EXTERNAL PRACTICAL EXPERIMENT

COMPUTER GRAPICS AND MULTIMEDIA

DDA Algorithm

Write a C program to draw a line using DDA algorithm.

Syeda Reeha Quasar
14114802719
Group – C6

EXTERNAL EXPERIMENT

AIM:

Write a C program to draw a line using DDA algorithm.

THEORY:

DDA stands for Digital Differential Analyzer. It is an incremental method of scan conversion of line. In this method calculation is performed at each step but by using results of previous steps.

Advantage:

1. It is a faster method than method of using direct use of line equation.
2. This method does not use multiplication theorem.
3. It allows us to detect the change in the value of x and y, so plotting of same point twice is not possible.
4. This method gives overflow indication when a point is repositioned.
5. It is an easy method because each step involves just two additions.

Disadvantage:

1. It involves floating point additions rounding off is done. Accumulations of round off error cause accumulation of error.
2. Rounding off operations and floating-point operations consumes a lot of time.
3. It is more suitable for generating line using the software. But it is less suited for hardware implementation.

DDA Algorithm:

Step1: Start Algorithm

Step2: Declare x_1,y_1,x_2,y_2,dx,dy,x,y as integer variables.

Step3: Enter value of x_1,y_1,x_2,y_2.

Step4: Calculate dx = x_2-x_1

Step5: Calculate dy = y_2-y_1

Step6: If ABS (dx) > ABS (dy)
Then step = abs (dx)
Else

Step7: x_{inc}=dx/step
y_{inc}=dy/step

assign $x = x_1$
assign $y = y_1$

Step8: Set pixel (x, y)

Step9: $x = x + x_{inc}$
$y = y + y_{inc}$
Set pixels (Round (x), Round (y))

