МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

(ФГБОУ ВО «ВятГУ»)

Факультет автоматики и вычислительной техники

Кафедра ЭВМ

Отчёт

Лабораторная работа № 1 по дисциплине

«Компьютерная графика»

«Реализация алгоритмов растровой развертки линий»

Выполнил студент группы ИВТб-2301-04-00 / Жеребцов К. А./

Проверил преподаватель / Коржавина А. С./

Киров 2021

**Цель работы:** закрепить лекционный материал по изучению базовых алгоритмов компьютерной графики – разложению отрезков и окружностей в растр.

Задание на лабораторную работу

1. Написать на языке Pascal программу, реализующую алгоритмы построения прямой: простой пошаговый алгоритм и алгоритмы Брезенхема для четырех- и восьмисвязной развертки;
2. Проверить правильность работы программы, нарисовав, например, каждым алгоритмом семейство радиальных прямых, выходящих из одной точки с шагом 15 градусов;
3. Написать и отладить программу, реализующую два алгоритма построения окружности: по формуле Y=+-SQRT(r\*r-x\*x) и Брезенхема. В обоих случаях использовать свойство симметрии окружности (в первом - найдя точки четверти окружности, остальные - отразив симметрично; во втором - свойство симметрии использовать полностью);

**Исходный код:**

Простой пошаговый алгоритм построения прямой:

unit Unit1;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls, ExtCtrls, math;

type

{ TForm1 }

TForm1 = class(TForm)

Button1: TButton;

Button2: TButton;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

PaintBox1: TPaintBox;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

private

public

end;

var

Form1: TForm1;

x1,x2,y1,y2,x,t1,t2,yy,y0: integer;

m,y,u,x0: real;

implementation

{$R \*.lfm}

{ TForm1 }

procedure TForm1.Button1Click(Sender: TObject);

begin

form1.Repaint;

Form1.Label5.Caption:='';

x1:= strtoint(Edit1.Text);

y1:= strtoint(Edit2.Text);

x2:= strtoint(Edit3.Text);

y2:= strtoint(Edit4.Text);

if x1>x2 then

begin

t1:=x1;

x1:=x2;

x2:=t1;

t2:=y1;

y1:=y2;

y2:=t2;

end;

if x1 <> x2 then

begin

m:=(y2-y1)/(x2-x1);

y:=y1;

for x:=x1 to x2 do

begin

sleep(1);

Form1.PaintBox1.Canvas.Pixels[x,Round(y)]:= ClBlack;

y:=y + m;

end;

end

else

if y1 = y2 then

begin

sleep(1);

Form1.PaintBox1.Canvas.Pixels[x1,y1]:=ClBlack;

end

else

Label5.Caption:='Вертикаль';

end;

procedure TForm1.Button2Click(Sender: TObject);

var

r:integer;

begin

form1.Repaint;

r:=450;

u:=0;

While u<2\*pi do

begin

x1:= 900;

y1:= 500;

x2:=x1+round(r\*cos(u));

y2:=y1-round(r\*sin(u));

if x1 <> x2 then

begin

m:=(y2-y1)/(x2-x1);

if (abs(m)<=1) then

begin

if x1>x2 then

begin

t1:=x1;

x1:=x2;

x2:=t1;

t2:=y1;

y1:=y2;

y2:=t2;

end;

y:=y1;

for x:=x1 to x2 do

begin

//sleep(1);

Form1.PaintBox1.Canvas.Pixels[x,Round(y)]:= ClBlack;

y:=y + m;

end;

end;

if (abs(m)>1) then

begin

if y1>y2 then

begin

t1:=x1;

x1:=x2;

x2:=t1;

t2:=y1;

y1:=y2;

y2:=t2;

end;

x0:=x1;

for y0:=y1 to y2 do

begin

Form1.PaintBox1.Canvas.Pixels[Round(x0),y0]:= ClBlack;

x0:=x0+1/m;

end;

end;

end

else

if y1 = y2 then

begin

//sleep(1);

Form1.PaintBox1.Canvas.Pixels[x1,y1]:=ClBlack;

end

else

if x1 = x2 then

begin

for yy:=y1+r downto y1-r do

begin

//sleep(1);

Form1.PaintBox1.Canvas.Pixels[x1,yy]:=ClBlack;

end;

end;

u:=u+pi/12;

end;

end;

end.

