ST. XAVIER’S COLLEGE

**Maitighar, Kathmandu**

**(Affiliated to Tribhuvan University)**



**Computer Graphics**

**Lab Assignment #5**

**DRAW A CIRCLE USING MIDPOINT ALGORITHM**

**Submitted By**

Alok Shrestha

013BSCIT005

B.Sc. CSIT Year II/IV Semester

**Submitted To**

Er. Anil Shah

Lecturer

Department of Computer Science

St. Xavier’s College

Maitighar, Kathmandu

**Submitted On**

August 2015

**STATEMENT**

DRAW A CIRCLE USING MIDPOINT ALGORITHM

**ALGORITHM**

1. Input radius r and circle center (xc,yc) and obtain the first point on the circumference of circle centered and the origin as (x0,y0)=(0,r)
2. Evaluate initial deciding parameter
3. For each xk position, starting at k=0
   * 1. If Pk< 0, then pixel to be plotted on boundary of circle centered (0,0) is (xk+1,yk) and deciding parameter Pk+1 = Pk + 2xk+1 + 1
     2. If Pk ≥ 0, then pixel to be plotted on boundary of circle centered (0,0) is (xk+1,yk-1) and deciding parameter Pk+1 = Pk + 2xk+1 - 2yk+1 +1
     3. Where 2xk+1 = 2xk + 2 and 2yk+1 = 2yk - 2
4. Determine symmetry points in other seven octants
5. Move each calculated pixel position (x,y) on the circular path centered on (xc,yc) and plot the coordinate values

x = x + xc , y = y + yc

1. Repeat step 3 to 5 until x ≥ y

**SOURCE CODE**

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

#include <vcl\vcl.h>

#pragma hdrstop

#include "Unit1.h"

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

#pragma resource "\*.dfm"

TForm1 \*Form1;

int xc, yc, r;

int p;

int x[150], y[150];

int xp, yp;

int i,j;

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

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

: TForm(Owner)

{

}

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

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

{

xc=StrToInt(Edit1->Text);

yc=StrToInt(Edit2->Text);

r=StrToInt(Edit3->Text);

x[0]=0;

y[0]=r;

p=1-r;

for(i=0;;i++)

{

x[i+1]=x[i]+1;

if(p<0)

{

y[i+1]=y[i];

p=p+2\*x[i+1]+1;

}

else

{

y[i+1]=y[i]-1;

p=p+2\*x[i+1]-2\*y[i+1]+1;

}

Image1->Canvas->Pixels[xc+x[i+1]][yc+y[i+1]]=RGB(0,0,0);

Image1->Canvas->Pixels[xc+x[i+1]][yc-y[i+1]]=RGB(0,0,255);

Image1->Canvas->Pixels[xc-x[i+1]][yc+y[i+1]]=RGB(0,255,0);

Image1->Canvas->Pixels[xc-x[i+1]][yc-y[i+1]]=RGB(0,255,255);

Image1->Canvas->Pixels[xc+y[i+1]][yc+x[i+1]]=RGB(255,0,0);

Image1->Canvas->Pixels[xc+y[i+1]][yc-x[i+1]]=RGB(255,0,255);

Image1->Canvas->Pixels[xc-y[i+1]][yc+x[i+1]]=RGB(255,255,0);

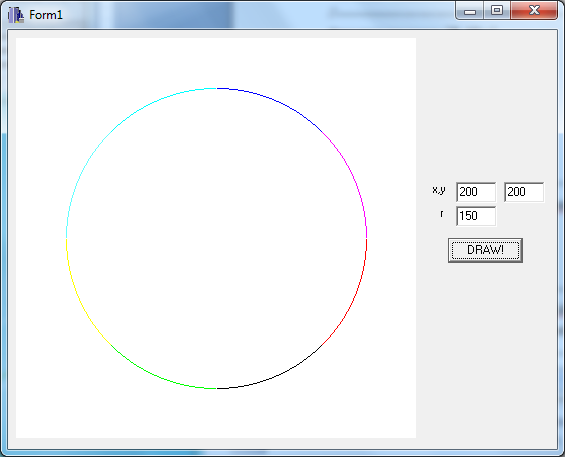
Image1->Canvas->Pixels[xc-y[i+1]][yc-x[i+1]]=RGB(100,255,255);

if(x[i+1]>=y[i+1]) break;

}

}

**OBSERVATION/S**



**CONCLUSION**

Hence a circle was designed using C++builder using mid-point algorithm.