Jahangirnagar University

3rd Year 1st Semester Examination 2022

***Course title**: Computer Graphics Laboratory*
***Course code**: CSE-304*

Lab Report -3

Submitted to-

Dr. Mohammad Shorif Uddin
Professor
Department of Computer Science and Engineering
Jahangirnagar University

Dr. Morium Akter
Associate Professor
Department of Computer Science and Engineering
Jahangirnagar University

Submitted by:

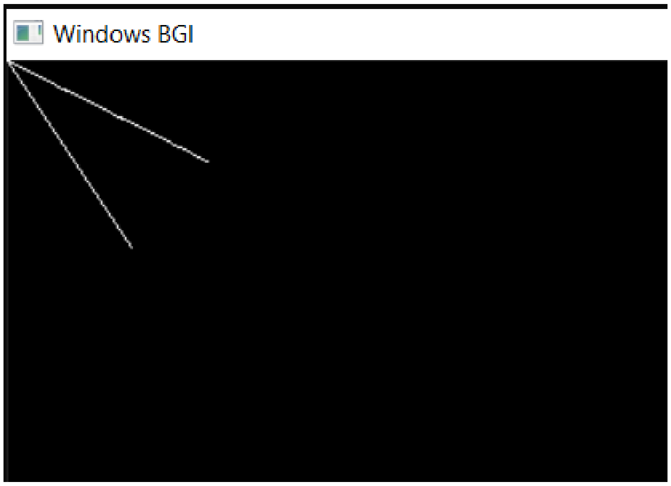
Name	**:**	**Akila Nipo**
Class Roll	**:**	**368**
Exam Roll	**:**	**202180**

❖ Experiment No-1: Scan Convert a Line Object from (0,0) to (100,50) and Rotate it by 30˚

Source Code:

```
#include <graphics.h>
#include <iostream>
#include <cmath>

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

    int x1 = 0, y1 = 0, x2 = 100, y2
= 50;

    // Original line
    line(x1, y1, x2, y2);

    // Rotate by 30 degrees
    float theta = 30 * (M_PI / 180);
// Convert degrees to radians

    int new_x1 = round(x1 *
cos(theta) - y1 * sin(theta));
    int new_y1 = round(x1 *
sin(theta) + y1 * cos(theta));
    int new_x2 = round(x2 *
cos(theta) - y2 * sin(theta));
    int new_y2 = round(x2 *
sin(theta) + y2 * cos(theta));

    // Rotated line
    line(new_x1, new_y1, new_x2,
new_y2);

    delay(5000*3600);
    closegraph();
    return 0;
}
```

Screenshot:

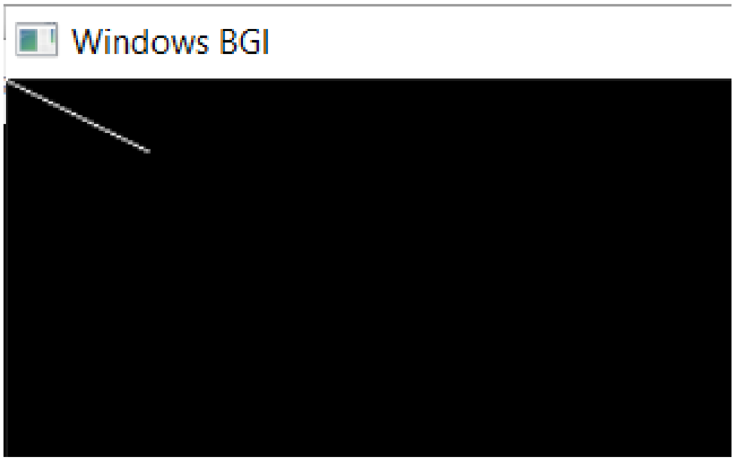


❖ Experiment No-2: Scan Conversion of A Line Object from (0,0) to (100,50) and Scaling it to 50%

