

Code Samples: The Chain Code with 4 Directions = Crack Code in C#

Copyright © by W. Kovalevski and V. Miszalok, last update: 01-05-2006

This code has been developed with Visual C#2.0.

Create a new Windows-project "chain_code". Delete the files `Form1.Designer.cs` and `Program.cs` from the project. Clear any prefabricated content from `Form1.cs` and replace it by the following code:

```
using System;
using System.Drawing;
using System.Windows.Forms;
using System.Collections;

public class Form1 : Form
{
    static void Main() { Application.Run( new Form1() ); }
    const Int32 xSize = 11;
    const Int32 ySize = 12;
    Byte[,] i0      = new Byte[ySize ,xSize ];
    Byte[,] clv     = new Byte[ySize ,xSize+1];
    Brush[] brush   = new Brush[10];
    Brush blackbrush = SystemBrushes.ControlText;
    Brush bluebrush  = new SolidBrush( Color.Blue );
    Pen redpen       = new Pen( Color.Red, 5 );
    Pen pinkpen      = new Pen( Color.FromArgb(255,128,128), 3 );
    Font arial10     = new Font( "Arial", 10 );
    Int32 i, x, y, dx, dy;
    Byte threshold = 1;
    Button[] button = new Button[ySize];
    Boolean NullCells;
    Point start = new Point();
    chaincode cc = new chaincode();
    ArrayList all_chaincodes = new ArrayList();

    class chaincode
    {
        public Point start;
        public String cracks;
        public Int32 perimeter, area;
    }

    public Form1()
    {
        BackColor = Color.White;
        Text      = "Chain Code";
        for ( i=0; i < 10; i++ )
            brush[i] = new SolidBrush(Color.FromArgb( i*25, i*25, i*25 ) );
        for ( y=0; y < ySize; y++ )
        {
            button[y] = new Button();
            Controls.Add(button[y]);
            button[y].BackColor = Color.Gray;
            button[y].Text = "nothing";
            button[y].Name = y.ToString();
            button[y].Click += new EventHandler( do_it );
        }
        button[0].Name = button[0].Text = "Homunculus";
        button[1].Name = "Threshold";
        button[2].Name = button[2].Text = "Noise";
        button[3].Name = button[3].Text = "Clear";
        button[4].Name = button[4].Text = "Code w/o 0-Cells";
        button[5].Name = button[5].Text = "Code with 0-Cells";
        button[1].Text = String.Format( "Threshold={0:#}", threshold );
        redpen.EndCap = System.Drawing.Drawing2D.LineCap.ArrowAnchor;
        pinkpen.EndCap = System.Drawing.Drawing2D.LineCap.ArrowAnchor;
        SetStyle( ControlStyles.ResizeRedraw, true );
        Width = 800;
        Height = 600;
    }
}
```

```

protected override void OnPaint(PaintEventArgs e)
{
    Graphics g = e.Graphics;
    Rectangle r = ClientRectangle;
    dx = r.Width / (xSize+2);
    dy = (r.Height - all_chaincodes.Count * FontHeight) / ySize;
    for ( y=0; y < ySize; y++ )
    {
        button[y].Top = y*dy+1;
        button[y].Left = xSize*dx+1;
        button[y].Width = r.Width - button[y].Left - 2;
        button[y].Height = dy-2;
    }
    Int32 textfieldY = button[ySize-1].Top + button[ySize-1].Height + 2;

    for ( y=0; y < ySize; y++ )
        for ( x=0; x < xSize; x++ )
            g.FillRectangle( brush[i0[y,x]], x*dx, y*dy, dx, dy );
    if ( all_chaincodes.Count == 0 ) return;
    for ( i=0; i < all_chaincodes.Count; i++ )
    {
        chaincode c = (chaincode)all_chaincodes[i];
        x = c.start.X;
        y = c.start.Y;
        for ( Int32 ii = 0; ii < c.perimeter; ii++ )
            switch ( c.cracks[ii] )
            {
                case 'e': g.DrawLine(redpen, x*dx, y*dy, (x+1)*dx, y*dy ); x++; break;
                case 's': g.DrawLine(redpen, x*dx, y*dy, x*dx, (y+1)*dy ); y++; break;
                case 'w': g.DrawLine(redpen, x*dx, y*dy, (x-1)*dx, y*dy ); x--; break;
                case 'n': g.DrawLine(redpen, x*dx, y*dy, x*dx, (y-1)*dy ); y--; break;
            }
        g.FillRectangle( bluebrush, x*dx-5, y*dy-5, 11, 11 );
        String s = "(" + c.start.X.ToString() + "/" +
            c.start.Y.ToString() + ")" +
            c.cracks + " P=" +
            c.perimeter.ToString() + " A=" +
            c.area.ToString();
        if ( c.area < 0 ) s+= " =negative";
        g.DrawString(s, arial10, blackbrush, 0, textfieldY );
        textfieldY += FontHeight;
    }
}