Алгоритм Брезенхема для построения прямой для 4-х связной развертки:

unit Unit1;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls, ExtCtrls;

type

{ TForm1 }

TForm1 = class(TForm)

Button1: TButton;

Button2: TButton;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

PaintBox1: TPaintBox;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure PaintBox1Click(Sender: TObject);

private

public

end;

var

Form1: TForm1;

x1,x2,y1,y2,x,y,dx,dy,s1,s2,i,e,t,r: integer;

f: boolean;

u:real;

implementation

{$R \*.lfm}

{ TForm1 }

function sign(a:integer):integer;

begin

if a < 0 then sign:=-1 else sign:= 1;

end;

procedure TForm1.Button1Click(Sender: TObject);

begin

repaint;

x1:=strtoint(Edit1.Text);

y1:=strtoint(Edit2.Text);

x2:=strtoint(Edit3.Text);

y2:=strtoint(Edit4.Text);

x:=x1;

y:=y1;

dx:=abs(x2-x1);

dy:=abs(y2-y1);

s1:=sign(x2-x1);

s2:=sign(y2-y1);

if dy < dx then f:= false

else

begin

t:=dx;

dx:=dy;

dy:=t;

f:=true;

end;

e:=2\*dy-dx;

for i:=1 to (dx+dy) do

begin

Form1.PaintBox1.Canvas.Pixels[x,y]:=ClBlack;

if e<0 then

begin

if f then y:=y+s2 else x:=x+s1;

e:=e+2\*dy;

end

else

begin

if f then x:=x+s1 else y:=y+s2;

e:=e-2\*dx;

end;

end;

Form1.PaintBox1.Canvas.Pixels[x,y]:=ClBlack;

end;

procedure TForm1.Button2Click(Sender: TObject);

begin

u:=0;

r:=200;

while u<2\*pi do

begin

x1:=900;

y1:=500;

x2:=x1+round(r\*cos(u));

y2:=y1-round(r\*sin(u));

x:=x1;

y:=y1;

dx:=abs(x2-x1);

dy:=abs(y2-y1);

s1:=sign(x2-x1);

s2:=sign(y2-y1);

if dy < dx then f:= false

else

begin

t:=dx;

dx:=dy;

dy:=t;

f:=true;

end;

e:=2\*dy-dx;

for i:=1 to (dx+dy) do

begin

Form1.PaintBox1.Canvas.Pixels[x,y]:=ClBlack;

if e<0 then

begin

if f then y:=y+s2 else x:=x+s1;

e:=e+2\*dy;

end

else

begin

if f then x:=x+s1 else y:=y+s2;

e:=e-2\*dx;

end;

end;

Form1.PaintBox1.Canvas.Pixels[x,y]:=ClBlack;

u:=u+pi/12;

end;

end;

end.

Алгоритм Брезенхема для построения прямой для 8-х связной развертки:

unit Unit1;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls, ExtCtrls;

type

{ TForm1 }

TForm1 = class(TForm)

Button1: TButton;

Button2: TButton;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

PaintBox1: TPaintBox;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

private

public

end;

var

Form1: TForm1;

x1,x2,y1,y2,x,y,dx,dy,s1,s2,temp,e,i,r: integer;

f: boolean;

u:real

implementation

{$R \*.lfm}

{ TForm1 }

function sign(a: integer):integer;

begin

if a < 0 then sign := -1;

if a >= 0 then sign := 1;

end;

procedure TForm1.Button1Click(Sender: TObject);

begin

repaint;

x1:= strtoint(Edit1.Text);

y1:= strtoint(Edit2.Text);

x2:= strtoint(Edit3.Text);

y2:= strtoint(Edit4.Text);

x:=x1;

y:=y1;

dx:=abs(x2-x1);

dy:=abs(y2-y1);

s1:=sign(x2-x1);

s2:=sign(y2-y1);

if dy > dx then

begin

temp:=dx;

dx:=dy;

dy:=temp;

f:=true;

end

else f:=false;

e:=2\*dy-dx;

for i:=1 to dx do

begin

Form1.PaintBox1.Canvas.Pixels[x,round(y)]:=ClBlack;

while e>=0 do

begin

if f then x:=x+s1 else y:=y+s2;

e:=e-2\*dx;

end;

if f then y:=y+s2 else x:=x+s1;

e:=e+2\*dy;

end;