Source Code:

```
#include <graphics.h>

void drawScaledLine(int x1, int y1,
int x2, int y2, float scaleFactor) {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "");

    // Scale the coordinates
    int scaledX1 = x1 * scaleFactor;
    int scaledY1 = y1 * scaleFactor;
    int scaledX2 = x2 * scaleFactor;
    int scaledY2 = y2 * scaleFactor;

    // Scan convert the scaled line
using Bresenham's algorithm
    int dx = abs(scaledX2 -
scaledX1);
    int dy = abs(scaledY2 -
scaledY1);
    int p = 2 * dy - dx;
    int twoDy = 2 * dy;
    int twoDyMinusDx = 2 * (dy -
dx);
    int x, y, xEnd;

    if (scaledX1 > scaledX2) {
        x = scaledX2;
        y = scaledY2;
        xEnd = scaledX1;
    } else {
        x = scaledX1;
        y = scaledY1;
        xEnd = scaledX2;
    }
while (x <= xEnd) {
        // Plot the scaled
coordinates
        putpixel(x, y, WHITE);

        // Move to the next pixel
        x++;

        // Update the decision
parameter
        if (p < 0)
            p += twoDy;
        else {
            y++;
            p += twoDyMinusDx;
        }
    }

    delay(5000); // Delay to show
the output
    closegraph();
}

int main() {
    int x1 = 0, y1 = 0;
    int x2 = 100, y2 = 50;
    float scaleFactor = 0.5;

    drawScaledLine(x1, y1, x2, y2,
scaleFactor);

    return 0;
}
```

Screenshot:

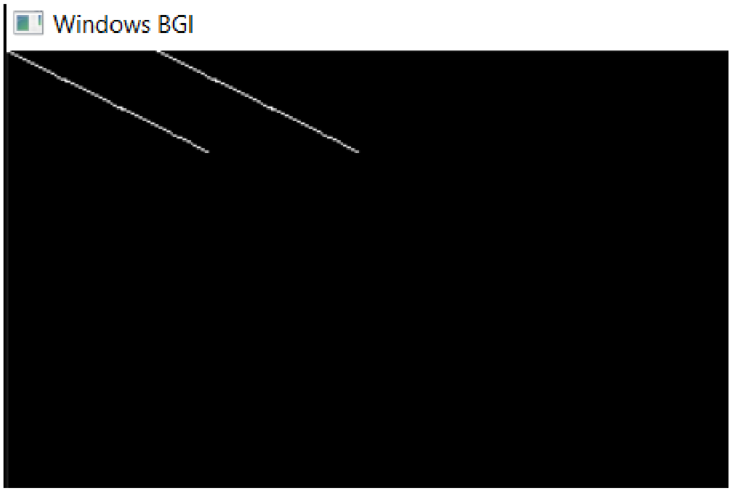


❖ Experiment No-3: Scan Conversion of A Line Object from (0,0) to (100,50) and Translate it on x-axis by 75 pixels

Source Code:

```
#include <graphics.h>
#include <iostream>

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

    int x1 = 0, y1 = 0, x2 = 100, y2
= 50;

    // Original line
    line(x1, y1, x2, y2);

    // Translate on x-axis by 75
pixels
    int translate_x = 75;
int new_x1 = x1 + translate_x;
    int new_y1 = y1;
int new_x2 = x2 + translate_x;
    int new_y2 = y2;

    // Translated line
    line(new_x1, new_y1, new_x2,
new_y2);

    delay(5000);
    closegraph();
    return 0;
}
```

Screenshot:



