আন্তর্জাতিক ইসলামী বিশ্ববিদ্যালয় চট্টগ্রাম

الجامعة الإسلامية العالمية شيتاغونغ

International Islamic University Chittagong

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

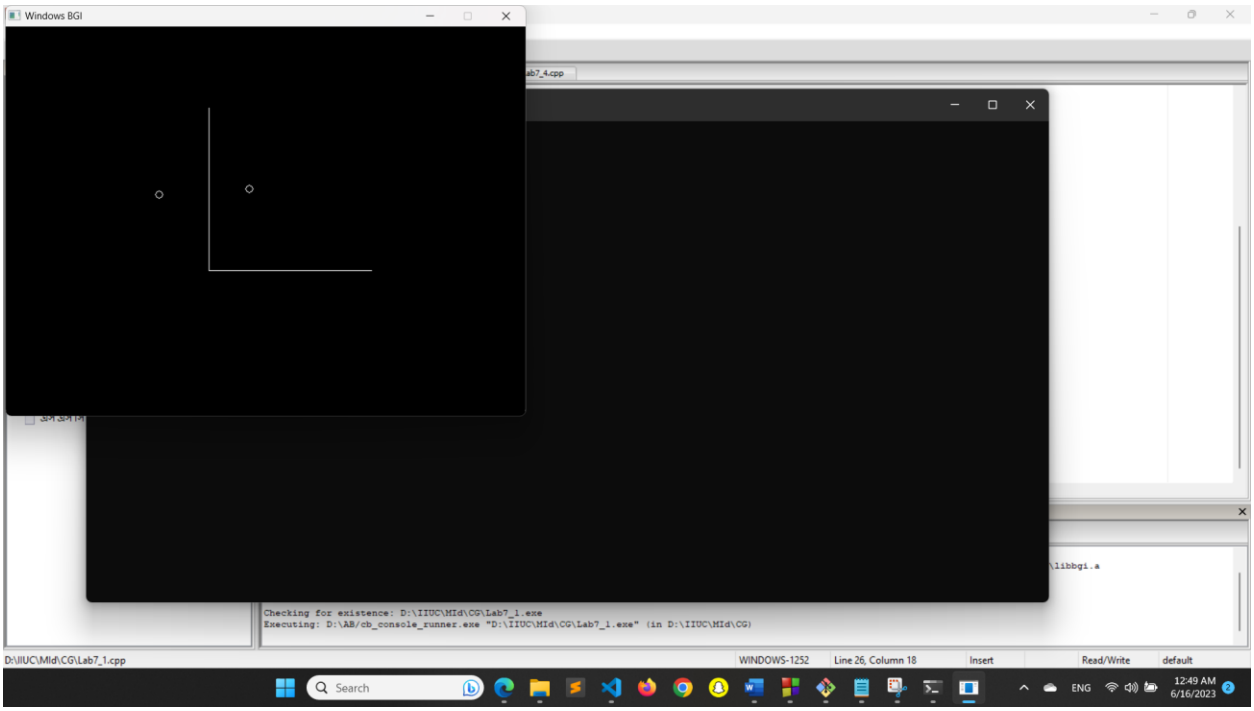
Department of CSE, IIUC

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;
}
```