Form1.PaintBox1.canvas.Pixels[x,round(y)]:=ClBlack;

end;

procedure TForm1.Button2Click(Sender: TObject);

begin

repaint;

u:=0;

r:=200;

while u<2\*pi do

begin

x1:=900;

y1:=500;

x2:=x1+round(r\*cos(u));

y2:=y1-round(r\*sin(u));

x:=x1;

y:=y1;

dx:=abs(x2-x1);

dy:=abs(y2-y1);

s1:=sign(x2-x1);

s2:=sign(y2-y1);

if dy > dx then

begin

temp:=dx;

dx:=dy;

dy:=temp;

f:=true;

end

else f:=false;

e:=2\*dy-dx;

for i:=1 to dx do

begin

Form1.PaintBox1.Canvas.Pixels[x,round(y)]:=ClBlack;

while e>=0 do

begin

if f then x:=x+s1 else y:=y+s2;

e:=e-2\*dx;

end;

if f then y:=y+s2 else x:=x+s1;

e:=e+2\*dy;

end;

Form1.PaintBox1.canvas.Pixels[x,round(y)]:=ClBlack;

u:=u+pi/12;

end;

end;

end.

Алгоритм Брезенхема для построения окружности и Простой алгоритм построения окружности по формуле:

unit Unit1;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls, ExtCtrls;

typ

{ TForm1 }

TForm1 = class(TForm)

Button1: TButton;

Button2: TButton;

Edit1: TEdit;

Label1: TLabel;

PaintBox1: TPaintBox;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

private

public

end;

var

Form1: TForm1;

x,Rad,e,y: Integer

implementation

{$R \*.lfm}

{ TForm1 }

procedure TForm1.Button1Click(Sender: TObject);

var

cx,cy: integer;

begin

Form1.PaintBox1.Repaint;

x:=0;

rad:=strtoint(Form1.Edit1.Text);

y:=rad;

e:=3-2\*rad;

cx:=900;

cy:=500;

while x<y do

begin

Form1.PaintBox1.Canvas.Pixels[x+cx,y+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[y+cx,x+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[y+cx,-x+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[x+cx,-y+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[-x+cx,-y+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[-y+cx,-x+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[-y+cx,x+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[-x+cx,y+cy]:=ClBlack;

if e<0 then

e:=e+4\*x+6

else

begin

e:=e+4\*(x-y)+10;

y:=y-1;

end;

x:=x+1;

end;

if x=y then

begin

Form1.PaintBox1.Canvas.Pixels[x+cx,y+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[y+cx,x+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[y+cx,-x+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[x+cx,-y+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[-x+cx,-y+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[-y+cx,-x+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[-y+cx,x+cy]:=ClBlack;

Form1.PaintBox1.Canvas.Pixels[-x+cx,y+cy]:=ClBlack;

end;

end;

procedure TForm1.Button2Click(Sender: TObject);

var xc,yc,x,y,d1,d2,r: integer;

begin

Form1.PaintBox1.Repaint;

xc:=900;

yc:=500;

x:=0;

r:=strtoint(Form1.Edit1.Text);

y:=r;

while ( x<=round(r/sqrt(2)) ) do begin

Paintbox1.Canvas.Pixels[xc+x,yc+y]:=clBlack;

Paintbox1.Canvas.Pixels[xc+x,yc-y]:=clBlack;

Paintbox1.Canvas.Pixels[xc-x,yc+y]:=clBlack;

Paintbox1.Canvas.Pixels[xc-x,yc-y]:=clBlack;

Paintbox1.Canvas.Pixels[xc+y,yc+x]:=clBlack;

