

**Department of Computer Science & Engineering(CSE)**

**Lab -07**

Name : Jabed Iqbal Joy

Student ID : C193049

Semester : 7th

Section : 7BM

Email : c193049@ugrad.iiuc.ac.bd

Contact : 01837844828

Course Code : CSE-4742

Course Title : Computer Graphics Lab

Name of the course Teacher :

**Mahadi Hassan**

Assistant Professor

Department of CSE, IIUC

Date of Submission **:** 10/04/2023

1. Rotate a point about origin.

Code:

#include<bits/stdc++.h>

#include <graphics.h>

using namespace std;

#define pi acos(-1.0)

int main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "");

line(250,100,250,300);

line(250,300,450,300);

int x=50,y=100,h=250,k=300;

double thita =60;

thita = (thita\*pi)/180;

int x\_prime = (x\*cos(thita)) - (y\*sin(thita));

int y\_prime = (x\*sin(thita)) + (y\*(cos(thita)));

circle(h+x,k-y,5);

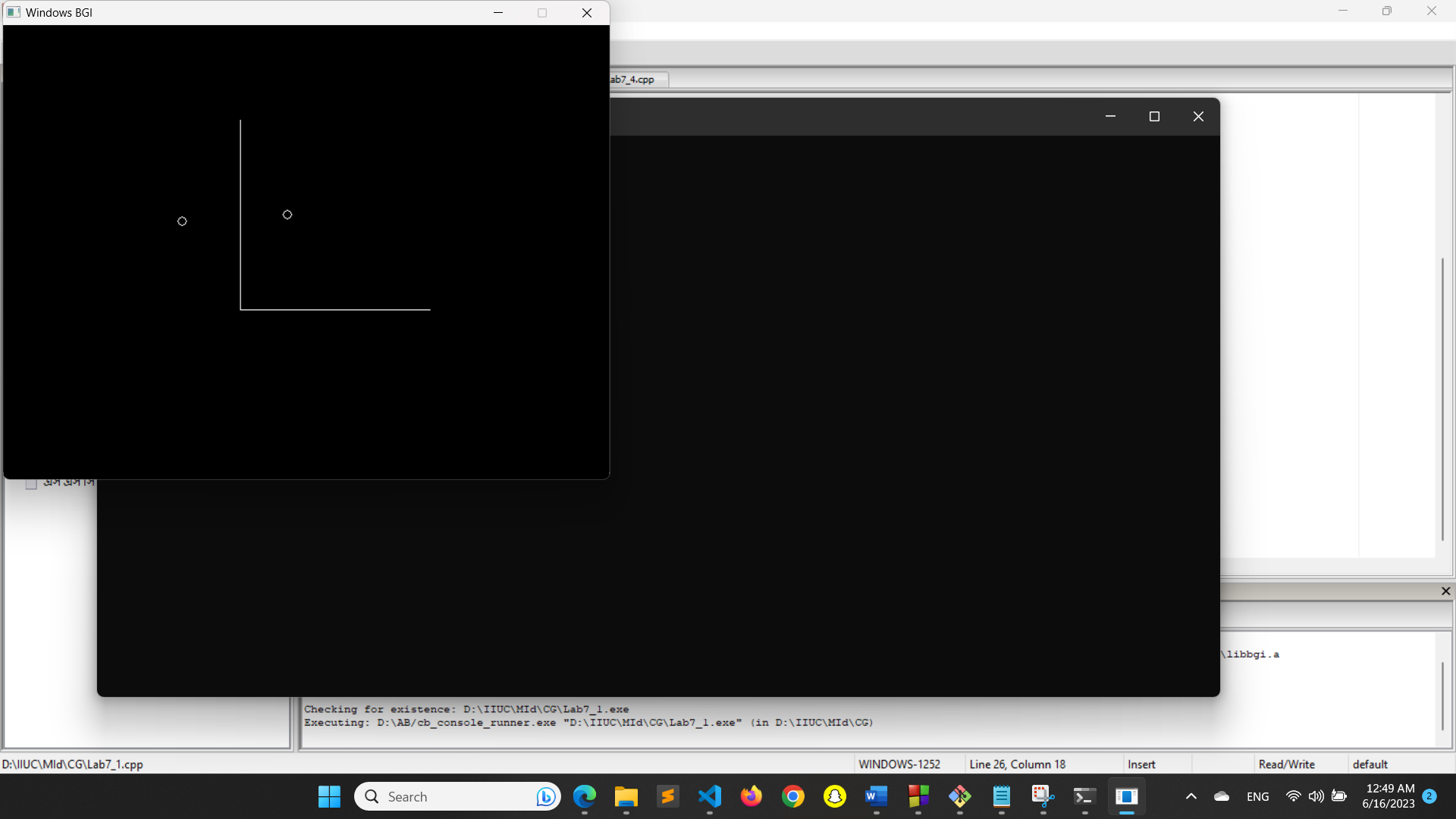
circle(h+x\_prime,k-y\_prime,5);

getch();

closegraph();

return 0;

