

**Lab Report:
Title:**

Course title: Computer Graphics Laboratory
Course code: CSE-304
3rd Year 1st Semester Examination 2022

Date of Submission: 11.06.2023



Submitted to-

Dr. Mohammad Shorif Uddin
Dr. Morium Akter

Department of Computer Science and Engineering
Jahangirnagar University
Savar, Dhaka-1342

Sl	Class Roll	Exam Roll	Name
01	351		Umma Sumaiya Jahan

1. Roate:

Code:

```
#include <math.h>
#include <conio.h>
#include <graphics.h>
#include <bits/stdc++.h>
#define PI acos(-1)
using namespace std;
void drawline(double x0, double y0, double x1, double y1)
{
    double dx, dy, p, x, y;
    dx=x1-x0;
    dy=y1-y0;
    x=x0;
    y=y0;
    p=2*dy-dx;
    while(x<x1)
    {
        if(p>=0)
        {
            putpixel(x,y,RED);
            y=y+1;
            p=p+2*dy-2*dx;
        }
        else
```

```

        {
            putpixel(x,y,RED);
            p=p+2*dy;
        }
        x=x+1;
    }
}

int main()
{
    int gd=0,gm,x1,y1,x2,y2,x3,x4,y3,y4;
    double s,c, angle;
    initgraph(&gd,&gm,"C:\\Tc\\BGI");
    x1=0,y1=0,x2=100,y2=150;
    printf("(x1,y1)=(%d, %d) and (x2,y2)=(%d,%d)\n",x1,y1,x2,y2);
    drawline(x1,y1,x2,y2);
    setcolor(CYAN);
    angle=30;
    c = cos(angle * PI /180);
    s = sin(angle * PI /180);
    x3 = floor(x1 * c + y1 * s);
    y3 = floor(-x1 * s + y1 * c);
    x4 = floor(x2 * c + y2 * s);
    y4 = floor(-x2 * s + y2 * c);

    printf("After 30degree rotation keeping (x1,y1) unchanged
(x2,y2)=(%d, %d)",x4,y4);

```

```

        drawline(x1,y1,x4,y4);

    getch();

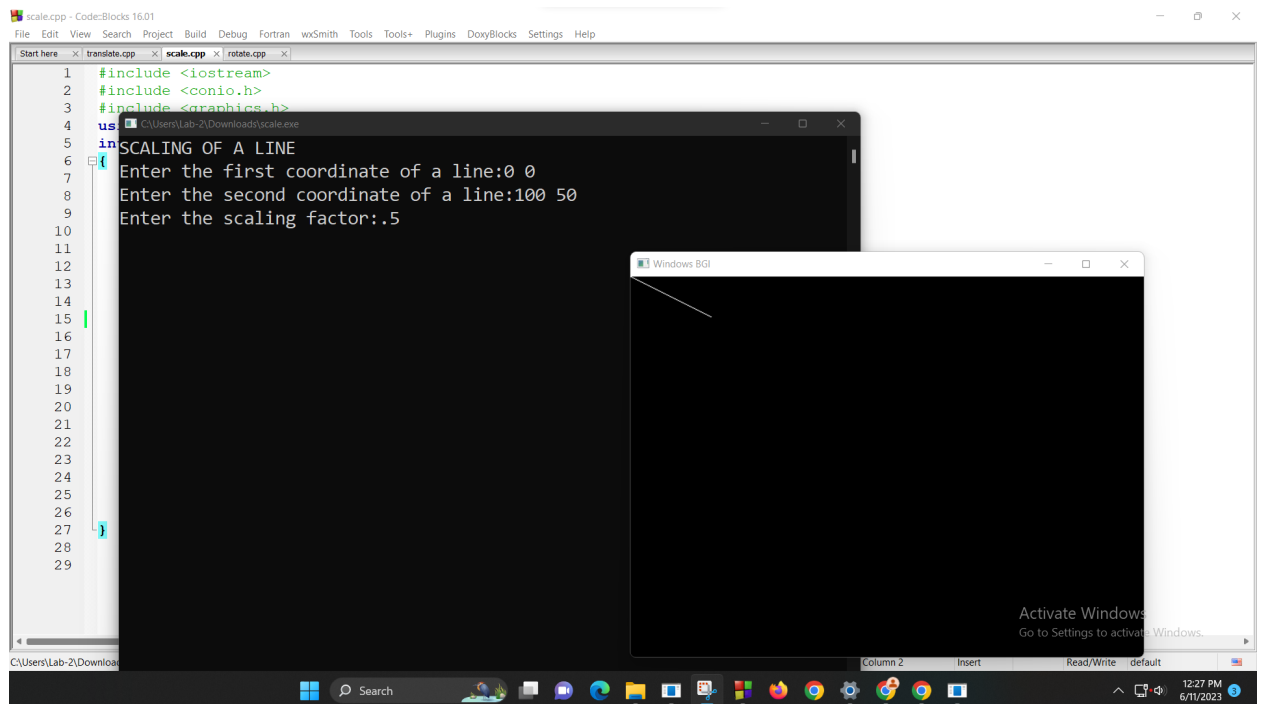
    closegraph();

    return 0;

}

