

**LAMPIRAN A**  
***CODING PROGRAM***

```
unit Thinning;

interface

uses

  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, Menus, ExtCtrls, ExtDlgs, StdCtrls;

type

  TRGBTripleArray = array[0..1000] of RGBTriple;
  PRGBTripleArray = ^TRGBTripleArray;
  TPoints = array of TPoint;

  TForm1 = class(TForm)
    OpenPictureDialog1: TOpenPictureDialog;
    SavePictureDialog1: TSavePictureDialog;
    Image1: TImage;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Open1: TMenuItem;
    SaveAs1: TMenuItem;
    N1: TMenuItem;
    Animate1: TMenuItem;
    N2: TMenuItem;
```

```

Exit1: TMenuItem;

Button1: TButton;

Button2: TButton;

Image2: TImage;

StaticText1: TStaticText;

StaticText2: TStaticText;

procedure Open1Click(Sender: TObject);

procedure SaveAs1Click(Sender: TObject);

procedure Animate1Click(Sender: TObject);

procedure Exit1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure Skeletonize(var Points : TPoints ;
                      ALebar, ATinggi : integer);

procedure SkeletonizeAnimate(var Points : TPoints ;
                              ALebar, ATinggi : integer);

function Tetangga(ATetangga : byte) : TPoint;

procedure Button1Click(Sender: TObject);

private
  { Private declarations }
  UndoBitmap : TBitmap;

public
  { Public declarations }

end;

```

```

var
    Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.FormCreate(Sender: TObject);
begin
    UndoBitmap := TBitmap.Create;
    UndoBitmap.PixelFormat := pf24bit;
end;

procedure TForm1.Open1Click(Sender: TObject);
begin
    if not OpenPictureDialog1.Execute then Exit;
    Image1.Picture.Bitmap.LoadFromFile(OpenPictureDialog1.FileName);
    Image1.Picture.Bitmap.PixelFormat := pf24bit;
    Image1.AutoSize := True;
    Image2.Picture.Bitmap.LoadFromFile(OpenPictureDialog1.FileName);
    Image2.Picture.Bitmap.PixelFormat := pf24bit;
    Image2.AutoSize := True;
    Button1.Enabled := true;
end;

```

```
procedure TForm1.SaveAs1Click(Sender: TObject);
begin
    if not SavePictureDialog1.Execute then Exit;
    Image1.Picture.Bitmap.SaveToFile(SavePictureDialog1.FileName);
    Image1.Picture.Bitmap.PixelFormat := pf24bit;
end;
```

```
procedure TForm1.Animate1Click(Sender: TObject);
begin
    Animate1.Checked := not Animate1.Checked;
end;
```

```
procedure TForm1.Exit1Click(Sender: TObject);
begin
    Close;
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
begin
    Image1.Canvas.Draw(0, 0, UndoBitmap);
    Button2.Enabled := False;
    Button1.Enabled := True;
end;
```

```
function TForm1.Tetangga(ATetangga : byte) : TPoint;
```

```

begin
  ATetangga := (ATetangga - 2 ) mod 8 + 2; // struktur ketetanggaan
  Case ATetangga of
    1 : begin Result.X := 0; Result.Y := 0; end; //
    2 : begin Result.X := 0; Result.Y := -1; end; //
    3 : begin Result.X := 1; Result.Y := -1; end; // |
    4 : begin Result.X := 1; Result.Y := 0; end; //
    5 : begin Result.X := 1; Result.Y := 1; end; //
    6 : begin Result.X := 0; Result.Y := 1; end; //
    7 : begin Result.X := -1; Result.Y := 1; end; //
    8 : begin Result.X := -1; Result.Y := 0; end; //
    9 : begin Result.X := -1; Result.Y := -1; end; //
  end;
end;

procedure TForm1.Skeletonize(var Points : TPoints ;
  ALebar, ATinggi : integer);

Var
  Lx, Ly, index : integer;
  B, A, N, AN2, AN4 : byte;
  NOff : TPoint;
  Ubah : boolean;
  Hapus : array of boolean;
  Image : array of array of byte;
begin

