Experiment No 6

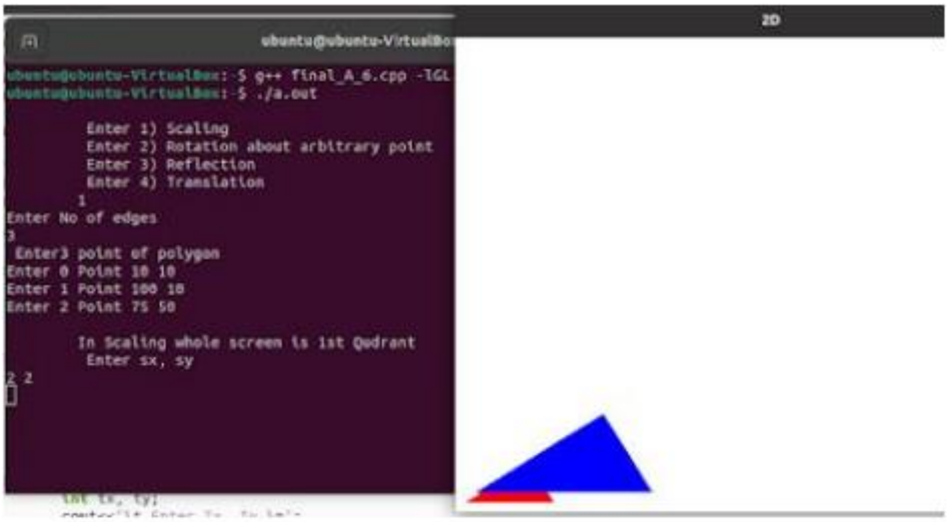
Title Implement following 2D transformations on the object with respect to axis : i) Scaling ii) Rotation about arbitrary point iii) Reflection

```
#include <iostream>

#include <math.h>

#include <time.h>

#include <GL/glut.h>

#include <vector>

using namespace std;

int edge;

vector<int> xpoint;

vector<int> ypoint;

int ch;

double round(double d){

 return floor(d + 0.5);

}

void init(){

 glClearColor(1.0,1.0,1.0,0.0);

 glMatrixMode(GL_PROJECTION);

 gluOrtho2D(0,640,0,480);

 glClear(GL_COLOR_BUFFER_BIT);

}

void translation(){

 int tx, ty;

 cout<<"\t Enter Tx, Ty \n";

 cin>> tx>> ty;


 //Translate the point

 for(int i=0;i<edge;i++){


 xpoint[i] = xpoint[i] + tx;
 ypoint[i] = ypoint[i] + ty;

 }


 glBegin(GL_POLYGON);
 glColor3f(0,0,1);
 for(int i=0;i<edge;i++){
 glVertex2i(xpoint[i],ypoint[i]);
 }
 glEnd();
 glFlush();
}
void rotaion(){
 int cx, cy;
 cout<<"\n Enter Ar point x , y ";
 cin >> cx >> cy;

 cx = cx+320;
 cy = cy+240;
 glColor3f(0.0, 1.0, 0.0);
 glBegin(GL_POINTS);
 glVertex2i(cx,cy);
 glEnd();
 glFlush();

 double the;
 cout<<"\n Enter thetha ";
 cin>>the;
 the = the * 3.14/180;
 glColor3f(0,0,1.0);
 glBegin(GL_POLYGON);
 for(int i=0;i<edge;i++){
 glVertex2i(round(((xpoint[i] - cx)*cos(the) - ((ypoint[i]-cy)*sin(the))) + cx),
 round(((xpoint[i] - cx)*sin(the) + ((ypoint[i]-cy)*cos(the))) + cy));
 }
 glEnd();
 glFlush();
}
void scale(){
 glColor3f(1.0,0,0);
 glBegin(GL_POLYGON);
 for(int i=0;i<edge;i++){
 glVertex2i(xpoint[i]-320,ypoint[i]-240);
 }
 glEnd();
