Aim:
Program to recursively subdivide a tetrahedron to from 3D Sierpinski gasket. The number of recursive steps is to be specified by the user.

##Theory
> A geometric method of creating the gasket is to start with a vertex of the object and get all the mid points to the other verticies. This results in smaller strucure of the original geometric object (repete for all the verticies). For another itteration take the smaller object ang perform the same as above. The gasket is perfectly self similar, an attribute of many fractal images. Any portion is an exact replica of the phase of the gasket
> The construction of the 3 dimensional version of the gasket follows similar rules for the 2D case except that the building blocks are square based pyramids instead of triangles.

Algorithm:
1. Start with a tetrahedron.
2. Inside this take the edges, calculate there mid points.
3. Now, the from each original vertex costrict a new tetrahedron consiating of vertex(original,midpoint,midpoint,midpoint)
4. repete it recursively (till m>0)
6. at m = 0 draw all the leaf nodes of the recursive tree

Code: sierpanski.c
```
    #include<stdio.h>
    #include<GL/glut.h>
    typedef float point[3];
    point v[] = {
                 {0.0,0.0,1.0},{0.0,0.942809,-0.333333},
                 {-0.816497,-0.471405,-0.333333},
                 {0.816497,-0.471405,-0.333333}
                 };
    int n;
    void draw_triangle(point a,point b,point c)
    {
      glBegin(GL_POLYGON);
      glVertex3fv(a);
      glVertex3fv(b);
      glVertex3fv(c);
      glEnd();
    }

    void midpoint(point save,point a,point b) {
      save[0]=(a[0]+b[0])/2;
      save[1]=(a[1]+b[1])/2;
      save[2]=(a[2]+b[2])/2;
    }

    void divide_tetrahedron(point a,point b,point c,point d,int m)
    {
      point ab,ac,ad,bc,bd,cd;
      //repeat 'm' no of times as specified by user
      if(m>0) {
        midpoint(ab,a,b);
        midpoint(ac,a,c);
        midpoint(ad,a,d);
        midpoint(bc,b,c);
        midpoint(bd,b,d);
        midpoint(cd,c,d);
        // consider midpoints as vertex and divide bigger triangle
        // to 3 parts recursively
        divide_tetrahedron(a,ab,ac,ad,m-1);
        divide_tetrahedron(ab,b,bc,bd,m-1);
        divide_tetrahedron(ac,bc,c,cd,m-1);
        divide_tetrahedron(ad,bd,cd,d,m-1);
        //note if u want the colors of phases to align just adjudt
the above points
      }
      //draw the sub divided traingles
      else {
        glColor3f(1.0,0.0,0.0);
        draw_triangle(a,b,c);
        glColor3f(0.0,0.0,1.0);
        draw_triangle(a,c,d);
        glColor3f(0.0,1.0,0.0);
        draw_triangle(c,b,d);
        glColor3f(0.0,0.0,0.0);
        draw_triangle(a,b,d);
      }
    }

    //this is called everytime the display is refreshed. Here it
draw a tetrahedron.
    void display(void)
    {
      glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
      glLoadIdentity();
      divide_tetrahedron(v[0],v[1],v[2],v[3],n);
      glFlush();
    }

    // This function is executed when the wiindow size is changed.
    void myReshape(int w,int h)
    {
      glViewport(0,0,w,h);
      glMatrixMode(GL_PROJECTION);
      glLoadIdentity();
      if(w<=h)
        glOrtho(-2.0,2.0,-2.0*(GLfloat)h/(GLfloat)w,2.0*(GLfloat)h/
(GLfloat)w,-10.0,10.0); //to get exact aspect ratio
      else
        glOrtho(-2.0*(GLfloat)w/(GLfloat)h,2.0*(GLfloat)w/
(GLfloat)h,-2.0,2.0,-10.0,10.0);
      glMatrixMode(GL_MODELVIEW);
      glutPostRedisplay();
    }

    int main(int argc,char **argv)
    {
      printf("no. of divisions \n");
      scanf("%d",&n);
      glutInit(&argc,argv);
      glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB|GLUT_DEPTH);
      glutInitWindowSize(500,500);
      glutCreateWindow("3DGasket");     //window with a title
      glutReshapeFunc(myReshape);
      glutDisplayFunc(display);
      glEnable(GL_DEPTH_TEST);
      glClearColor(1.0,1.0,1.0,1.0);
      glutMainLoop();
    }


## Output:
*Commands for execution:-*

* Open a terminal and Change directory to the file location in both
the terminals.
* compile as gcc -lGLU -lGL -lglut sierpanski.c -o sierpanski
* If no errors, run as ./sierpanski
```