Paintbox1.Canvas.Pixels[xc+y,yc-x]:=clBlack;

Paintbox1.Canvas.Pixels[xc-y,yc+x]:=clBlack;

Paintbox1.Canvas.Pixels[xc-y,yc-x]:=clBlack;

x:=x+1;

d1:=ABS(r\*r-x\*x-y\*y);

d2:=ABS(r\*r-x\*x-(y-1)\*(y-1));

if(d1>d2) then

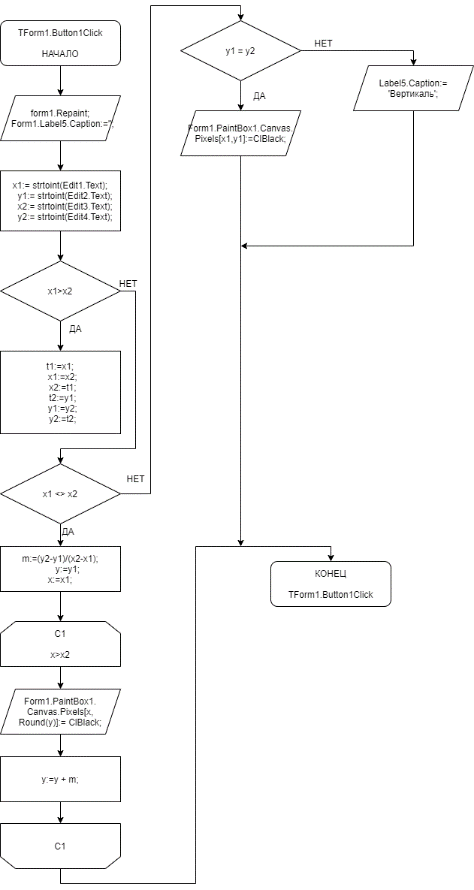
y:=y-1;

end;

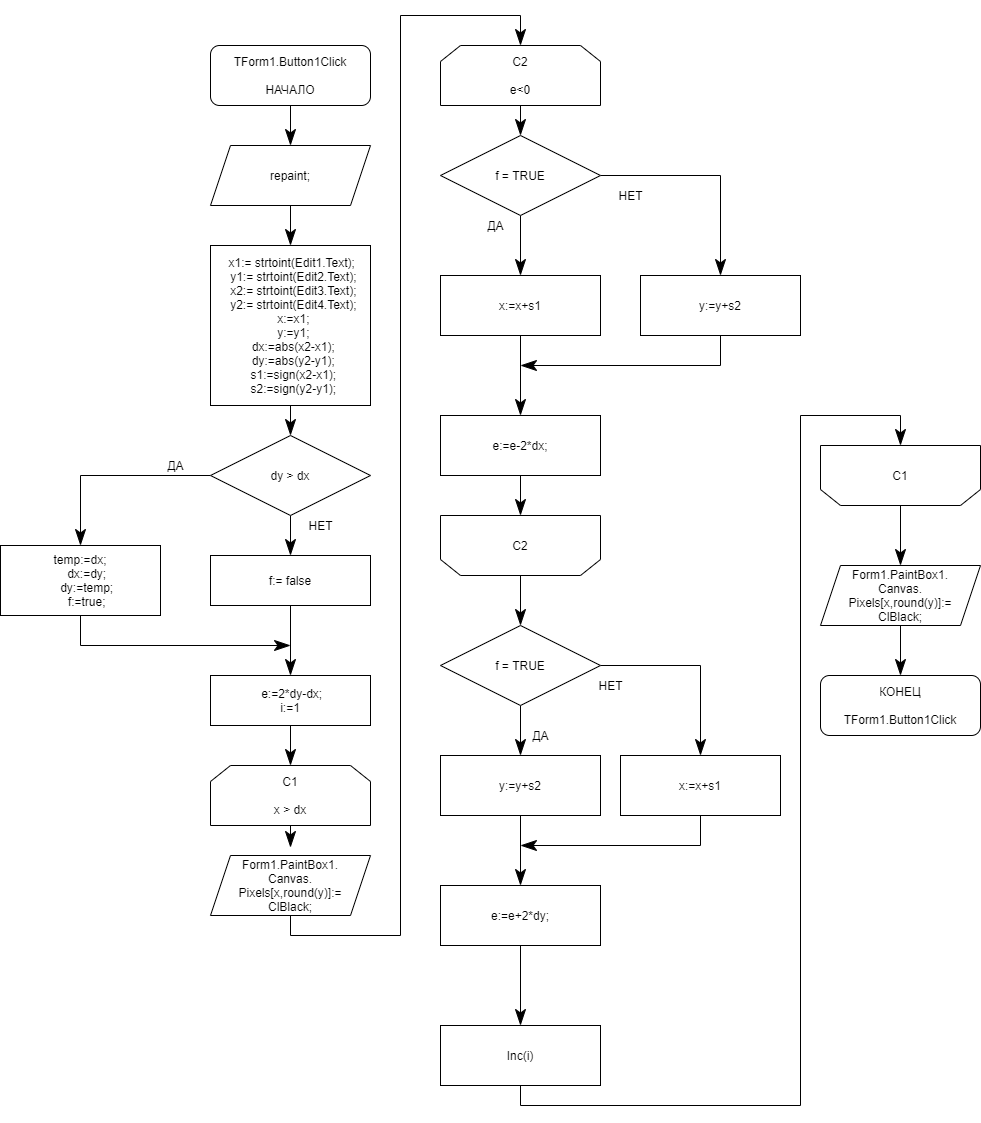
end;