 glFlush();
 cout<<"\n\tIn Scaling whole screen is 1st Qudrant \n";
 int sx, sy;
 cout<<"\t Enter sx, sy \n";
 cin>> sx>> sy;

 //scale the point
 for(int i=0;i<edge;i++){

 xpoint[i] = (xpoint[i]-320) * sx;
 ypoint[i] = (ypoint[i]-240) * sy;
 }

 glColor3f(0,0,1.0);
 glBegin(GL_POLYGON);
 for(int i=0;i<edge;i++){
 glVertex2i(xpoint[i],ypoint[i]);
 }
 glEnd();
 glFlush();
}
void reflection(){
 char reflection;
 cout<<"Enter Reflection Axis \n";
 cin>> reflection;

 if(reflection == 'x' || reflection == 'X'){

 glColor3f(0.0,0.0,1.0);
 glBegin(GL_POLYGON);
 for(int i=0;i<edge;i++){
 glVertex2i(xpoint[i], (ypoint[i] * -1)+480);
 }
 glEnd();
 glFlush();


 }
 else if(reflection == 'y' || reflection == 'Y'){
 glColor3f(0.0,0.0,1.0);
 glBegin(GL_POLYGON);
 for(int i=0;i<edge;i++){
 glVertex2i((xpoint[i] * -1)+640,(ypoint[i]));
 }
 glEnd();
 glFlush();
 }
```

```
}
void Draw(){
 if(ch==2 || ch==3 || ch==4){
 glColor3f(1.0,0,0);
 glBegin(GL_LINES);
 glVertex2i(0,240);
 glVertex2i(640,240);
 glEnd();
 glColor3f(1.0,0,0);
 glBegin(GL_LINES);
 glVertex2i(320,0);
 glVertex2i(320,480);
 glEnd();
 glFlush();

 glColor3f(1.0,0,0);
 glBegin(GL_POLYGON);
 for(int i=0;i<edge;i++){
 glVertex2i(xpoint[i],ypoint[i]);
 }
 glEnd();
 glFlush();
 }
 if(ch==1){
 scale();
 }
 else if(ch == 2){
 rotaion();
 }
 else if( ch == 3){
 reflection();
 }
 else if (ch == 4){
 translation();
 }
}
int main(int argc, char** argv){

 cout<<"\n \t Enter 1) Scaling ";
 cout<<"\n \t Enter 2) Rotation about arbitrary point";
 cout<<"\n \t Enter 3) Reflection";
 cout<<"\n \t Enter 4) Translation \n \t";

