**ST. Xavier's College**

**Maitighar, Kathmandu**

****

**Lab Assignment on Computer Graphics #5**

**Midpoint Circle Drawing Algorithm**

# 

# SUBMITTED BY

Shaurabh Chapagain

O13BSCCSIT037

**SUBMITTED TO**

|  |  |
| --- | --- |
| Er. Anil K. Sah  Lecturer  Department of Computer Science  St. Xavier’s College |  |

**Date of Submission: 21th August, 2015**

**Statement: To Implement Mid-point circle drawing algorithm in C++ builder.**

**Algorithm:**

1. Input the radius and circle centre (xc,yc) and obtain the first point on circle centers at origin as

(X0, y0)= (0, r).

2. Calculate initial decision parameter

P0=5/4-r

3. At each x, position starting at k=0 perform the tests

If p0<0 next point along the circle centre at (0, 0) is (xk+1,yk)

Pk+1=pk+2xk+1+1)

Otherwise the next point along circle is (xk+1,yk-1)

Pk+1=pk+2xk+1+1-2yk-1

4. Determine symmetry point on the other seven octants

5. Move each calculated pixels positions (x,y) in to circle path centered at (xc,yc) as

x= x+xc

y= y+yc

6. Repeat 3 through 5 until x>=y

**Program Code:**

//---------------------------------------------------------------------------

#include <vcl\vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma resource "\*.dfm"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

int xc,yc,r,x,y,p;

xc=StrToInt(Edit1->Text);

yc=StrToInt(Edit2->Text);

r=StrToInt(Edit3->Text);

Image1->Canvas->Pixels[xc+x][yc+y]=RGB(200,100,50);

Image1->Canvas->Pixels[xc-x][yc+y]=RGB(255,200,0);

Image1->Canvas->Pixels[xc+x][yc-y]=RGB(150,0,255);

Image1->Canvas->Pixels[xc-x][yc-y]=RGB(0,0,255);

Image1->Canvas->Pixels[xc+y][yc+x]=RGB(200,150,25);

Image1->Canvas->Pixels[xc-y][yc+x]=RGB(255,255,0);

Image1->Canvas->Pixels[xc+y][yc-x]=RGB(150,255,255);

Image1->Canvas->Pixels[xc-y][yc-x]=RGB(180,50,110);

x=0;

y=r;

p=1-r;

while(x<y)

{

if(p<0)

{

x=x+1;

p=p+2\*x+1;

}

else

{

x=x+1;

y=y-1;

p=p+2\*(x-y)+1;

}

Image1->Canvas->Pixels[xc+x][yc+y]=RGB(200,100,50);

Image1->Canvas->Pixels[xc-x][yc+y]=RGB(255,200,0);

Image1->Canvas->Pixels[xc+x][yc-y]=RGB(150,0,255);

Image1->Canvas->Pixels[xc-x][yc-y]=RGB(0,0,255);

Image1->Canvas->Pixels[xc+y][yc+x]=RGB(200,150,25);

Image1->Canvas->Pixels[xc-y][yc+x]=RGB(255,255,0);

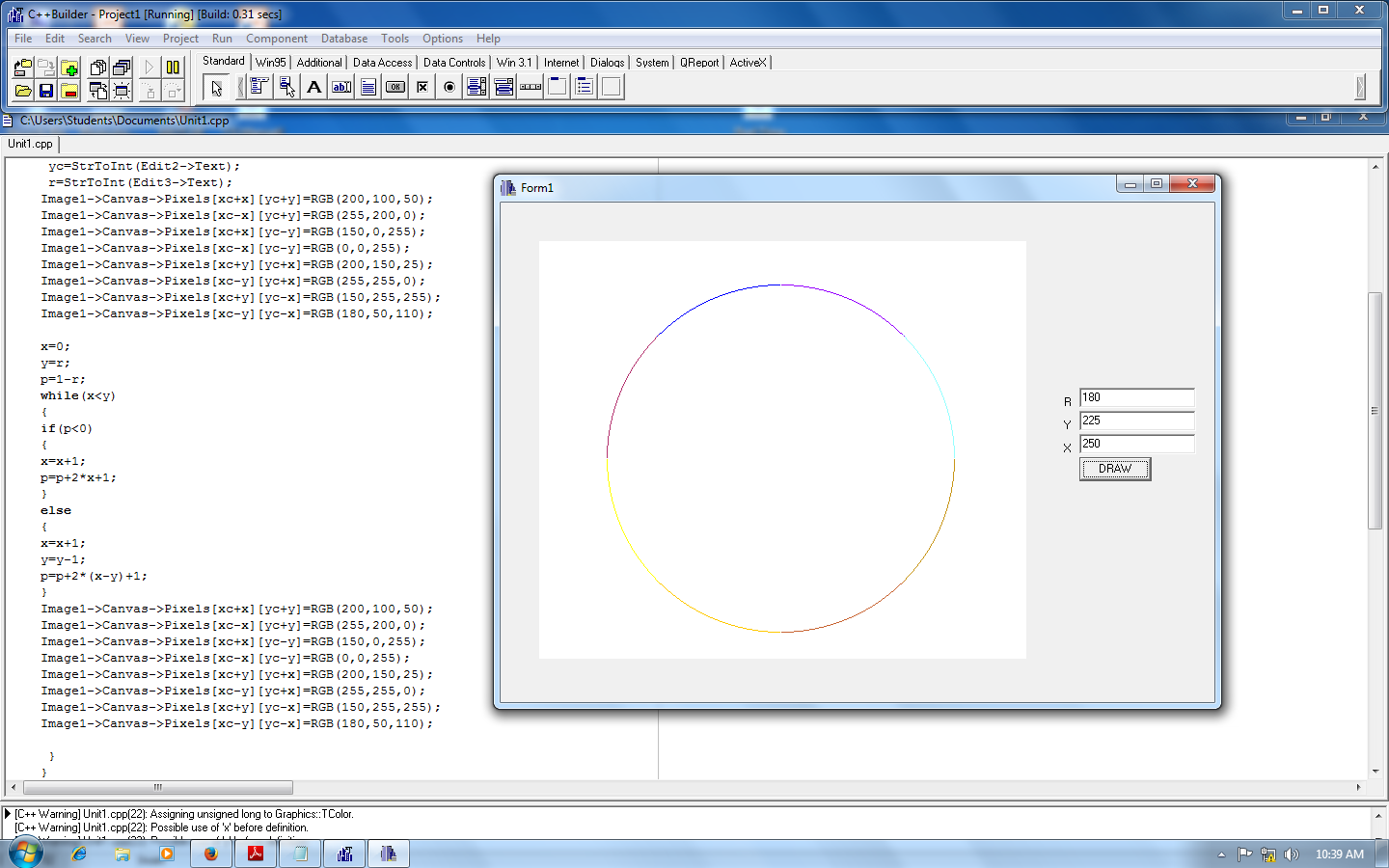
Image1->Canvas->Pixels[xc+y][yc-x]=RGB(150,255,255);

Image1->Canvas->Pixels[xc-y][yc-x]=RGB(180,50,110);

}

}

//---------------------------------------------------------------------------

**Output screen:** 

**Conclusion:**

Hence, Mid-point circle drawing algorithm is implemented using C++ builder.