```

Output:

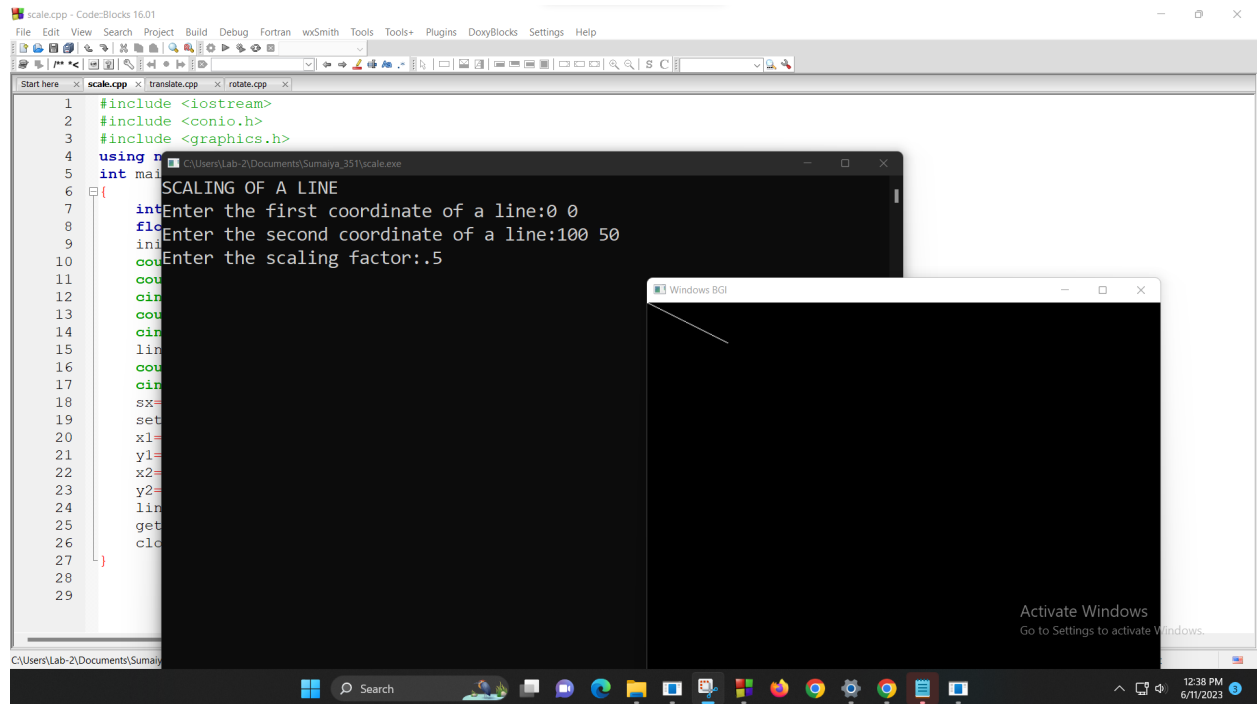


Scale:

```
#include <iostream>
#include <conio.h>
#include <graphics.h>
using namespace std;
int main()
{
    int gd=DETECT,gm;
    float x1,y1,x2,y2,sx,sy,s;
    initgraph(&gd,&gm,"C:\\\\Tc\\\\BGI");
    cout<<"SCALING OF A LINE\n";
    cout<<"Enter the first coordinate of a line:";
    cin>>x1>>y1;
    cout<<"Enter the second coordinate of a line:";
    cin>>x2>>y2;
    line(x1,y1,x2,y2);
    cout<<"Enter the scaling factor:";
    cin>>s;
    sx=s/100,sy=s/100;
    setcolor(RED);
    x1=x1*sx;
    y1=y1*sy;
    x2=x2*sx;
    y2=y2*sy;
    line(x1,y1,x2,y2);
```

```
    getch();  
    closegraph();  
}
```

Output:



The screenshot displays a Windows desktop environment. In the background, the Code::Blocks IDE is open, showing a C++ source file named 'scale.cpp'. The code includes headers for `<iostream>`, `<conio.h>`, and `<graphics.h>`, and uses the `using namespace std;` directive. The main function contains a loop for scaling a line, with variables for coordinates and a scaling factor. In the foreground, a command prompt window titled 'C:\Users\Lab-2\Documents\Sumaiya_351\scale.exe' shows the program's output. It displays the title 'SCALING OF A LINE' and prompts the user to enter the first coordinate (0), the second coordinate (100), and the scaling factor (5). Another window titled 'Windows BGI' is also visible, showing a black screen with a white line segment drawn from the origin.

```
scale.cpp - Code::Blocks 16.01  
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help  
Start here x scale.cpp x translate.cpp x rotate.cpp x  
1 #include <iostream>  
2 #include <conio.h>  
3 #include <graphics.h>  
4 using namespace std;  
5 int main()  
6 {  
7     SCALING OF A LINE  
8     int x1, y1, x2, y2;  
9     float sx, sy;  
10    cout << "Enter the first coordinate of a line:" << endl;  
11    cin >> x1;  
12    cout << "Enter the second coordinate of a line:" << endl;  
13    cin >> y1;  
14    cout << "Enter the scaling factor:" << endl;  