Step10: Repeat step 9 until $x = x_2$

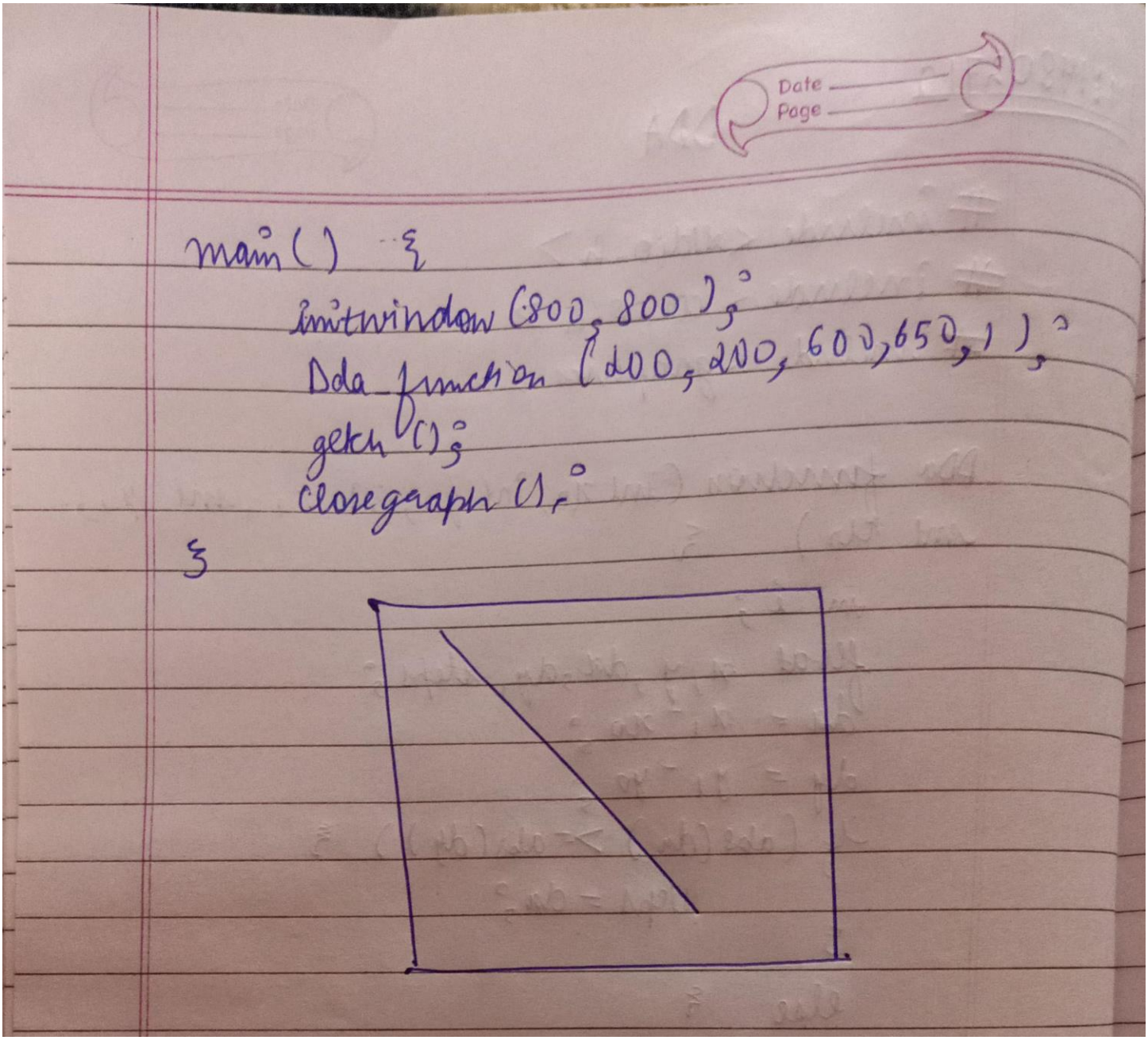
Step11: End Algorithm

1411480271g

DDA

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h>

DDa function (int x0, int y0, int x1, int y1,
int clr)  {
    int i;
    float x, y, dx, dy, steps;
    dx = x1 - x0;
    dy = y1 - y0;
    if (abs(dx) >= abs(dy)) {
        steps = dx;
    }
    else  {
        steps = dy;
    }
    dx = dx / steps;
    dy = dy / steps;
    x = x0;
    y = y0;
    i = 1;
    while (i <= steps) {
    putpixel (x, y, clr);
    x += dx;
    y += dy;
    i += 1;  }
```



Drawing line using DDA Algorithm :

CODE

```
#include<stdio.h>

#include<conio.h>

#include<graphics.h>

Dda_function(int x0,int y0,int x1,int y1,int clr)

{

        int i;
```

```
        float x,y,dx,dy ,steps;
        dx=x1-x0;
        dy=y1-y0;
        if(abs(dx)>=abs(dy))        {
                steps=dx;
        }
        Else    {
                steps=dy;
        }
        dx=dx/steps;
        dy=dy/steps;
        x=x0;
        y=y0;
        i=1;
        while(i<=steps)       {
                putpixel(x,y,clr);
                x+=dx;
                y+=dy;
                i=i+1;
        }
}
main(){
        initwindow(800,800);
        Dda_function(200, 200, 600, 650, 1);
        getch();
        closegraph();
}
```

Source Code:

```
// C program for drawing a chair
#include<stdio.h>
#include<conio.h>
#include<graphics.h>

Dda_function(int x0,int y0,int x1,int y1,int clr)
{
        int i;
        float x,y,dx,dy ,steps;

        dx=x1-x0;
```

```
	dy=y1-y0;
	if(abs(dx)>=abs(dy)){
		steps=dx;
	}
	else{
		steps=dy;
	}
	dx=dx/steps;
	dy=dy/steps;
	x=x0;
	y=y0;
	i=1;
	while(i<=steps){
		putpixel(x,y,clr);
		x+=dx;
		y+=dy;
		i=i+1;
	}
}
main(){
	initwindow(800,800);
	// rest
	Dda_function(200, 200, 400, 150, 1); //top
	Dda_function(200, 200, 200, 400, 1); //l line
	Dda_function(400, 150, 400, 350, 1); //r line
	Dda_function(200, 400, 400, 350, 1); //join
	// seat
	Dda_function(200, 400, 350, 450, 1);
	Dda_function(400, 350, 550, 400, 1);
	Dda_function(350, 450, 550, 400, 1);
	// legs
	Dda_function(200, 400, 200, 500, 1);
```

```
        Dda_function(210, 410, 210, 510, 1);
        Dda_function(200, 500, 210, 510, 1);



        Dda_function(400, 440, 400, 520, 1);
        Dda_function(410, 440, 410, 520, 1);
        Dda_function(400, 520, 410, 520, 1);

        Dda_function(340, 450, 340, 590, 1);
        Dda_function(350, 450, 350, 590, 1);
        Dda_function(340, 590, 350, 590, 1);

        Dda_function(550, 400, 550, 500, 1);
        Dda_function(540, 400, 540, 500, 1);
        Dda_function(540, 500, 550, 500, 1);

        //c
        Dda_function(500, 200, 500, 300, 3);
        Dda_function(500, 200, 610, 200, 3);
        Dda_function(500, 300, 610, 300, 3);
        Dda_function(610, 280, 610, 300, 3);
        Dda_function(610, 200, 610, 220, 3);
        Dda_function(520, 220, 610, 220, 3);
        Dda_function(520, 280, 610, 280, 3);
        Dda_function(520, 220, 520, 280, 3);

getch();
closegraph();

}
```

OUTPUT