}



1. Rotate a point about another point.

Code:

#include <graphics.h>

#include <stdlib.h>

#include <math.h>

#define PI 3.14159265

void rotate\_point(int x1, int y1, int x2, int y2, int \*new\_x, int \*new\_y, float angle)

{

// Convert angle to radians

angle = angle \* PI / 180.0;

// Translate point (x2, y2) to origin

int x = x1 - x2;

int y = y1 - y2;

// Rotate point around origin

int new\_x\_temp = x \* cos(angle) - y \* sin(angle);

int new\_y\_temp = x \* sin(angle) + y \* cos(angle);

// Translate point back to original position

\*new\_x = new\_x\_temp + x2;

\*new\_y = new\_y\_temp + y2;

}

int main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "");

// Original point

int x1 = 100, y1 = 100;

circle(x1, y1, 3);

// Point to rotate around

int x2 = 200, y2 = 200;

circle(x2, y2, 3);

// Angle of rotation in degrees

float angle = 45;

// Rotate point

int new\_x, new\_y;

rotate\_point(x1, y1, x2, y2, &new\_x, &new\_y, angle);

// Display rotated point

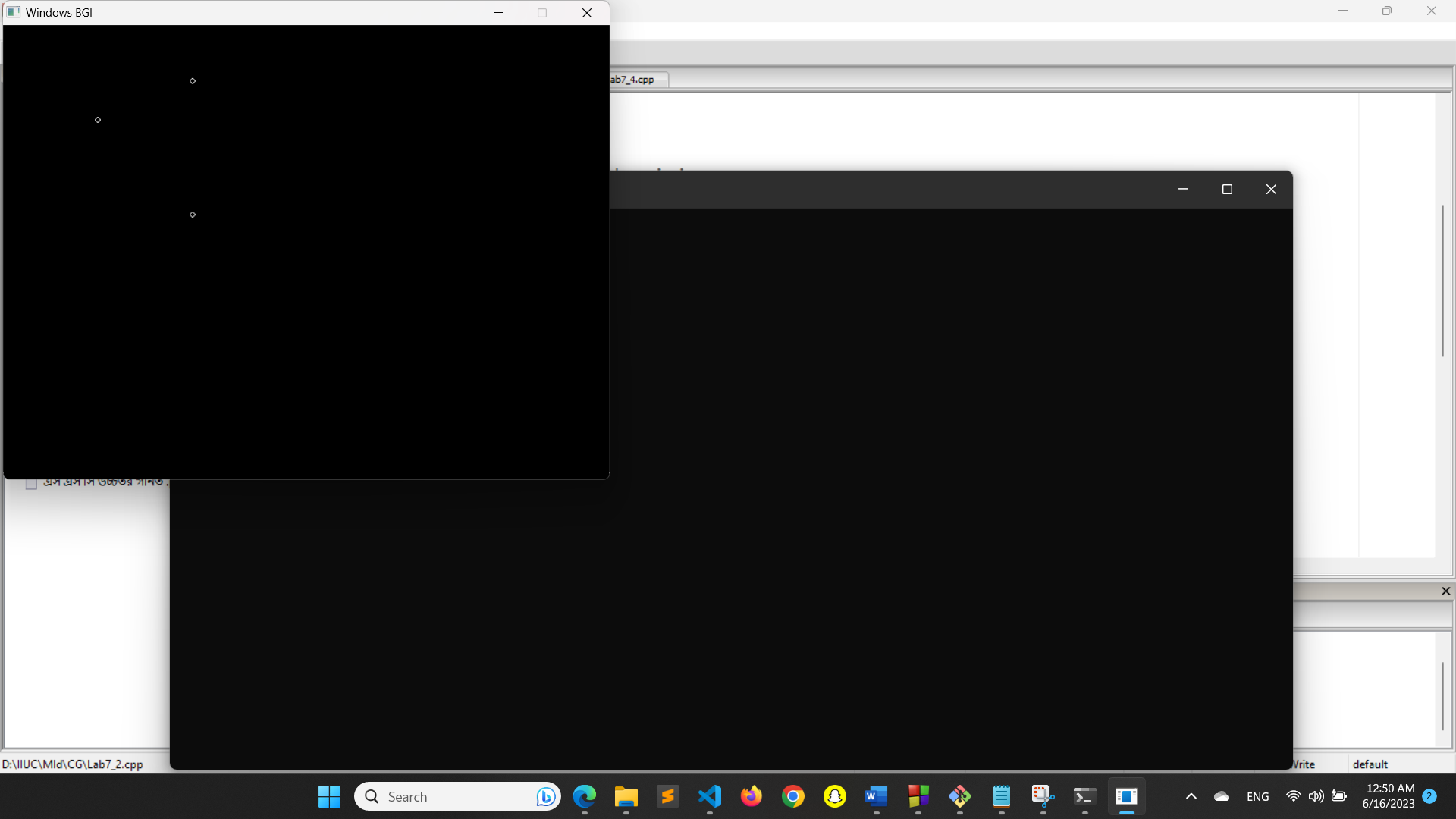
circle(new\_x, new\_y, 3);