```

```

// gambar diperbesar untuk tinggi dan lebarnya sebesar 2 pixel untuk
// menghindari access violation pada program
SetLength(Image, ALebar+2);      // membuat array sebesar image
for Lx := 0 to ALebar + 1 do begin
    SetLength(Image[Lx], ATinggi+2);
    for Ly := 0 to ATinggi + 1 do begin
        Image[Lx,Ly] := 0;
    end;
end;

for index := 0 to High(Points) do
    Image[Points[index].X+1][Points[index].Y+1] := 1; // array 2 dimensi
    diisi oleh Points dari image dengan nilai bit 1

    SetLength(Hapus, length(Points));      // membuat array dengan
    nama Hapus panjangnya disesuaikan dengan image

    Ubah := true;                          // memulai perubahan citra

    while Ubah do begin // memulai loop untuk perubahan dan berhenti
    apabila sudah tidak ada perubahan

        Ubah := false;

        // loop melalui seluruh citra biner

        for index := 0 to High(Points) do begin

            Hapus[index] := False; //array hapus di set false permulaan dan akan
            di set true apabila memenuhi 4 kriteria thinning

```

```
Lx := Points[index].X +1; //dari array yang telah di isi point dari  
image, X dan Y dideklarasikan pointnya +1
```

```
Ly := Points[index].Y +1;
```

```
// kriteria 1
```

```
// menghitung B
```

```
// B adalah jumlah tetangga yang bernilai bit 1
```

```
B := 0;
```

```
for N := 2 to 9 do begin
```

```
  NOff := Tetangga(N);
```

```
  Inc(B, Image[Lx+NOff.X][Ly+NOff.Y]); // B di increment  
  sebanyak jumlah tetangga yang bernilai bit 1
```

```
end;
```

```
if (B < 2) or (B > 6) then Continue; // jika B bernilai 2 sampai 6 maka  
lanjutkan
```

```
// kriteria 2
```

```
// menghitung A
```

```
// A adalah jumlah transisi bit 0->1 hanya boleh berjumlah 1  
[A(p1)=1]
```

```
A := 0;
```

```
for N := 2 to 9 do begin
```

```
  NOff := Tetangga(N);
```

```
  if Image[Lx+NOff.X][Ly+NOff.Y] = 0 then begin
```

```
    NOff := Tetangga(N+1);
```

```

    if Image[Lx+NOff.X][Ly+NOff.Y] = 1 then Inc(A);
end;
    if A > 1 then Break; // bila A sudah melebihi 1 maka (p) tersebut
tidak melanjutkan program ke kriteria selanjutnya
end;

```

```

    if A <> 1 then Continue; // bila A sama dengan dengan 1 maka lanjut
ke kriteria berikutnya

```

```

// kriteria 3

```

```

// Menghitung nilai A untuk p2

```

```

AN2 := 0;

```

```

for N := 2 to 9 do begin

```

```

    NOff := Tetangga(N);

```

```

    if Image[Lx+NOff.X][Ly+NOff.Y-1] = 0 then begin

```

```

        NOff := Tetangga(N+1);

```

```

        if Image[Lx+NOff.X][Ly+NOff.Y-1] = 1 then Inc(AN2);

```

```

    end;

```

```

end;

```

```

    if (Image[Lx][Ly-1] * Image[Lx+1][Ly] * Image[Lx-1][Ly]=0) or //
p2.p4.p8 = 0 atau p2 != 1

```

```

        (AN2 <> 1) then Continue;

```

```

// kriteria 4

```

```

// menghitung nilai A untuk p4

```



```

AN4 := 0;
for N := 2 to 9 do begin
  NOff := Tetangga(N);
  if Image[Lx+NOff.X+1][Ly+NOff.Y] = 0 then begin
    NOff := Tetangga(N+1);
    if Image[Lx+NOff.X+1][Ly+NOff.Y] = 1 then Inc(AN4);
  end;
end;

  if (Image[Lx][Ly-1] * Image[Lx+1][Ly] * Image[Lx][Ly+1] = 0) or
// p2.p4.p6 = 0 atau p4 != 1
  (AN4 <> 1) then begin

  Hapus[index] := True;
  Ubah := True;
end;
end;

// menghapus point yang telah diberi tanda pada array hapus
if Ubah then begin
  index := 0;
  while index < length(Points) do begin
    if Hapus[index] then begin
      // penghapusan array Hapus yang sudah diberi tanda
      Hapus[index] := Hapus[High(Hapus)];
    end;
  end;
end;

```

```

        SetLength(Hapus, High(Hapus));
        Points[index] := Points[High(Points)];
        SetLength(Points, High(Points));
        Image[Points[index].X+1][Points[index].Y+1] := 0;
    end
    else Inc(index);
    end;
end;
end;
end;

// program penganimasian
procedure TForm1.SkeletonizeAnimate(var Points : TPoints ;
                                   ALebar, ATinggi : integer);

Var
    Lx, Ly, index : integer;
    B, A, N, AN2, AN4 : byte;
    NOff : TPoint;
    Ubah : boolean;
    Hapus : array of boolean;
    Image : array of array of byte;
begin
    SetLength(Image, ALebar+2);
    for Lx := 0 to ALebar + 1 do begin
        SetLength(Image[Lx], ATinggi+2);
    end;
end;

