7. Design, develop and implement recursively subdivide a tetrahedron to form 3D sierpinski gasket. The number of recursive steps is to be specified by the user.

```
#include <stdio.h>
#include<GL/glut.h>
typedef float point[3];
point v[] = { { 0,0,1 },{ 0,1,-1 },{ -1,-1,-1 },{ 1,-1,-1 } };
int n;

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

void divTri(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] = (c[j] + b[j]) / 2;

		divTri(a, v1, v2, m - 1);
		divTri(c, v2, v3, m - 1);
		divTri(b, v3, v1, m - 1);
	}
	else(triangle(a, b, c));
}

void tetra(int m)
{
	glColor3f(1, 0, 0);
	divTri(v[0], v[1], v[2], m);

	glColor3f(0, 1, 0);
	divTri(v[3], v[2], v[1], m);

	glColor3f(0, 0, 1);
	divTri(v[0], v[3], v[1], m);

	glColor3f(0, 0, 0);
	divTri(v[0], v[2], v[3], m);
}

void display(void)
{
```

```
	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
	glLoadIdentity();
	tetra(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();

}

void main(int argc, char **argv)
{
	printf("Enter number of division:");
	scanf_s("%d", &n);
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
	glutInitWindowSize(640, 840);
	glutCreateWindow("3D gasket");
	glutReshapeFunc(myReshape);
	glutDisplayFunc(display);
	glEnable(GL_DEPTH_TEST);
	glClearColor(1, 1, 1, 1);
	glutMainLoop();
}
```

Output