end.

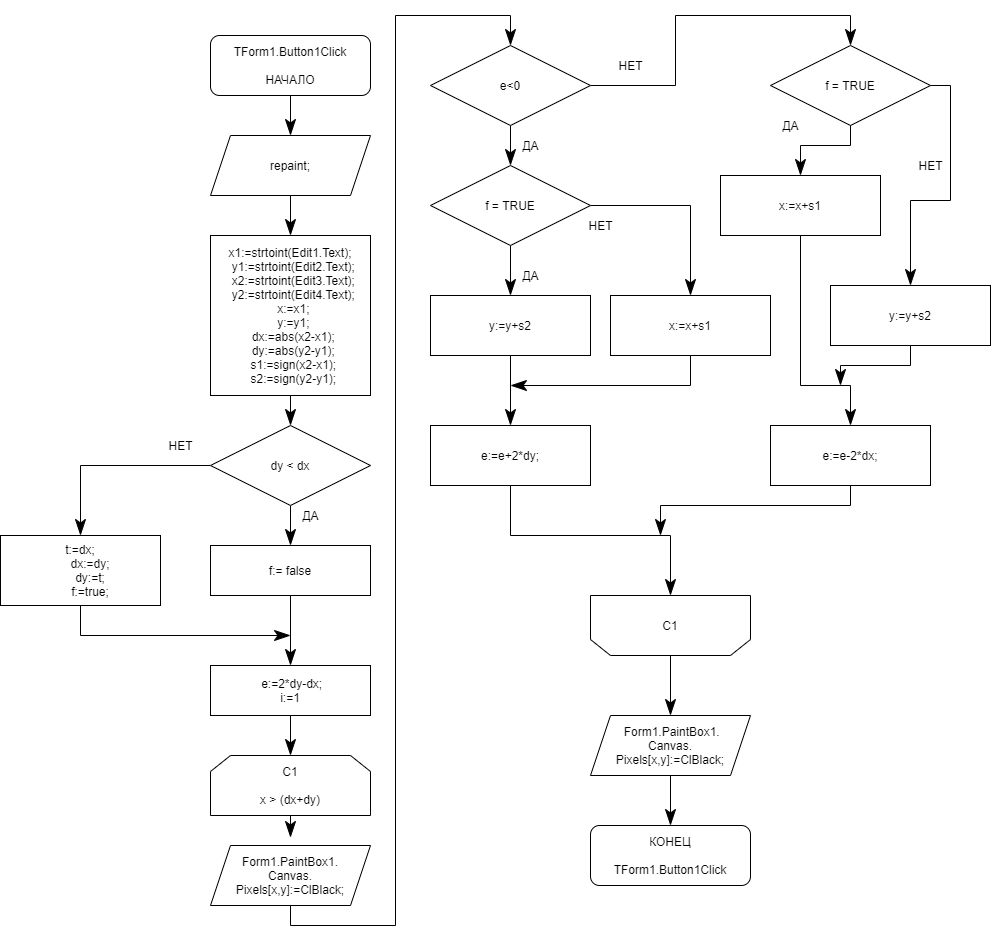
**Схемы алгоритмов:**

Прямая пошаговым алгоритмом

