```c
#include<stdlib.h>
#include<stdio.h>
#include<GL/glut.h>

typedef float point[3];
point v[]= {{0, 0, 1}, {0, 1, 0}, {-1, -0.5, 0}, {1, -0.5, 0}};
int n;

void triangle(point a,point b,point c)
{
    glBegin(GL_POLYGON)
    ; glVertex3fv(a);
    glVertex3fv(b);
    glVertex3fv(c);
    glEnd();
}

void divide_triangle(point a,point b,point c,int n)
{
    point v1,v2,v3;
    int j;
    if(n>0)
    {
        for(j=0; j<3; j++)
            v1[j] = (a[j]+b[j])/2;

        for(j=0; j<3; j++)
            v2[j] = (a[j]+c[j])/2;

        for(j=0; j<3; j++)
            v3[j] = (c[j]+b[j])/2;

        divide_triangle(a,v1,v2,n-1)
        ; glFlush();
        divide_triangle(c,v2,v3,n-1)
        ; glFlush();
        divide_triangle(b,v3,v1,n-1)
        ; glFlush();
    }
    else(triangle(a,b,c));
}

void tetrahedron(int n)
{
    glColor3f(1, 0, 0);
    divide_triangle(v[0], v[1], v[2],
    n);


    glColor3f(0, 1, 0);
    divide_triangle(v[3], v[2], v[1],
    n);


    glColor3f(0, 0, 1);
    divide_triangle(v[0], v[3], v[1],
    n);


    glColor3f(0, 0, 0);
    divide_triangle(v[0], v[2], v[3],
    n);

}
void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    tetrahedron(n);
    glFlush();
}
void myReshape(int w,int h)
{
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
```

```
    if(w<=h)
        glOrtho(-2, 2, -2*(GLfloat)h/(GLfloat)w, 2*(GLfloat)h/(GLfloat)w, -10, 10);
    else
        glOrtho(-2*(GLfloat)w/(GLfloat)h, 2*(GLfloat)w/(GLfloat)h, -2, 2, -10, 10);

    glMatrixMode(GL_MODELVIEW)
    ; glutPostRedisplay();
}
int main(int argc,char ** argv)
{
    printf("No of Recursive steps/Division: ");
    scanf("%d",&n);
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB|GLUT_DEPTH);
    glutCreateWindow(" 3D Sierpinski gasket");

    glutReshapeFunc(myReshape);

    glutDisplayFunc(display);

    glEnable(GL_DEPTH_TEST);

    glClearColor(1, 1, 1, 0);
    glutMainLoop();

    return 0;
}
```