Experiment No : 4

Aim : Using DDA algorithm draw a house.

Theory :
Digital Differential Analyzer
The digital differential analyzer (DDA) is a scan-conversion line algorithm. A line is sampled at unit intervals in one coordinate and the corresponding integer values nearest the line path are determined for the other coordinate.
The DDA algorithm is a faster method for calculating pixel positions than one that directly implements the line equation. It eliminates the multiplication in the line equation by using raster characteristics, so that appropriate increments are applied in the x or y directions to step from one pixel position to another along the line path. The accumulation of round-off error in successive additions of the floating-point.

DDA Algorithm :

Consider start point of the line as (X0,Y0) and the end point of the line as (X1,Y1). Then
1. Calculate dx,dy using: dx = X1 – X0; dy = Y 1 – Y 0;
2. Depending upon absolute value of dx and dy choose number of steps to put pixel as:
steps = abs(dx) > abs(dy)?abs(dx) : abs(dy);
3. Calculate increment in x and y for each steps:
Xinc = dx/(float)steps;
Yinc = dy/(float)steps;
4. Using the above 3 steps, draw pixels as :

```
X = X0;
Y = Y0;
for (int i = 0; i <= steps; i++)
{
        putpixel (round(X),round(Y),WHITE);
        X += Xinc;
        Y += Yinc;
}
```

