PROGRAM -7

7. Design, develop and implement recursively subdivide a tetrahedron to form 3D sierpinski gasket.

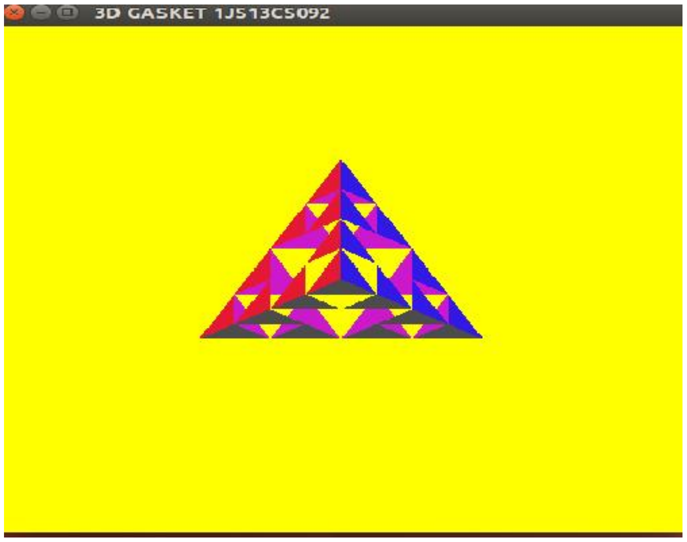
```
#include<stdio.h>
#include<GL/glut.h>
float v[][3]={{-1,-0.5,0},{1,-0.5,0},{0,1,0},{0,0,1}};
int m;
void triangle(float *p, float *q, float *r)
{
glVertex3fv(p);
glVertex3fv(q);
glVertex3fv(r);
}
void tetra(float *a, float *b, float *c, float *d)
{
glColor3f(1,0,0);
triangle(a,b,c);
glColor3f(1,1,0);
triangle(a,b,d);
glColor3f(1,0,1);
triangle(a,d,c);
glColor3f(0,1,1);
triangle(b,c,d);
}
void dt(float *a, float *b, float *c, float *d, int m)
{
float mid[6][3];
int j;
if(m>0)
{
for(j=0;j<3;j++)
{
mid[0][j]=(a[j]+b[j])/2;
mid[1][j]=(a[j]+c[j])/2;
mid[2][j]=(a[j]+d[j])/2;
mid[3][j]=(b[j]+c[j])/2;
mid[4][j]=(b[j]+d[j])/2;
mid[5][j]=(c[j]+d[j])/2;
}
dt(a,mid[0],mid[1],mid[2],m-1);
dt(mid[0],b,mid[3],mid[4],m-1);
dt(mid[1],mid[3],c,mid[5],m-1);
dt(mid[2],mid[4],mid[5],d,m-1);
}
else
tetra(a,b,c,d);
}
void display()
{
glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
glBegin(GL_TRIANGLES);
dt(v[0],v[1],v[2],v[3],m);
glEnd();
glFlush();
}
void init()
{
glClearColor(0,0,0,1);
glOrtho(-2,2,-2,2,-14,10);
}
int main(int argc, char **argv)
{
printf("enter the no. of steps\n");
scanf("%d",&m);
glutInit(&argc,argv);
glutInitDisplayMode(GLUT_SINGLE|GLUT_DEPTH);
glutInitWindowSize(700,700);
glutCreateWindow("3d Gasket");
init();
glutDisplayFunc(display);
glEnable(GL_DEPTH_TEST);
glutMainLoop();
}
```

<u>Output command</u>

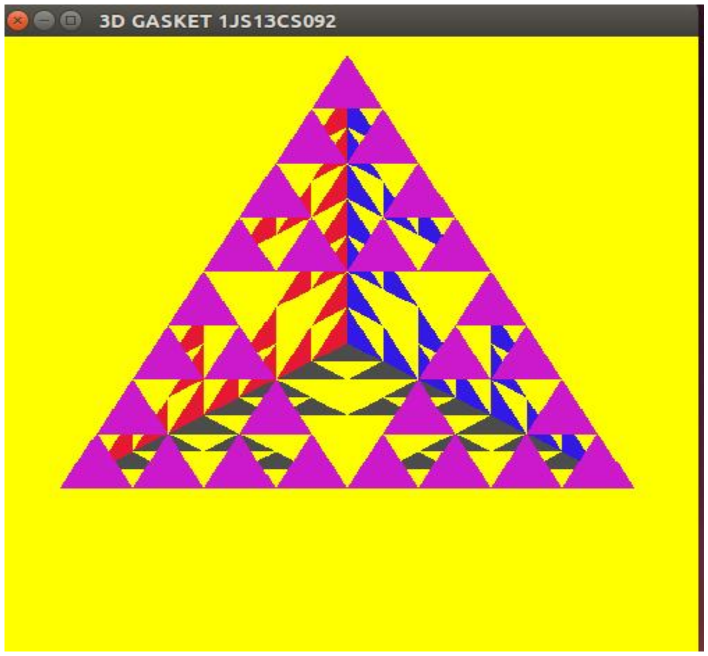
To create file - `gedit filename.c`

To compile file - `gcc filename.c -lGL -lGLU -lglut`

To execute - `./a.out`



Sierpinski gasket (front view)



Sierpinski gasket (rear view)