 cin>>ch;

 if(ch==1 || ch==2 || ch==3 || ch==4){

 cout<<"Enter No of edges \n";
 cin>> edge;
 int xpointnew, ypointnew;
 cout<<" Enter"<< edge <<" point of polygon \n";
 for(int i=0;i<edge;i++){

 cout<<"Enter "<< i << " Point ";
 cin>>xpointnew>>ypointnew;

 xpoint.push_back(xpointnew+320);
 ypoint.push_back(ypointnew+240);

 }
 glutInit(&argc, argv);
 glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
 glutInitWindowSize(640,480);
 glutInitWindowPosition(200,200);
 glutCreateWindow("2D");
 init();
 glutDisplayFunc(Draw);


 glutMainLoop();
 return 0;
 }
 else{
 cout<<"\n \t Check Input run again";
 return 0;
 }
}
```

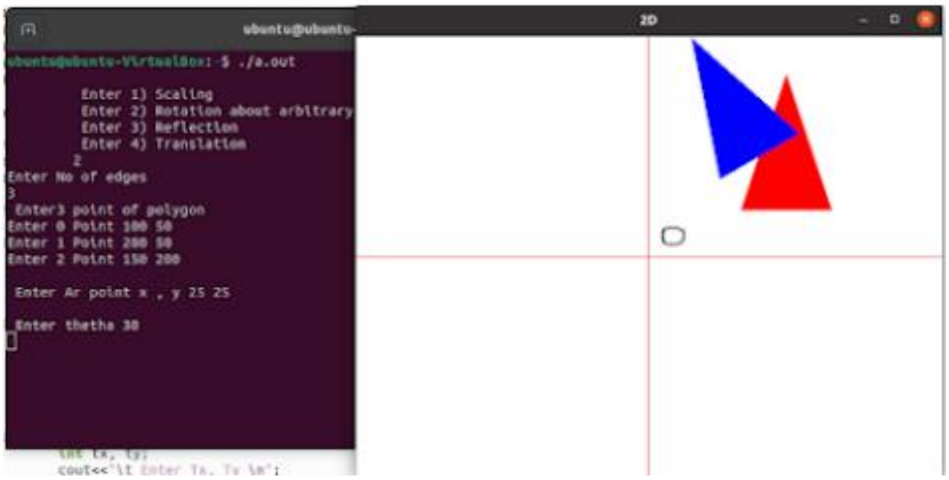
OUTPUT

```
g++ filename.cpp -lGL -lGLU -lglut
./a.out
```



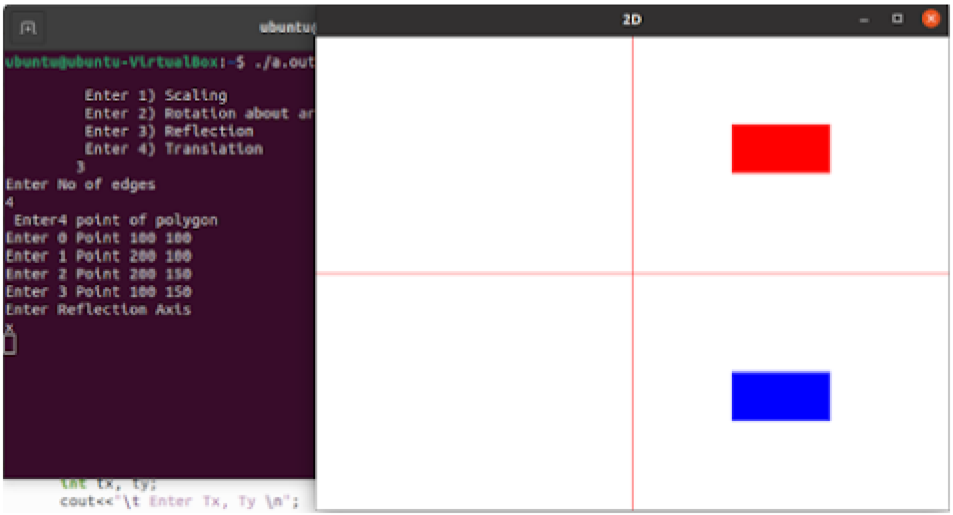
```
ubuntu@ubuntu-VirtualBox:~$ g++ final_A_6.cpp -lGL
ubuntu@ubuntu-VirtualBox:~$ ./a.out

         Enter 1) Scaling
         Enter 2) Rotation about arbitrary point
         Enter 3) Reflection
         Enter 4) Translation
        1
Enter No of edges
3
 Enter3 point of polygon
Enter 0 Point 10 10
Enter 1 Point 100 10
Enter 2 Point 75 50

        In Scaling whole screen is 1st Qudrant
         Enter sx, sy
2 2
```

```
      int tx, ty;
```



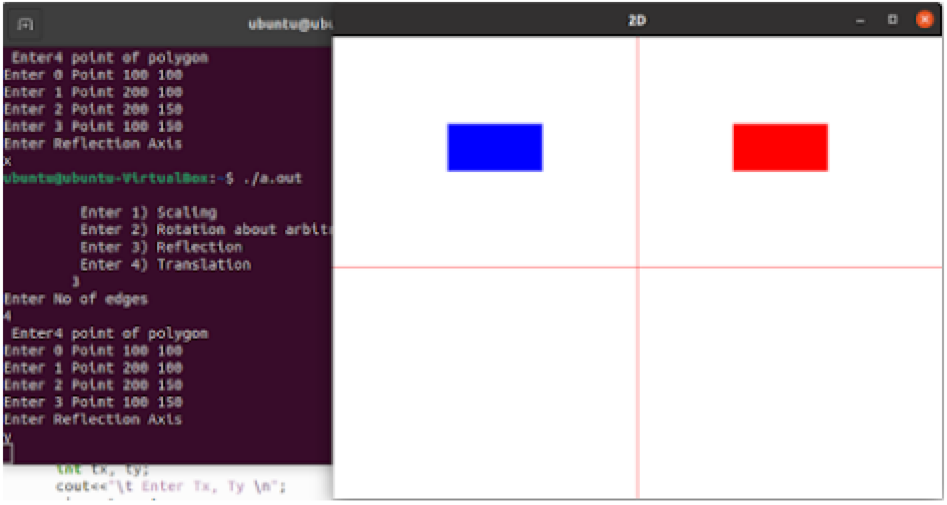
```
ubuntu@ubuntu-VirtualBox:~$ ./a.out

         Enter 1) Scaling
         Enter 2) Rotation about arbitrary
         Enter 3) Reflection
         Enter 4) Translation
        2
Enter No of edges
3
 Enter3 point of polygon
Enter 0 Point 100 50
Enter 1 Point 200 50
Enter 2 Point 150 200

