```
//Program: To draw 2-d objects and perform basic transformations (1.Scaling
2.Translation 3.Rotation)
#include<iostream>
#include<graphics.h>
#include<cstdlib>
#include<math.h>
using namespace std;

int n;
class transformer
{
 private:
  float ax[10][3];
  float sx,sy,tx,ty,x;
  float s[10][3],t[10][3],r[10][3]; //s=scale result ; t=translate result ;
r=rotate result
  int j;

 public:
  void dda(float x1, float y1, float x2, float y2) //dda start
  {
   int steps, dx=(x2-x1), dy=(y2-y1);

   if(abs(dx)>abs(dy))
     steps=abs(dx);
   else
     steps=abs(dy);

   float xinc=(float)dx/steps, yinc=(float)dy/steps;
   float x=x1, y=y1;

   putpixel(x,y,15);
   int a,b;

   for(int i=1 ; i<=steps ; i++)
   {
    x = (x + xinc);
    y = (y + yinc);

    a=x + 0.5;
    b=y + 0.5;

    putpixel(a,b,15);
   }
  } //dda end

  void get(int n)
  {
   cout<<"\nENTER THE CO-ORDINATES OF 2D OBJECT : \n";
   for(int i=0;i<n;i++) //Entering 2d object's matrix
   {
    cout<<"ENTER THE "<<i+1<<" COORDINATE : ";
    cin>>ax[i][0]>>ax[i][1];
    ax[i][2]=1;
   }
  }

  void show(int n)
  {
   for(int i=0;i<n;i++) //Ploting 2d object
   {
    j=i+1;
    if(i==n-1)
     j=0;
    dda(ax[i][0],ax[i][1],ax[j][0],ax[j][1]);
   }
```

```
}

void scale(int n)
{
 cout<<"\nENTER SCALING IN X-DIRECTION : ";
 cin>>sx;
 cout<<"\nENTER SCALING IN Y-DIRECTION : ";
 cin>>sy;

 for(int i=0;i<n;i++) //scaling
 {
  s[i][0]=ax[i][0]*sx;
  s[i][1]=ax[i][1]*sy;
 }
 for(int i=0;i<n;i++) //Ploting scaled object
 {
  j=i+1;
  if(i==n-1)
   j=0;
  setlinestyle(3, 0, 1);
  line(s[i][0],s[i][1],s[j][0],s[j][1]);
 }
}

void translate(int n)
{
 cout<<"\nENTER TRANSLATION IN X-DIRECTION : ";
 cin>>tx;
 cout<<"\nENTER TRANSLATION IN Y-DIRECTION : ";
 cin>>ty;

 for(int i=0;i<n;i++) //translation
 {
  t[i][0]=ax[i][0]+tx;
  t[i][1]=ax[i][1]+ty;
 }
 for(int i=0;i<n;i++) //Ploting translated object
 {
  j=i+1;
  if(i==n-1)
   j=0;
  setlinestyle(3, 0, 1);
  line(t[i][0],t[i][1],t[j][0],t[j][1]);
 }
}

void rotate(int n, int rot)
{
 cout<<"ENTER ROTATION ANGLE(IN DEGREE)(wrt origin) : ";
 cin>>x;

 x=x*0.01745; //to convert (degree -> radian) multiply x by pi/180

 if(rot == 'a') //clockwise rotation
 {
  for(int i=0;i<n;i++)
  {
   r[i][0]=ax[i][0]*cos(x)-ax[i][1]*sin(x);
   r[i][1]=ax[i][0]*sin(x)+ax[i][1]*cos(x);
  }
 }
 else if(rot == 'b') //anticlockwise rotation
 {
  for(int i=0;i<n;i++)
  {
   r[i][0]=ax[i][0]*cos(x)+ax[i][1]*sin(x);
```

```
     r[i][1]=-ax[i][0]*sin(x)+ax[i][1]*cos(x);
    }
   }

   for(int i=0;i<n;i++) //plotting rotated object
   {
    j=i+1;
    if(i==n-1)
     j=0;
    setlinestyle(3, 0, 1);
    line(r[i][0],r[i][1],r[j][0],r[j][1]);
   }
  }

  void operator <<(transformer w) //operator overloading
  {
   for(int i=0; i<n; i++)
   {
    cout<<" ";
    for(int j=0; j<3; j++)
      cout<<ax[i][j]<<"\t";
    cout<<"\n";
   }
  }

}t,a;

int main()
{
