Yesil**,** Mavi**;**

**for** **(**int r **=** 0**;** r **<** RenkSayisi**;** r**++)** //sonraki renkler.

**{**

Kirmizi **=** Rastgele**.**Next**(**5**,** 25**)** **\*** 10**;** //Açık renkler elde etmek ve 10 katları şeklinde olmasını sağlıyor. yani 150-250 arasındaki sayıları atıyor.

Yesil **=** Rastgele**.**Next**(**5**,** 25**)** **\*** 10**;**

Mavi **=** Rastgele**.**Next**(**5**,** 25**)** **\*** 10**;**

Renkler**[**r**]** **=** Color**.**FromArgb**(**Kirmizi**,** Yesil**,** Mavi**);**

**}**

//Color[] Renkler= { Color.Black, Color.Blue, Color.Red, Color.Orange, Color.LightPink, Color.LightYellow, Color.LimeGreen, Color.MediumPurple, Color.Olive, Color.Magenta, Color.Maroon, Color.AliceBlue, Color.AntiqueWhite, Color.Aqua, Color.LightBlue, Color.Azure, Color.White };

**for** **(**x **=** 1**;** x **<** ResimGenisligi **-** 1**;** x**++)** //Resmin 1 piksel içerisinden başlayıp, bitirecek. Çünkü çekirdek şablon en dış kenardan başlamalı.

**{**

**for** **(**y **=** 1**;** y **<** ResimYuksekligi **-** 1**;** y**++)**

**{**

int RenkSiraNo **=** Array**.**IndexOf**(**RenkDizisi**,** EtiketNumarasi**[**x**,** y**]);** //Dikkat: önemli bir komut. Dizinin değerinden sıra numarasını alıyor. int[] array = { 2, 3, 5, 7, 11, 13 }; int index = Array.IndexOf(array, 11); // returns 4

**if** **(**GirisResmi**.**GetPixel**(**x**,** y**).**R **<** 128**)** //Eğer bu pikselin rengi siyah ise aynı pikselin CikisResmi resmide siyah yapılacak.

**{**

CikisResmi**.**SetPixel**(**x**,** y**,** Color**.**Black**);**

**}**

**else**

**{**

CikisResmi**.**SetPixel**(**x**,** y**,** Renkler**[**RenkSiraNo**]);**

**}**

**}**

**}**

pictureBox3**.**Image **=** CikisResmi**;**

**}**

**}**

**}**