```c
#include<gl/glut.h>
#include<stdio.h>

int m;
typedef float point[3];
point tetra[4] = { {0,100,-100},{0,0,100},{100,-100,-100},
{-100,-100,-100} };
void tetrahedron(void);
void myinit(void);
void divide_triangle(point a, point b, point c, int m);
void draw_triangle(point p1, point p2, point p3);
int main(int argv, char** argc)
{
        //int m;
        printf("Enter the number of iterations: ");
        scanf_s("%d", &m);
        glutInit(&argv, argc);
        glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
        glutInitWindowPosition(100, 200);
        glutInitWindowSize(500, 500);
        glutCreateWindow("Seirpinski Gasket");
        glutDisplayFunc(tetrahedron);
        glEnable(GL_DEPTH_TEST);
        myinit();
        glutMainLoop();
}
void divide_triangle(point a, point b, point c, int m)
{
        point v1, v2, v3;
        int j;
        if (m > 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] = (b[j] + c[j]) / 2;

                divide_triangle(a, v1, v2, m - 1);
                divide_triangle(c, v2, v3, m - 1);
                divide_triangle(b, v3, v1, m - 1);
        }
        else
                draw_triangle(a, b, c);
}
void myinit()
{
        glClearColor(1, 1, 1, 1);

        //glFlush();
        glOrtho(-500.0, 500.0, -500.0, 500.0, -500.0, 500.0);
        //gluOrtho(-500.0,500.0,-500.0,500.0,-500.0,500.0);
}
void tetrahedron(void)
{
        //myinit();
```

```c
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glColor3f(1.0, 0.0, 0.0);
        divide_triangle(tetra[0], tetra[1], tetra[2], m);
        glColor3f(0.0, 1.0, 0.0);
        divide_triangle(tetra[3], tetra[2], tetra[1], m);
        glColor3f(0.0, 0.0, 1.0);
        divide_triangle(tetra[0], tetra[3], tetra[1], m);
        glColor3f(0.0, 0.0, 0.0);
        divide_triangle(tetra[0], tetra[2], tetra[3], m);
        glFlush();
}
void draw_triangle(point p1, point p2, point p3)
{
        glBegin(GL_TRIANGLES);
        glVertex3fv(p1);
        glVertex3fv(p2);
        glVertex3fv(p3);
        glEnd();
}
```