 Enter Ar point x , y 25 25

 Enter thetha 30
```

```
      int tx, ty;
      cout<<"\t Enter Tx, Ty \n";
```



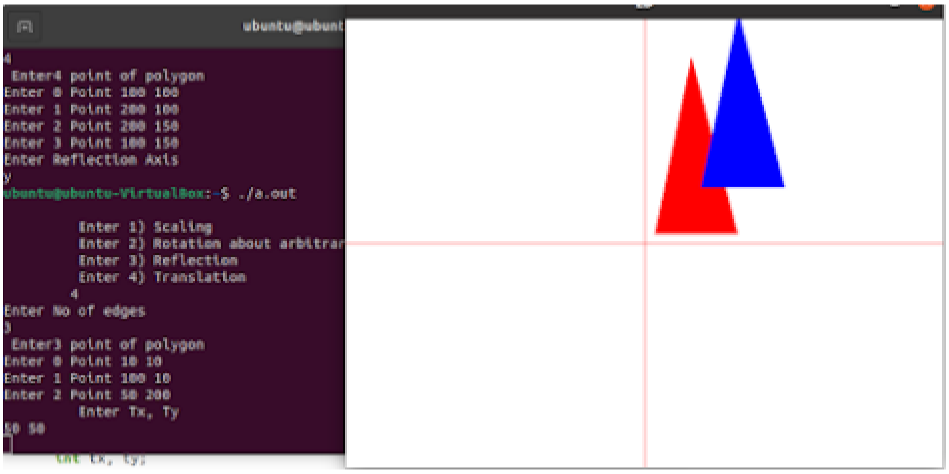
```
ubuntu@ubuntu-VirtualBox:~$ ./a.out

         Enter 1) Scaling
         Enter 2) Rotation about ar
         Enter 3) Reflection
         Enter 4) Translation
        3
Enter No of edges
4
 Enter4 point of polygon
Enter 0 Point 100 100
Enter 1 Point 200 100
Enter 2 Point 200 150
Enter 3 Point 100 150
Enter Reflection Axis
x
```

```
       int tx, ty;
       cout<<"\t Enter Tx, Ty \n";
```



```
 Enter4 point of polygon
Enter 0 Point 100 100
Enter 1 Point 200 100
Enter 2 Point 200 150
Enter 3 Point 100 150
Enter Reflection Axis
x
ubuntu@ubuntu-VirtualBox:~$ ./a.out

         Enter 1) Scaling
         Enter 2) Rotation about arbitr
         Enter 3) Reflection
         Enter 4) Translation
        3
Enter No of edges
4
 Enter4 point of polygon
Enter 0 Point 100 100
Enter 1 Point 200 100
Enter 2 Point 200 150
Enter 3 Point 100 150
Enter Reflection Axis
y
```

```
       int tx, ty;
       cout<<"\t Enter Tx, Ty \n";
```



```
4
 Enter4 point of polygon
Enter 0 Point 100 100
Enter 1 Point 200 100
Enter 2 Point 200 150
Enter 3 Point 100 150
Enter Reflection Axis
y
ubuntu@ubuntu-VirtualBox:~$ ./a.out

         Enter 1) Scaling
         Enter 2) Rotation about arbitrar
         Enter 3) Reflection
         Enter 4) Translation
        4
Enter No of edges
3
 Enter3 point of polygon
Enter 0 Point 10 10
Enter 1 Point 100 10
Enter 2 Point 50 200
         Enter Tx, Ty
50 50
```

```
       int tx, ty;
```