15    cin >> sx;  
16    cout << "Enter the scaling factor:" << endl;  
17    cin >> sy;  
18    setviewport(0, 0, 600, 600);  
19    x1 = x1 * sx;  
20    y1 = y1 * sy;  
21    x2 = x1 + 100;  
22    y2 = y1 + 100;  
23    line(x1, y1, x2, y2);  
24    getch();  
25    closegraph();  
26    return 0;  
27 }  
28  
29
```

```
SCALING OF A LINE  
Enter the first coordinate of a line:0  
Enter the second coordinate of a line:100  
Enter the scaling factor:5
```

Windows BGI

Activate Windows
Go to Settings to activate Windows.

12:38 PM
6/11/2023

Translate:

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

using namespace std;

int main()
{
    int gd=DETECT,gm,x1,x2,y1,y2,tx,ty;
    initgraph(&gd,&gm,"C:\\Tc\\BGI");
    cout<<"Enter the first co-ordinate of a line:";
    cin>>x1>>y1;
    cout<<"Enter the second co-ordinate of a line:";
    cin>>x2>>y2;
    line(x1,y1,x2,y2);
    cout<<"Enter the translation vector:";
    cin>>tx;
    setcolor(RED);
    x1=x1+tx;
    x2=x2+tx;
    line(x1,y1,x2,y2);
    getch();
    closegraph();
}
```

Output :

The screenshot displays a Windows desktop environment. In the background, a Code::Blocks IDE window is open, showing the source code for a program named 'translate.cpp'. The code is as follows:

```
1 Enter the first co-ordinate of a line:0 0
2 Enter the second co-ordinate of a line:100 50
3
4 Enter the translation vector:75
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
```

In the foreground, a console window titled 'translate.exe' shows the program's execution. It displays the same prompts and user input as the code editor. To the right of the console, a graphics window titled 'Windows BGI' is open, showing a black canvas with a white line segment. The line starts at the origin (0,0) and extends to the point (75, 50), representing the translation vector. The status bar at the bottom of the IDE indicates 'Line 19, Column 14' and 'Insert' mode. The Windows taskbar at the bottom shows the system clock as 12:39 PM on 6/11/2023.