**ST. Xavier's College**

**Maitighar, Kathmandu**



**LAB ASSIGNMENT #3**

**DDA line drawing algorithm in C++ Builder**

# 

# SUBMITTED BY

Arjun Neupane

O13BSCCSIT009

Sem IV

**SUBMITTED TO**

Er. Anil K. Sah

Lecturer,

Department of Computer Science

Date of submission: 14th August 14, 2015

**STATEMENT: Implement DDA line drawing algorithm in C++ builder**

**ALGORITHM:**

The DDA starts by calculating the smaller of dy or dx for a unit increment of the other. A line is then sampled at unit intervals in one coordinate and corresponding integer values nearest the line path are determined for the other coordinate.

Considering a line with positive slope, if the slope is less than or equal to 1, we sample at unit x intervals (dx=1) and compute successive y values as

y_{k+1} = y_k + m

Subscript k takes integer values starting from 0, for the 1st point and increases by 1 until endpoint is reached. y value is rounded off to nearest integer to correspond to a screen pixel.

For lines with slope greater than 1, we reverse the role of x and y i.e. we sample at dy=1 and calculate consecutive x values as

x_{k+1} = x_k + \frac{1}{m}

Similar calculations are carried out to determine pixel positions along a line with negative slope. Thus, if the absolute value of the slope is less than 1, we set dx=1 if  x_{start}<x_{end} i.e. the starting extreme point is at the left.

**SOURCE CODE:**

#include <vcl\vcl.h>

#pragma hdrstop

#include "Unit1.h"

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

#pragma resource "\*.dfm"

TForm1 \*Form1;

int x1,y1,x2,y2;

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

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

: TForm(Owner)

{

}

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

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

{

x1=StrToInt(Edit1->Text);

y1=StrToInt(Edit2->Text);

x2=StrToInt(Edit3->Text);

y2=StrToInt(Edit4->Text);

int dx, dy, steps, k;

float incrx, incry, x, y;

dx = x2 - x1;

dy = y2 - y1;

if ( abs(dx) > abs(dy) )

steps = abs(dx);

else

steps = abs(dy);

incrx = dx/steps;

incry = dy/steps;

x = x1;

y = y1;

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

for ( k = 1; k <= steps; k++ )

{

x = x + incrx;

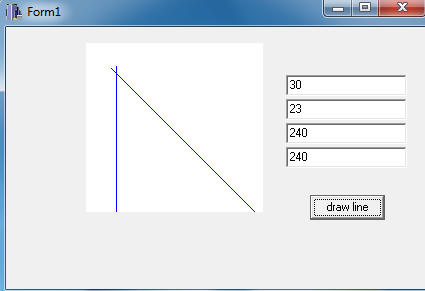
y = y + incry;

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

}

}

**OUTPUT:**

****