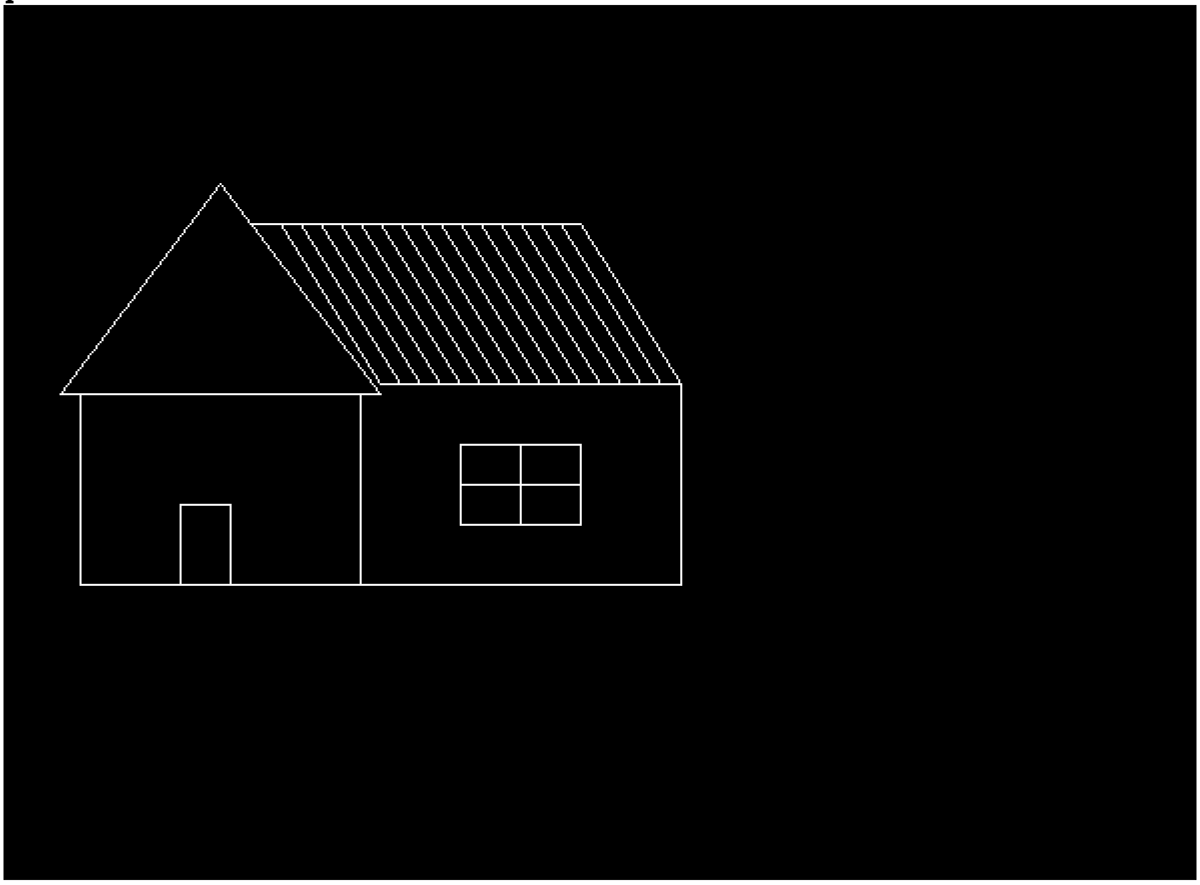
Code & Output :

```
#include<graphics.h>
#include<iostream>
#include<cmath>
using namespace std;

void positiveLeftToRight(int x1, int y1, int x2, int y2, float m);
void positiveRightToLeft(int x1, int y1, int x2, int y2, float m);
void negativeLeftToRight(int x1, int y1, int x2, int y2, float m);
void negativeRightToLeft(int x1, int y1, int x2, int y2, float m);
void DDA(int x1, int y1, int x2, int y2);

int main(){
   int gd=DETECT,gm;
   initgraph(&gd,&gm,NULL);

   DDA(40,205,120,100);
   DDA(120,100,200,205);
   DDA(40,205,200,205);


   DDA(50,205,50,300);
   DDA(190,205,190,300);
   DDA(50,300,190,300);


   DDA(100,260,100,300);
   DDA(100,260,125,260);
   DDA(125,260,125,300);

   DDA(135,120,300,120);
   DDA(300,120,350,200);
   DDA(200,200,350,200);

   DDA(290,120,340,200);
   DDA(280,120,330,200);
   DDA(270,120,320,200);
   DDA(260,120,310,200);
   DDA(250,120,300,200);
   DDA(240,120,290,200);
   DDA(230,120,280,200);
   DDA(220,120,270,200);
   DDA(210,120,260,200);
   DDA(200,120,250,200);
   DDA(190,120,240,200);
   DDA(180,120,230,200);
   DDA(170,120,220,200);
   DDA(160,120,210,200);
   DDA(150,120,200,200);

   DDA(350,200,350,300);
   DDA(190,300,350,300);


   DDA(240,230,300,230);
   DDA(240,230,240,270);
   DDA(240,270,300,270);
   DDA(300,230,300,270);
   DDA(270,230,270,270);
   DDA(240,250,300,250);
```

```
    delay(100000);
    closegraph();
    return 0;
}

void DDA(int x1, int y1, int x2, int y2){
float m;
m = float(y2 - y1)/float(x2 - x1);
if((m > 0.0) && (x1 < x2))
    positiveLeftToRight(x1,y1,x2,y2,m);
else if((m > 0.0) && (x1 > x2))
    positiveRightToLeft(x1,y1,x2,y2,m);
else if(((1/m)==0) && (y1 < y2))
    positiveLeftToRight(x1,y1,x2,y2,m);
else if(((1/m)==0) && (y1 > y2))
    negativeLeftToRight(x1,y1,x2,y2,m);
else if((m < 0.0) && (x1 < x2))
    negativeLeftToRight(x1,y1,x2,y2,m);
else if((m < 0.0) && (x1 > x2))
    negativeRightToLeft(x1,y1,x2,y2,m);
else if((m == 0) && (x1 < x2))
    positiveLeftToRight(x1,y1,x2,y2,m);
else if((m == 0) && (x1 > x2))
    positiveRightToLeft(x1,y1,x2,y2,m);

}

void positiveLeftToRight(int x1, int y1, int x2, int y2, float m){
float X = x1, Y = y1;
int plotX = x1, plotY = y1;
if(m <= 1){
    while(plotX <= x2){
        putpixel(plotX,plotY,WHITE);
        X = X + 1;
        Y = Y + m;
        plotX = X;
        plotY = round(Y);

    }
}
else if (m > 1){
    while(plotY <= y2){
        putpixel(plotX,plotY,WHITE);
        X = X + (1/m);
        Y = Y + 1;
        plotX = round(X);
        plotY = Y;

    }
}
}

void positiveRightToLeft(int x1, int y1, int x2, int y2, float m){
float X = x1, Y = y1;
int plotX = x1, plotY = y1;
if(m <= 1){
    while(plotX >= x2){
        putpixel(plotX,plotY,WHITE);
```

```
            X = X - 1;
            Y = Y - m;
            plotX = X;
            plotY = round(Y);

    }
}
else if (m > 1){
    while(plotY >= y2){
            putpixel(plotX,plotY,WHITE);
            X = X - (1/m);
            Y = Y - 1;
            plotX = round(X);
            plotY = Y;

    }
}
}

void negativeLeftToRight(int x1, int y1, int x2, int y2, float m){
float X = x1, Y = y1;
int plotX = x1, plotY = y1;
if(abs(m) <= 1){
    while(plotX <= x2){
            putpixel(plotX,plotY,WHITE);
            X = X + 1;
            Y = Y + m;
            plotX = X;
            plotY = round(Y);

    }
}
else if (abs(m) > 1){
    while(plotY >= y2){
            putpixel(plotX,plotY,WHITE);
            X = X - (1/m);
            Y = Y - 1;
            plotX = round(X);
            plotY = Y;

    }
}
}

void negativeRightToLeft(int x1, int y1, int x2, int y2, float m){
float X = x1, Y = y1;
int plotX = x1, plotY = y1;
if(abs(m) <= 1){
    while(plotX >= x2){
            putpixel(plotX,plotY,WHITE);
            X = X - 1;
            Y = Y - m;
            plotX = X;
            plotY = round(Y);

    }
}
else if (abs(m) > 1){
    while(plotY <= y2){
            putpixel(plotX,plotY,WHITE);
            X = X + (1/m);
            Y = Y + 1;
            plotX = round(X);
            plotY = Y;

    }
}

}
```

Output :



Conclusion : Program to draw a house using DDA was successfully written and executed

Deepraj Bhosale
181105016