getch();

closegraph();

return 0;

}



1. Rotate a line about a point.

Code:

#include <graphics.h>

#include <stdlib.h>

#include <math.h>

#define PI 3.14159265

void rotate\_line(int x1, int y1, int x2, int y2, int x, int y, float angle, int \*new\_x1, int \*new\_y1, int \*new\_x2, int \*new\_y2)

{

// Convert angle to radians

angle = angle \* PI / 180.0;

// Translate points (x1, y1) and (x2, y2) to origin

int a1 = x1 - x;

int b1 = y1 - y;

int a2 = x2 - x;

int b2 = y2 - y;

// Rotate points around origin

int new\_a1 = a1 \* cos(angle) - b1 \* sin(angle);

int new\_b1 = a1 \* sin(angle) + b1 \* cos(angle);

int new\_a2 = a2 \* cos(angle) - b2 \* sin(angle);

int new\_b2 = a2 \* sin(angle) + b2 \* cos(angle);

// Translate points back to original position

\*new\_x1 = new\_a1 + x;

\*new\_y1 = new\_b1 + y;

\*new\_x2 = new\_a2 + x;

\*new\_y2 = new\_b2 + y;

}

int main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "");

// Original line

int x1 = 100, y1 = 100, x2 = 200, y2 = 200;

line(x1, y1, x2, y2);

// Point to rotate around

int x = 200, y = 200;

circle(x, y, 5);

// Angle of rotation in degrees

float angle = 45;

// Rotate line

int new\_x1, new\_y1, new\_x2, new\_y2;

rotate\_line(x1, y1, x2, y2, x, y, angle, &new\_x1, &new\_y1, &new\_x2, &new\_y2);