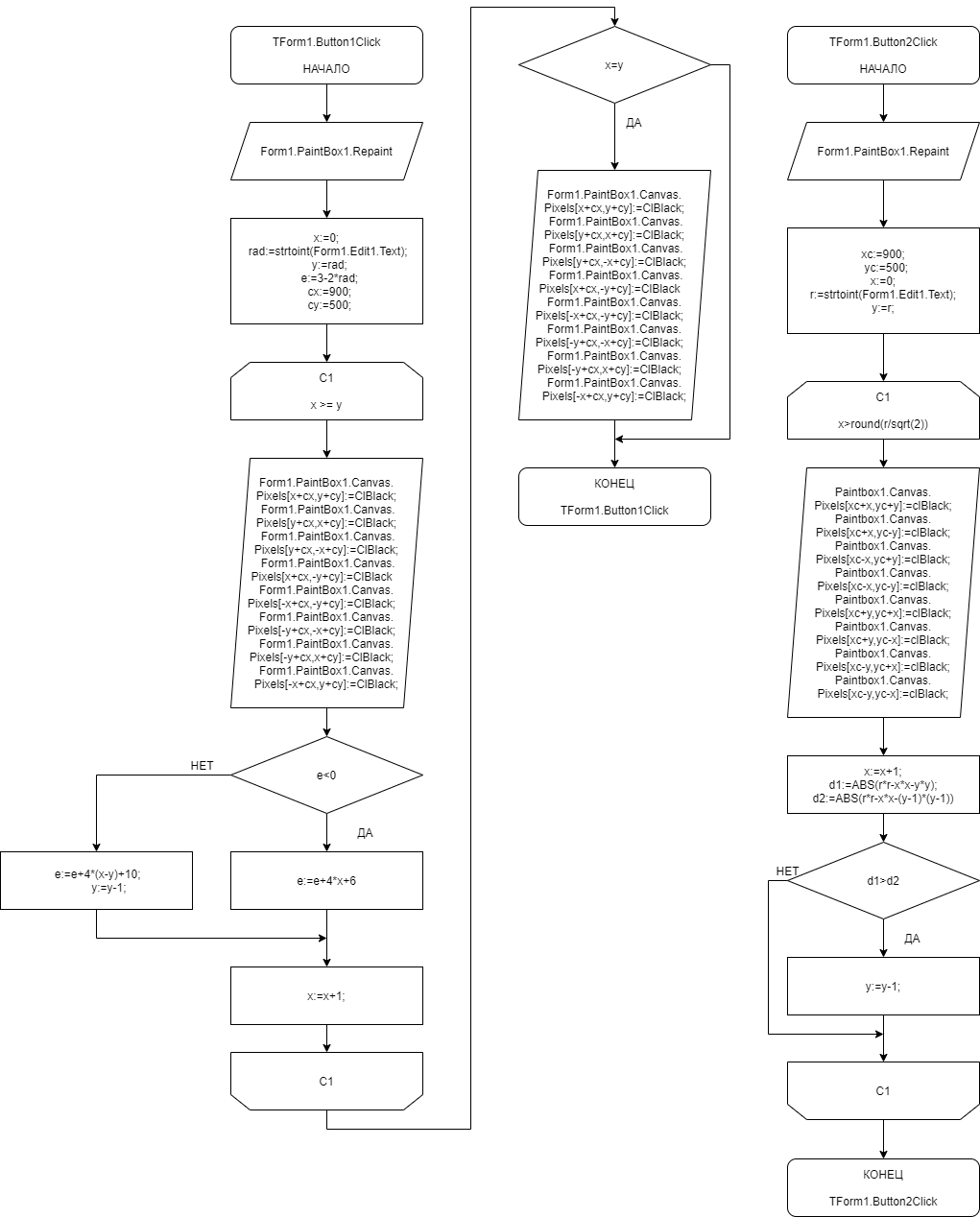
Алгоритм Брезенхема для восьмисвязной развертки