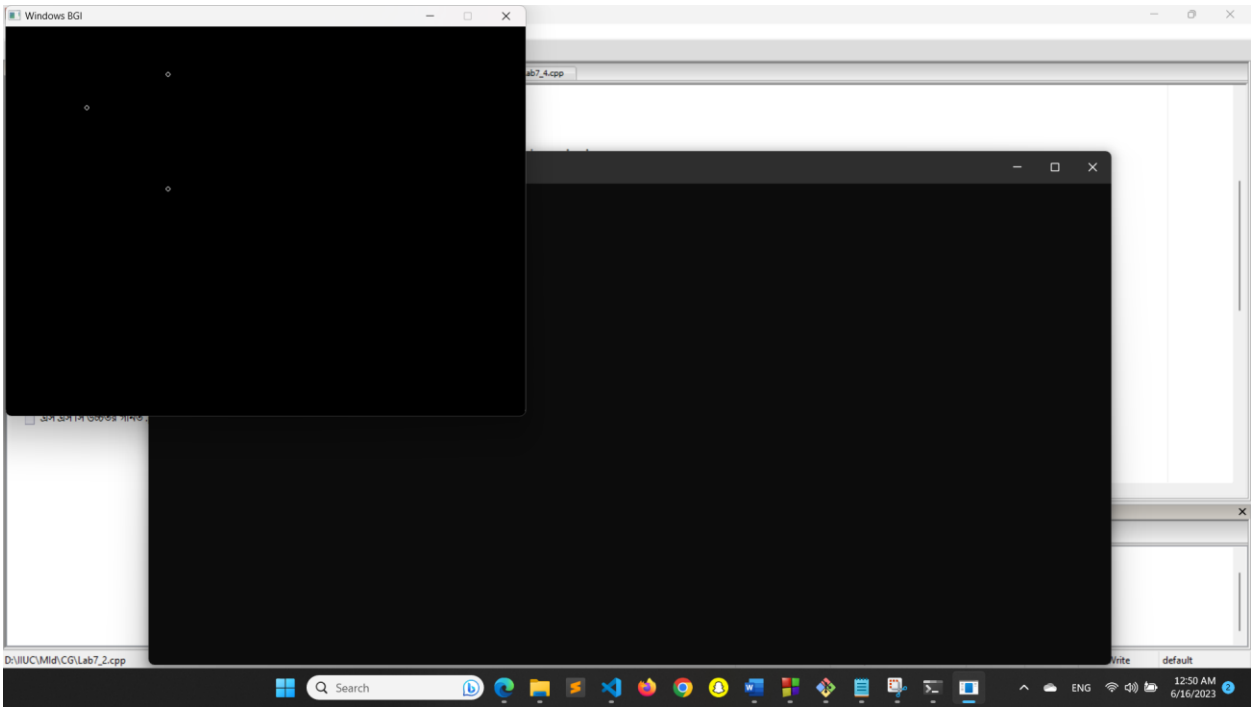


2. Rotate a point about another point.

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;
}
```



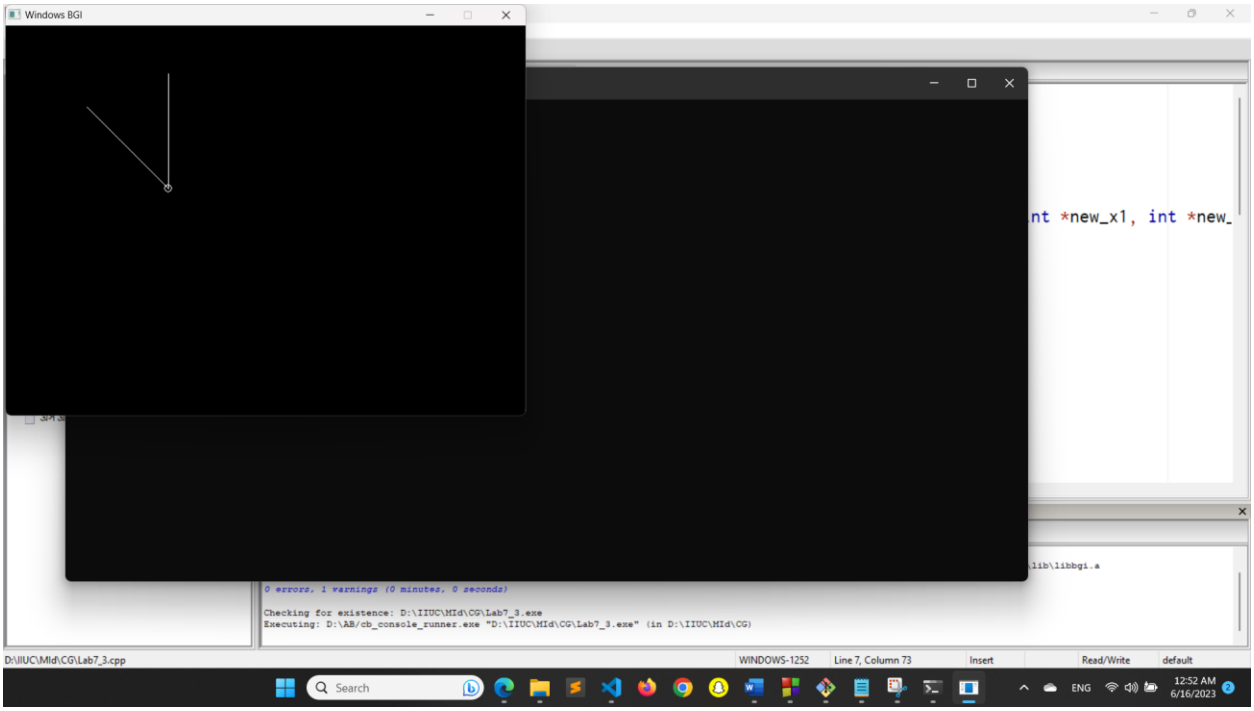
3. Rotate a line about a point.

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;
}
```



4. 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;
}
```