// Display rotated line

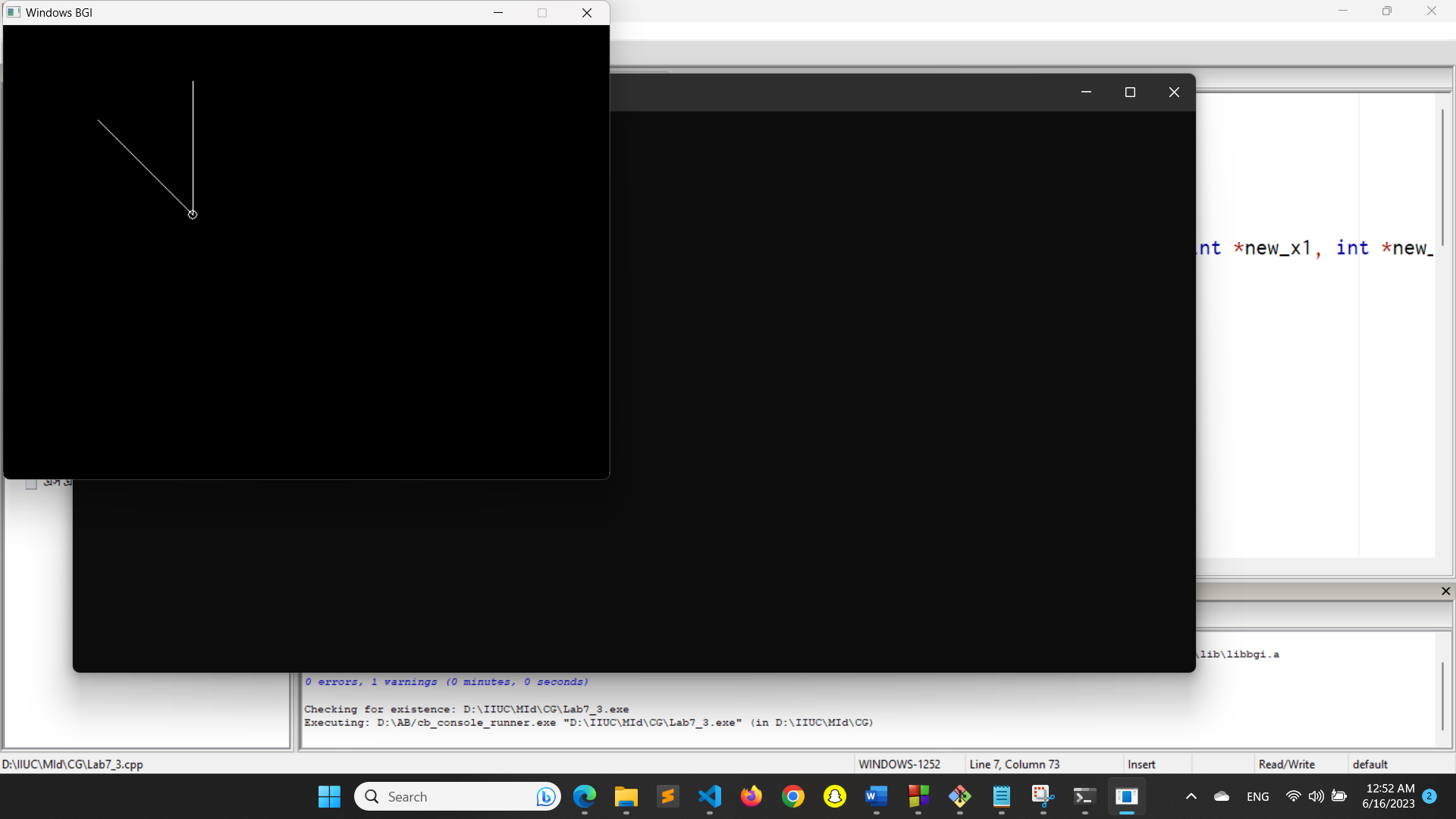
line(new\_x1, new\_y1, new\_x2, new\_y2);

getch();

closegraph();

return 0;

}



1. Rotate a triangle about a point

Code:

#include <graphics.h>

#include <stdlib.h>

#include <math.h>

#define PI 3.14159265

void rotate\_triangle(int x1, int y1, int x2, int y2, int x3, int y3, int x, int y, float angle,

int \*new\_x1, int \*new\_y1, int \*new\_x2, int \*new\_y2, int \*new\_x3, int \*new\_y3)

{

// Convert angle to radians

angle = angle \* PI / 180.0;

// Translate points (x1, y1), (x2, y2) and (x3, y3) to origin

int a1 = x1 - x;

int b1 = y1 - y;

int a2 = x2 - x;

int b2 = y2 - y;

int a3 = x3 - x;

int b3 = y3 - y;

// Rotate points around origin

int new\_a1 = a1 \* cos(angle) - b1 \* sin(angle);

int new\_b1 = a1 \* sin(angle) + b1 \* cos(angle);

int new\_a2 = a2 \* cos(angle) - b2 \* sin(angle);

int new\_b2 = a2 \* sin(angle) + b2 \* cos(angle);

int new\_a3 = a3 \* cos(angle) - b3 \* sin(angle);

int new\_b3 = a3 \* sin(angle) + b3 \* cos(angle);

// Translate points back to original position

\*new\_x1 = new\_a1 + x;

\*new\_y1 = new\_b1 + y;

\*new\_x2 = new\_a2 + x;

\*new\_y2 = new\_b2 + y;

\*new\_x3 = new\_a3 + x;

\*new\_y3 = new\_b3 + y;

}

int main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "");

// Original triangle

int x1 = 100, y1 = 100, x2 = 200, y2 = 200, x3 = 150, y3 = 50;

line(x1, y1, x2, y2);

line(x2, y2, x3, y3);

line(x3, y3, x1, y1);

// Point to rotate around

int x = 200, y = 200;

circle(x, y, 3);

// Angle of rotation in degrees

float angle = 45;

// Rotate triangle

int new\_x1, new\_y1, new\_x2, new\_y2, new\_x3, new\_y3;

rotate\_triangle(x1, y1, x2, y2, x3, y3, x, y, angle, &new\_x1, &new\_y1, &new\_x2, &new\_y2, &new\_x3, &new\_y3);

// Display rotated triangle

line(new\_x1, new\_y1, new\_x2, new\_y2);

line(new\_x2, new\_y2, new\_x3, new\_y3);

line(new\_x3, new\_y3, new\_x1, new\_y1);

getch();

closegraph();

return 0;

}

