**EXPE**

**PROGRAM NO. 5(a)**

**AIM-** To perform translation transformation on a given triangle.

#include<iostream.h>

#include<conio.h>

#include<graphics.h>

void main()

{

inttx,ty;

intgd=DETECT,gm;

initgraph(&gd,&gm,"C:\\turboc3\\bgi");

line(50,150,200,150);

line(50,150,125,50);

line(125,50,200,150);

cout<<"Enter translation factors"<<endl;

cout<<"Enter tx: ";

cin>>tx;

cout<<"Enter ty: ";

cin>>ty;

line(50+tx,150+ty,200+tx,150+ty);

line(50+tx,150+ty,125+tx,50+ty);

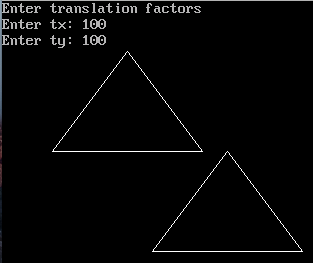
line(125+tx,50+ty,200+tx,150+ty);

getch();

closegraph();

}

**OUTPUT:-**



**PROGRAM NO. 5(b)**

**AIM-** To perform scaling transformation on a given triangle.

#include<iostream.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int x;

intgd=DETECT,gm;

initgraph(&gd,&gm,"C:\\turboc3\\bgi");

line(100,200,250,200);

line(100,200,175,100);

line(175,100,250,200);

cout<<"Enter Scaling factors"<<endl;

cin>>x;

line(100-2\*(5\*x),200+(5\*x),250+2\*(5\*x),200+(5\*x));

line(100-2\*(5\*x),200+(5\*x),175,100-2\*(5\*x));

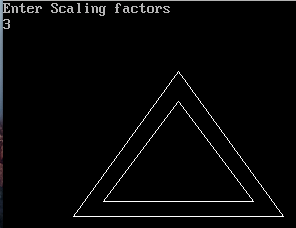
line(175,100-2\*(5\*x),250+2\*(5\*x),200+(5\*x));

getch();

closegraph();

}

**OUTPUT:-**



**PROGRAM NO. 5(c)**

**AIM-** To perform rotation transformation on a given triangle.

#include<iostream.h>

#include<conio.h>

#include<graphics.h>

#include<math.h>

void main()

{

intgd = DETECT, gmode;

initgraph(&gd, &gmode, "C:\\turboc3\\bgi");

double x1, x2, x3, y1, y2, y3, rot;

cout<<"Enter First X - Coordinate : ";

cin>>x1;

cout<<"Enter First Y - Coordinate : ";

cin>>y1;

cout<<"Enter Second X - Coordinate : ";

cin>>x2;

cout<<"Enter Second Y - Coordinate : ";

cin>>y2;

cout<<"Enter Third X - Coordinate : ";

cin>>x3;

cout<<"Enter Third Y - Coordinate : ";

cin>>y3;

line(x1, y1, x2, y2);

line(x2, y2, x3, y3);

line(x3, y3, x1, y1);

cout<<"Enter Rotation Angle in Degrees : ";

cin>>rot;

doublefpx = (x1+x2+x3)/3;

doublefpy = (y1+y2+y3)/3;

double rad = rot\*3.14/180;

double X1 = (x1-fpx)\*cos(rad) - (y1-fpy)\*sin(rad) + fpx;

double Y1 = (x1-fpx)\*sin(rad) + (y1-fpx)\*cos(rad) + fpy;

double X2 = (x2-fpx)\*cos(rad) - (y2-fpy)\*sin(rad) + fpx;

double Y2 = (x2-fpx)\*sin(rad) + (y2-fpx)\*cos(rad) + fpy;

double X3 = (x3-fpx)\*cos(rad) - (y3-fpy)\*sin(rad) + fpx;

double Y3 = (x3-fpx)\*sin(rad) + (y3-fpx)\*cos(rad) + fpy;

setcolor(BLUE);

line(X1, Y1, X2, Y2);

line(X2, Y2, X3, Y3);

line(X3, Y3, X1, Y1);

getch();

closegraph();

}

**OUTPUT:-**

