

```

#include<iostream>
#include<graphics.h>
using namespace std;
class transform
{
public:
int a[10][10],m,i,j;
double c[10][10];
void object();
void accept();
void display();
void mul(double b[10][10]);
};

void transform::accept()
{
cout<<"Enter no. of edges:";
cin>>m;
for(i=0;i<m;i++)
{
    for(j=0;j<3;j++)
    {
        if(j>=2)
            a[i][j]=1;
        else
            cin>>a[i][j];
    }
}
}

void transform::object()
{
    for(i=0;i<(m-1);i++)
    {
        line(a[i][0],a[i][1],a[i+1][0],a[i+1][1]);
    }
    line(a[0][0],a[0][1],a[i][0],a[i][1]);
}

void transform::display()
{
    for(i=0;i<(m-1);i++)
    {
        line(c[i][0],c[i][1],c[i+1][0],c[i+1][1]);
    }
}

```

```

        }
        line(c[0][0],c[0][1],c[i][0],c[i][1]);
    }

void transform::mul(double b[10][10])
{
    int i,j,k;
    for(i=0;i<m;i++)
    {
        for(j=0;j<m;j++)
        {
            c[i][j]=0;
            for(k=0;k<m;k++)
            {
                c[i][j]+=a[i][k]*b[k][j];
            }
        }
    }
}

int main()
{
    int gd=DETECT,gm,ch,sx,sy,tx,ty,deg;
    transform s1;
    double b[10][10];
    s1.accept();
    cout<<"\n1.Translation";
    cout<<"\n2.Scaling";
    cout<<"\n3.Rotation";
    cin>>ch;
    switch(ch)
    {
        case 1: cout<<"TRANSLATION OPERATION";
                  cout<<"\nEnter tx and ty";
                  cin>>tx>>ty;
                  b[0][0]=b[1][1]=b[2][2]=1;
                  b[2][0]=tx;
                  b[2][1]=ty;
                  b[0][1]=b[0][2]=b[1][0]=b[1][2]=0;
                  break;
        case 2: cout<<"SCALING OPERATION";
                  cout<<"\nEnter Sx and Sy";
                  cin>>sx>>sy;
                  b[0][0]=sx;

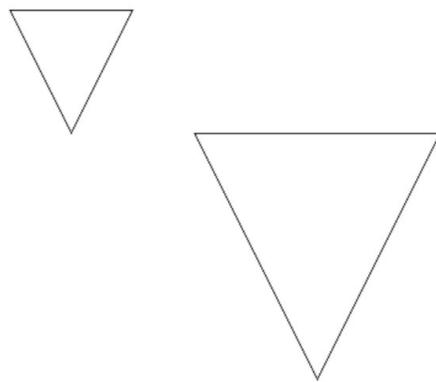
```

```

        b[1][1]=sy;
        b[2][2]=1;
        b[0][1]=b[0][2]=b[1][0]=b[1][2]=b[2][0]=b[2][1]=0;
        break;
    case 3:cout<<"ROTATION OPERATION";
        cout<<"\nEnter DEGREE";
        cin>>deg;
        b[0][0]=b[1][1]=cos(deg*3.142/180);
        b[0][1]=sin(deg*3.142/180);
        b[1][0]=-sin(deg*3.142/180);
        b[2][2]=1;
        b[0][2]=b[1][2]=b[2][0]=b[2][1]=0;
        break;
    }
initgraph(&gd,&gm,NULL);
s1.object();
s1.mul(b);
s1.display();
getch();
closegraph();
}

```

  SDL-libgraph – Graphics on GNU/Linux




---