```

```

for Ly := 0 to ATinggi + 1 do begin
    Image[Lx][Ly] := 0;
end;
end;
for index := 0 to High(Points) do
    Image[Points[index].X+1][Points[index].Y+1] := 1;
SetLength(Hapus, length(Points));
Ubah := true;

while Ubah do begin
    Ubah := false;

    for index := 0 to High(Points) do begin

        Hapus[index] := False;

        Lx := Points[index].X + 1;
        Ly := Points[index].Y + 1;

        B := 0;
        for N := 2 to 9 do begin
            NOff := Tetangga(N);
            Inc(B, Image[Lx+NOff.X][Ly+NOff.Y]);
        end;

```

if (B < 2) or (B > 6) then Continue;

A := 0;

for N := 2 to 9 do begin

    NOff := Tetangga(N);

    if Image[Lx+NOff.X][Ly+NOff.Y] = 0 then begin

        NOff := Tetangga(N+1);

        if Image[Lx+NOff.X][Ly+NOff.Y] = 1 then Inc(A);

    end;

    if A > 1 then Break;

end;

if A <> 1 then Continue;

AN2 := 0;

for N := 2 to 9 do begin

    NOff := Tetangga(N);

    if Image[Lx+NOff.X][Ly+NOff.Y+1] = 0 then begin

        NOff := Tetangga(N+1);

        if Image[Lx+NOff.X][Ly+NOff.Y+1] = 1 then Inc(AN2);

    end;

end;

and if (Image[Lx][Ly+1] \* Image[Lx+1][Ly] \* Image[Lx-1][Ly] <> 0)

(AN2 = 1) then Continue;

AN4 := 0;

for N := 2 to 9 do begin

NOff := Tetangga(N);

if Image[Lx+NOff.X+1][Ly+NOff.Y] = 0 then begin

NOff := Tetangga(N+1);

if Image[Lx+NOff.X+1][Ly+NOff.Y] = 1 then Inc(AN4);

end;

end;

if (Image[Lx][Ly-1] \* Image[Lx+1][Ly] \* Image[Lx][Ly+1] = 0) or

(AN4 <> 1) then begin

Hapus[index] := True;

Ubah := True;

end;

end;

if Ubah then begin

index := 0;

while index < length(Points) do begin

if Hapus[index] then begin

Image[Points[index].X+1][Points[index].Y+1] := 0;

```

        Points[index] := Points[High(Points)];
        SetLength(Points, High(Points));
        Hapus[index] := Hapus[High(Hapus)];
        SetLength(Hapus, High(Hapus));
    end
else Inc(index);
end;
end;

Image1.Canvas.FillRect(Image1.ClientRect);
for Ly := 0 to High(Points) do begin
    Image1.Canvas.Pixels[Points[Ly].X, Points[Ly].Y] := clblack;
end;

Image1.Refresh;
Sleep(200);
end;
end;

procedure TForm1.Button1Click(Sender: TObject);
var
    Scan : PRGBTripleArray;
    Lx, Ly : integer;
    Points : TPoints;
begin

```

```

UndoBitmap.Lebar := Image1.Lebar;
UndoBitmap.Tinggi := Image1.Tinggi;
UndoBitmap.Canvas.Draw(0, 0, Image1.Picture.Bitmap);
Button2.Enabled := True;
Button1.Enabled := False;

//mengambil array Points dari image
SetLength(Points, 0);
for Ly := 0 to Image1.Tinggi -1 do begin
  Scan := Image1.Picture.Bitmap.ScanLine[Ly];
  for Lx := 0 to Image1.Lebar -1 do begin
    if Scan[Lx].rgbtgreen = 0 then begin
      SetLength(Points, length(Points)+1 );
      Points[High(Points)] := Point(Lx, Ly);
    end;
  end;
end;

if Animate1.Checked then
  SkeletonizeAnimate(Points, Image1.Lebar, Image1.Tinggi) else
  Skeletonize(Points, Image1.Lebar, Image1.Tinggi);

Image1.Canvas.FillRect(Image1.ClientRect);
for Lx := 0 to High(Points) do begin
  Image1.Canvas.Pixels[Points[Lx].X, Points[Lx].Y] := clblack;
end;

```

end;

end;

end.