**ST. Xavier's College**

**Maitighar, Kathmandu**



**LAB ASSIGNMENT #6**

**Midpoint Ellipse Algorithm in C++ Builder**

# 

# SUBMITTED BY

# Binod Paneru

013BScCSIT014

**SUBMITTED TO**

Er. Anil K. Sah

Department of Computer Science

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

**ALGORITHM:**

1. Get parameters *a*, *b*, *h*, *k* for center coordinate *h* and *k* and major & minor axis lengths 2*a* and 2*b.*
2. Calculate the initial decision parameter value in the first region: http://geofhagopian.net/sablog/Slog-october/sablog-10-24-05_files/empty.gifhttp://geofhagopian.net/sablog/Slog-october/sablog-10-24-05_files/eq0023M.gifhttp://geofhagopian.net/sablog/Slog-october/sablog-10-24-05_files/empty.gif.
3. Use the formulas above to iterate *px*k+1 until *b*2*x*>*a*2*y*.
4. Rename the current (*x*k,*y*k) as (*x*0,*y*0) and calculate the initial decision parameter value in the 2nd region: http://geofhagopian.net/sablog/Slog-october/sablog-10-24-05_files/empty.gifhttp://geofhagopian.net/sablog/Slog-october/sablog-10-24-05_files/eq0024M.gifhttp://geofhagopian.net/sablog/Slog-october/sablog-10-24-05_files/empty.gif.
5. Use the formulas above to iterate *py*k+1 until *y* <= 0.
6. For both regions plot the other three symmetry points.

1. Shift to center at *h*, *k*.

**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::Edit1Change(TObject \*Sender)

{

xc=StrToInt(Edit1->Text);

}

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

void \_\_fastcall TForm1::Edit3Change(TObject \*Sender)

{

yc=StrToInt(Edit3->Text);

}

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

void \_\_fastcall TForm1::Edit2Change(TObject \*Sender)

{

rx=StrToInt(Edit2->Text);

}

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

void \_\_fastcall TForm1::Edit4Change(TObject \*Sender)

{

ry=StrToInt(Edit4->Text);

}

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

void \_\_fastcall TForm1::Go\_for\_EllipseClick(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(100,125,150);

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(100,125,150);

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);

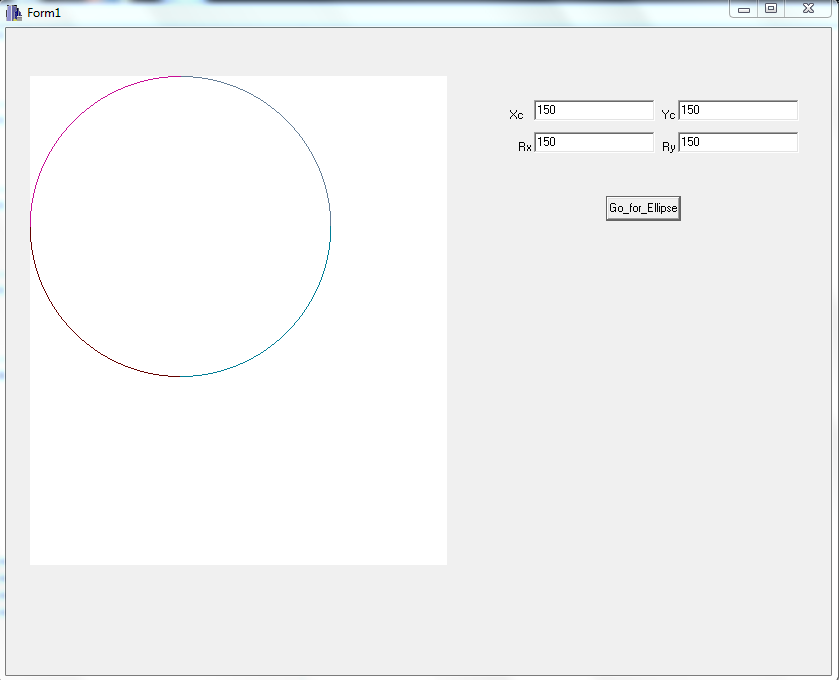
}

}

}

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

**OUTPUT SCREEN**

****

**CONCLUSION:**

The midpoint ellipse algorithm was implemented in C++ Builder to draw an ellipse as per the centre coordinates and axial radii input by the user. Different colors were used for arcs drawn using the symmetric property of ellipse.

**REFERENCE:**

<http://geofhagopian.net/sablog/Slog-october/sablog-10-24-05.htm>