```
#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 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 Draw(){
        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();
    }



int main(int argc, char** argv){

        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);
scale();
glutMainLoop();
   return 0;
}
```