 int ch;
 char r;

 int gd=DETECT, gm;
 initgraph(&gd, &gm, NULL);

 cout<<"\nENTER THE NO OF VERTICES OF 2D OBJECT : \n";
 cin>>n;

 t.get(n);

 cout<<"2D OBJECT IN MATRIX FORM IS: \n";
 t<<a;  //operator overloaded

 t.show(n);

 cout<<"\n***************** BASIC 2-D TRANSFORMATION *************************";
 cout<<"\n1. SCALING\n2. ROTATION\n3. TRANSLATION \nEnter your choice : ";
 cin>>ch;

 switch(ch)
 {
  case 1: {
           t.scale(n);
           break;
          }
  case 2: {
           cout<<"a. CLOCKWISE ROTATION\nb. ANTICLOCKWISE ROTATION\nENTER YOUR
CHOICE : ";
           cin>>r;
           t.rotate(n,r);
           break;
          }
  case 3: {
           t.translate(n);
           break;
          }
  default: {
            cout<<"\nYOU HAVE ENTERED WRONG CHOICE!!!\n";
            break;
           }
 }

 getch();
 closegraph();

 return(0);
}
/////////////////////////////////////
OUTPUT     ///////////////////////////////////
gaurav@gaurav-Inspiron-3542:~$ g++ cgprac8.cpp -lgraph
gaurav@gaurav-Inspiron-3542:~$ ./a.out
ENTER THE NO OF VERTICES OF 2D OBJECT : 3

ENTER THE CO-ORDINATES OF 2D OBJECT :
ENTER THE 1 COORDINATE : 200  200
ENTER THE 2 COORDINATE : 25   200
ENTER THE 3 COORDINATE : 100  50
2D OBJECT IN MATRIX FORM IS:
 200    200     1
 25     200     1
 100    50      1

***************** BASIC 2-D TRANSFORMATION ************************
1. SCALING
2. ROTATION
3. TRANSLATION
Enter your choice : 1

ENTER SCALING IN X-DIRECTION : 1.5
ENTER SCALING IN Y-DIRECTION : 1.5

gaurav@gaurav-Inspiron-3542:~$ g++ cgprac8.cpp -lgraph
gaurav@gaurav-Inspiron-3542:~$ ./a.out
ENTER THE NO OF VERTICES OF 2D OBJECT : 4

ENTER THE CO-ORDINATES OF 2D OBJECT :
ENTER THE 1 COORDINATE : 150  150
ENTER THE 2 COORDINATE : 150  250
ENTER THE 3 COORDINATE : 300  250
ENTER THE 4 COORDINATE : 300  150
2D OBJECT IN MATRIX FORM IS:
 150    150     1
 150    250     1
 300    250     1
 300    150     1

***************** BASIC 2-D TRANSFORMATION ************************
1. SCALING
2. ROTATION
3. TRANSLATION
Enter your choice : 2

a. CLOCKWISE ROTATION
b. ANTICLOCKWISE ROTATION
ENTER YOUR CHOICE : b
ENTER ROTATION ANGLE(IN DEGREE)(wrt origin) : 20

gaurav@gaurav-Inspiron-3542:~$ g++ cgprac8.cpp -lgraph
gaurav@gaurav-Inspiron-3542:~$ ./a.out
ENTER THE NO OF VERTICES OF 2D OBJECT : 5

ENTER THE CO-ORDINATES OF 2D OBJECT :
```

```
ENTER THE 1 COORDINATE : 150  150
ENTER THE 2 COORDINATE : 150  250
ENTER THE 3 COORDINATE : 300  250
ENTER THE 4 COORDINATE : 300  150
ENTER THE 5 COORDINATE : 225  100
2D OBJECT IN MATRIX FORM IS:
 150    150     1
 150    250     1
 300    250     1
 300    150     1
 225    100     1

***************** BASIC 2-D TRANSFORMATION ************************
1. SCALING
2. ROTATION
3. TRANSLATION
Enter your choice : 3

ENTER TRANSLATION IN X-DIRECTION : 50
ENTER TRANSLATION IN Y-DIRECTION : 50
gaurav@gaurav-Inspiron-3542:~$
```