Алгоритм Брезенхема для четырехсвязной развертки

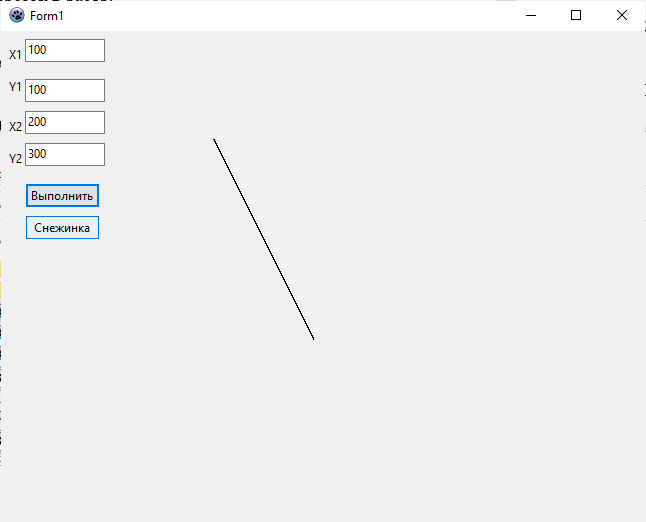


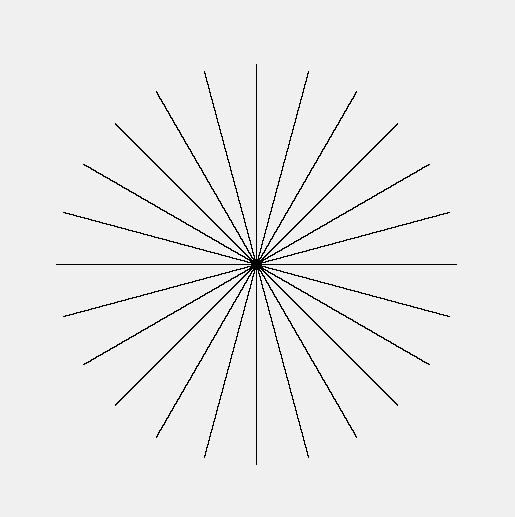
Алгоритм Брезенхема и простой алгоритм для окружности



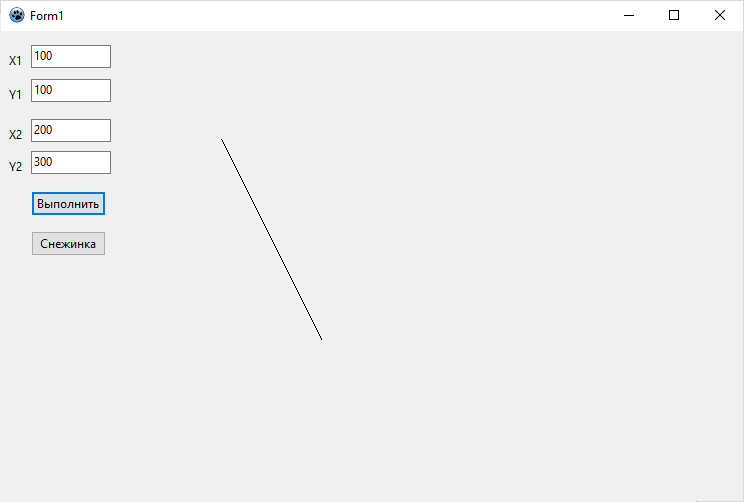
**Экранные формы:**

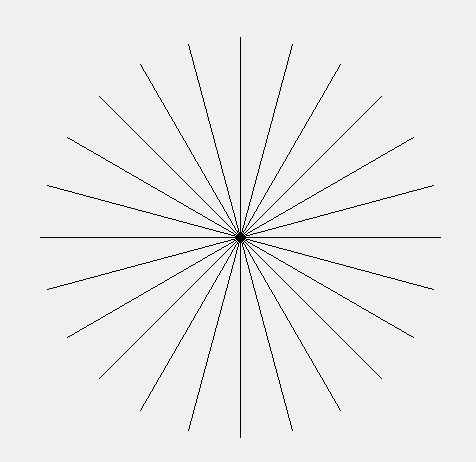
Алгоритм Брезенхема для четырехсвязной развертки:



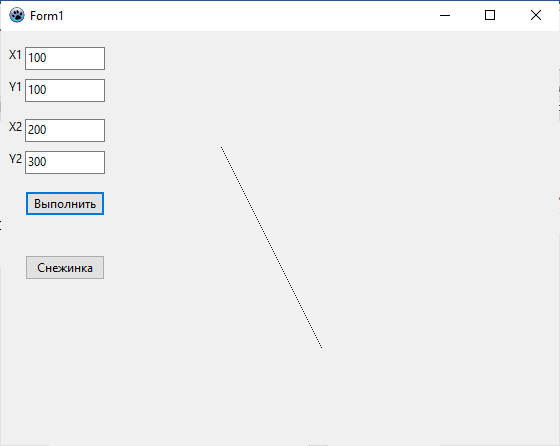


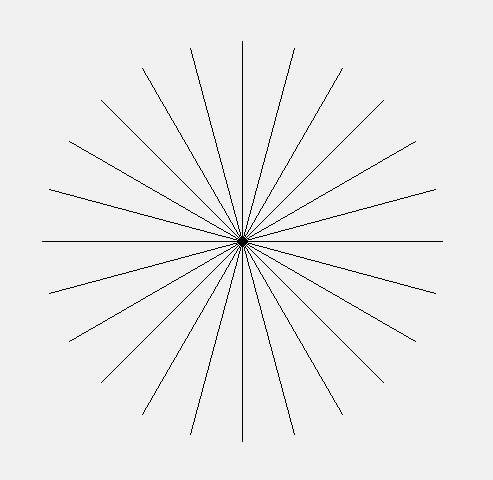
Алгоритм Брезенхема для восьмисвязной развертки:



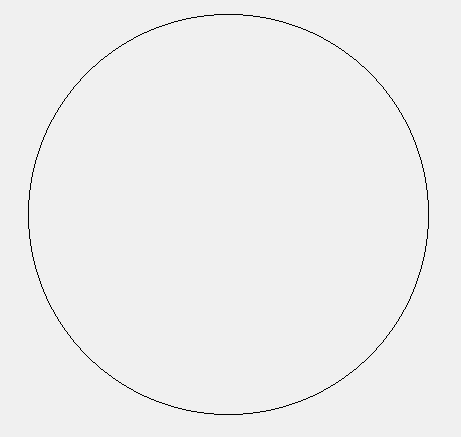


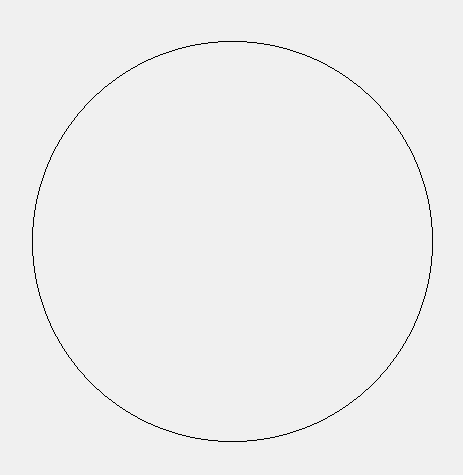
Простой алгоритм для прямой:





Алгоритм Брезенхема и простой алгоритм для окружности:





**Вывод:**

Был закреплен лекционный материал по изучению базовых алгоритмов компьютерной графики – разложению отрезков и окружностей в растр. Были написаны программы, реализующие простые алгоритмы и алгоритмы Брезенхема разложения в растр для прямой и окружности. Для проверки правильности алгоритмов разложения в растр для прямых были построены семейства радиальных прямых с шагом в 15 градусов.