**ST. XAVIER’S COLLEGE**

**(Affiliated to Tribhuvan University)**

Maitighar, Kathmandu



COMPUTER GRAPHICS

Lab Assignment #6

**Submitted by:**

Urendra Khakurel  
013BSCCSIT045

**Submitted to:**

Er. Anil K Sah

Lecturer  
Department of Computer Science

**Date of Submission: Aug 14 , 2015**

**STATEMENT: Implement midpoint ellipse algorithm in C++ builder**

**ALGORITHM:**

Step 1: input center (xc,yc) and rx and ry for the ellipse and obtain the first point as (x0,y0)=(0, ry)

Step 2: calculate initial decision parameter value in region 1 as

P10=

Step 3: at each xk­ position, in region1, starting at k=0, compute

xk+1=xk+1

if p1k=p1k+2xk+1+

yk+1=yk+1

otherwise next point to plot is

yk+1=yk-1

P1k+1=P1k+xk+1+-2yk+1 with xk+1=xk+1 and yk+1=yk-1

Step 4: calculate the initial value of decision parameter at region 2 using last calculated point say (x0,y0) in region 1 as

P20=

Step 5: at each yk position in region 2 starting at k=0, perform computation

yk+1=y-1

if P2k>0, then

xk+1=xk

P2k+1=P2k-

Otherwise

xk+1=xk+1

P2k+1=P2k+2 where xk+1=xk+1 and yk+1=yk+1

Step 6: determine the symmetry points in other three quadrants.

Step 7: move each calculated point (xk,yk) on to the centered (xc,yc) ellipse path as

xk=xk+xc

yk=yk+yc

Step 8: repeat the process for region 1 until and region 2 until (xk,yk)=(rx,0).

**SOURCE CODE:**

#include <vcl\vcl.h>

#pragma hdrstop

#include "Unit1.h"

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

#pragma resource "\*.dfm"

TForm1 \*Form1;

int xc,yc,rx,ry,x,y,p;

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

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

        : TForm(Owner)

{

}

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

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

{

xc=StrToInt(Edit1->Text);

yc=StrToInt(Edit2->Text);

rx=StrToInt(Edit3->Text);

ry=StrToInt(Edit4->Text);

x=0;

   y=ry;

   p=(ry\*ry)-(rx\*rx\*ry)+((rx\*rx)/4);

   while((2\*x\*ry\*ry)<(2\*y\*rx\*rx))

   {

        Image1->Canvas->Pixels[xc+x][yc-y]=RGB(0,125,150);

        Image1->Canvas->Pixels[xc-x][yc+y]=RGB(100,1,0);

        Image1->Canvas->Pixels[xc+x][yc+y]=RGB(0,125,150);

        Image1->Canvas->Pixels[xc-x][yc-y]=RGB(200,15,150);

        if(p<0)

        {

     x=x+1;

     p=p+(2\*ry\*ry\*x)+(ry\*ry);

        }

        else

        {

     x=x+1;

     y=y-1;

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

        }

   }

   p=((float)x+0.5)\*((float)x+0.5)\*ry\*ry+(y-1)\*(y-1)\*rx\*rx-rx\*rx\*ry\*ry;

         while(y>=0)

   {

        Image1->Canvas->Pixels[xc+x][yc-y]=RGB(0,125,150);

        Image1->Canvas->Pixels[xc-x][yc+y]=RGB(100,1,0);

        Image1->Canvas->Pixels[xc+x][yc+y]=RGB(0,125,150);

        Image1->Canvas->Pixels[xc-x][yc-y]=RGB(200,15,150);

        if(p>0)

        {

     y=y-1;

     p=p-(2\*rx\*rx\*y)+(rx\*rx);

        }

        else

        {

     y=y-1;

     x=x+1;

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

        }

  }

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

**OUTPUT SCREENSHOT**