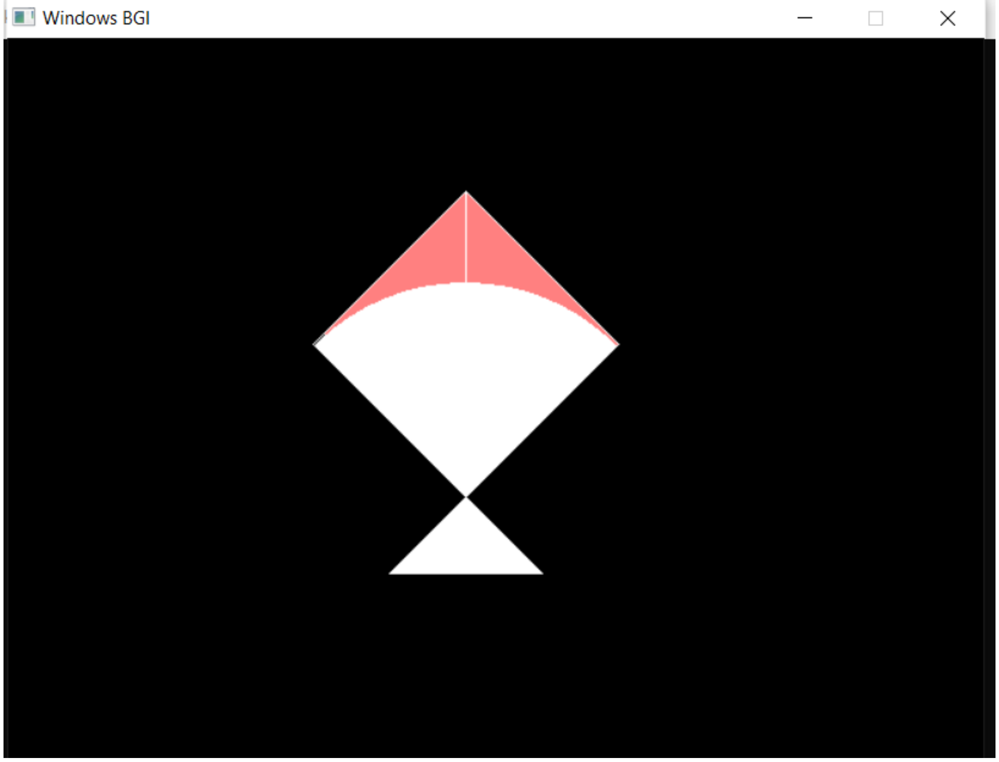
❖ Experiment No-4: Drawing a kite using Bressenham line algorithm:

Source Code:

```
#include <graphics.h>
#include <iostream>
#include <conio.h>
#include <math.h>

using namespace std;
void line_bressenham(int x1,int
y1,int x2,int y2)
{
    int dx = abs(x2 - x1);
    int dy = abs(y2 - y1);
    int x, y;

    if (x1 < x2) {
        x = x1;
        y = y1;
    } else {
        x = x2;
        y = y2;
        x2 = x1;
        y2 = y1;
    }

    int p = 2 * dy - dx;
    putpixel(x, y, WHITE);

    while (x < x2) {
        x++;
        if (p < 0) {
            p += 2 * dy;
        } else {
            y++;
            p += 2 * (dy - dx);
        }
        putpixel(x, y, BLACK);
    }
```

```
void kite()
{
    line(200, 200, 300, 100);
    line(300, 100, 400, 200);
    line(400, 200, 300, 300);
    line(300, 100, 300, 300);
    line(300,300,200,200);

    //arc(300, 300, 45, 135, 140);
    setfillstyle(SOLID_FILL, 12);
    floodfill(301, 105, WHITE);
    setfillstyle(SOLID_FILL, 12);

    floodfill(299, 105, WHITE);
    setfillstyle(SOLID_FILL, WHITE);

    floodfill(299, 275, WHITE);
    setfillstyle(SOLID_FILL, WHITE);

    floodfill(301, 275, WHITE);
    line(300, 300, 250, 350);
    line(250, 350, 350, 350);
    line(300, 300, 350, 350);
    setfillstyle(SOLID_FILL, WHITE);

    floodfill(300, 310, WHITE);
}
int main()
{
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "");

    kite();
    getch();
    closegraph();
    return 0;
}
```

Screenshot:



---------------------------THE END---------------------------