```

```

protected void do_it( object sender, System.EventArgs e )
{ switch( ((Button)sender).Name)
  { case "Homunculus"://*****
    i0 = new Byte[,] { {0,0,0,0,0,0,0,0,0,0},
                      {0,0,0,0,9,9,9,0,0,0},
                      {0,0,0,0,9,0,9,0,0,0},
                      {0,0,0,0,9,8,9,0,0,0},
                      {0,2,0,0,0,7,0,0,0,2},
                      {0,0,6,6,6,6,6,6,0,0},
                      {0,0,0,0,5,5,5,0,0,0},
                      {0,0,0,0,4,4,4,0,0,0},
                      {0,0,0,0,3,0,3,0,0,0},
                      {0,0,0,0,2,0,2,0,0,0},
                      {0,0,0,0,2,0,0,2,0,0},
                      {0,0,0,2,2,0,0,2,2,0} };
    all_chaincodes.Clear();
    break;
  case "Threshold"://*****
    if ( ++threshold > 9 ) threshold = 1;
    button[1].Text = "Threshold=" + threshold.ToString();
    all_chaincodes.Clear();
    break;
  case "Noise"://*****
    Random random = new Random();
    for ( y=0; y < ySize; y++ )
      for ( x=0; x < xSize; x++ )
        { Int32 noise = random.Next() % 3 - 1;//gives -1 or 0 or +1
          noise += i0[y,x]);//add former gray value
          if (noise < 0) i0[y,x] = 0;
          else if (noise > 9) i0[y,x] = 9;
          else i0[y,x] = (Byte)noise;
        }
    all_chaincodes.Clear();
    break;
  case "Clear"://*****
    for ( y=0; y < ySize; y++ )
      for ( x=0; x < xSize; x++ ) i0[y,x] = 0;
    threshold = 1; button[1].Text = "Threshold=1";
    all_chaincodes.Clear();
    break;
  case "Code w/o 0-Cells": NullCells = false; all_chaincodes.Clear();
                          Invalidate(); Application.DoEvents();
                          chaincode_finder(); break;
  case "Code with 0-Cells": NullCells = true; all_chaincodes.Clear();
                           Invalidate(); Application.DoEvents();
                           chaincode_finder(); break;
  }
  Invalidate();
}

```

```

private void chaincode_finder()
{
    SByte[,] Negative = new SByte[,] {{0,-1},{ 0,0},{-1, 0},{-1,-1}};
    SByte[,] Positive = new SByte[,] {{0, 0},{-1,0},{-1,-1},{ 0,-1}};
    Point Neg = new Point(), Pos = new Point();
    Boolean NEG, POS;
    int direction;
    for ( y=0; y < ySize; y++ )
        for ( x=0; x <= xSize; x++ ) clv[y,x] = 0;
    start.X = start.Y = 0;
    for ( Byte cc_no = 1; start_crack_search(); cc_no++ )
    {
        chaincode cc = new chaincode();
        System.Text.StringBuilder cracks = new System.Text.StringBuilder();
        cc.start = start; cracks.Append( 's' ); cc.area = 0;
        x = start.X;
        y = start.Y + 1;
        direction = 1;
        Graphics g = this.CreateGraphics();
        do
        {
            g.DrawLine( pinkpen, x*dx, y*dy, x*dx+dx/4, y*dy );
            g.DrawLine( pinkpen, x*dx, y*dy, x*dx, y*dy+dy/4 );
            g.DrawLine( pinkpen, x*dx, y*dy, x*dx-dx/4, y*dy );
            g.DrawLine( pinkpen, x*dx, y*dy, x*dx, y*dy-dy/4 );
            switch ( direction ) //last crack
            {
                case 0: g.DrawLine( redpen, (x-1)*dx, y*dy, x*dx, y*dy ); break;
                case 1: g.DrawLine( redpen, x*dx, (y-1)*dy, x*dx, y*dy ); break;
                case 2: g.DrawLine( redpen, (x+1)*dx, y*dy, x*dx, y*dy ); break;
                case 3: g.DrawLine( redpen, x*dx, (y+1)*dy, x*dx, y*dy ); break;
            }
            Int64 ticks = DateTime.Now.Ticks + 20000;
            do { } while ( ticks > DateTime.Now.Ticks );
            Neg.X = x + Negative[direction,0];
            Neg.Y = y + Negative[direction,1];
            Pos.X = x + Positive[direction,0];
            Pos.Y = y + Positive[direction,1];
            try { NEG = i0[Neg.Y,Neg.X] >= threshold; } catch { NEG = false; }
            try { POS = i0[Pos.Y,Pos.X] >= threshold; } catch { POS = false; }
            if ( POS && ( NEG || NullCells ) ) direction = (direction+1) % 4;
            else if ( !NEG && ( !POS || !NullCells ) ) direction = (direction+3) % 4;
            switch ( direction )
            {
                case 0: cracks.Append( 'e' ); x++; cc.area += y; break;
                case 1: cracks.Append( 's' ); y++; clv[y-1,x] = cc_no; break;
                case 2: cracks.Append( 'w' ); x--; cc.area -= y; break;
                case 3: cracks.Append( 'n' ); y--; clv[y ,x] = cc_no; break;
            }
        } while ( x != start.X || y != start.Y ); //end of do
        switch ( direction ) //draw the final crack
        {
            case 0: g.DrawLine( redpen, (x-1)*dx, y*dy, x*dx, y*dy ); break;
            case 2: g.DrawLine( redpen, (x+1)*dx, y*dy, x*dx, y*dy ); break;
        }
        cc.cracks = cracks.ToString();
        cc.perimeter = cc.cracks.Length;
        all_chaincodes.Add( cc );
    } // end of for
}

private Boolean start_crack_search()
{
    Byte left;
    for ( y=start.Y; y < ySize; y++ )
        for ( x=0; x < xSize; x++ )
        {
            if ( x > 0 ) left = i0[y,x-1]; else left = 0;
            if ( left < threshold && i0[y,x] >= threshold && clv[y,x] == 0 )
            {
                start.X = x; start.Y = y; clv[y,x] = 1; return true; }
        }
    return false;
}
}

```