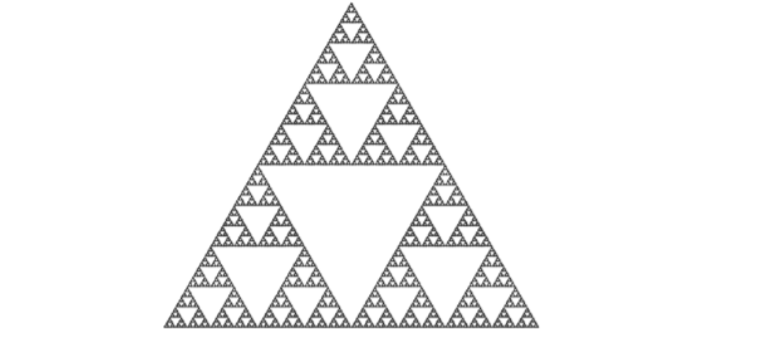
**VIVA PROGRAMS**

**1.** 

**#include <windows.h>**

**#include <GL/glut.h>**

**#include <cstdlib>**

**#include <ctime>**

**#define MENU\_SMOOTH 1**

**#define MENU\_FLAT 0**

**void generateColors();**

**double random(bool reset = false);**

**void drawPyramid(GLfloat \*a, GLfloat \*b, GLfloat \*c, GLfloat \*d);**

**void dividePyramid(GLfloat \*a, GLfloat \*b, GLfloat \*c, GLfloat \*d, int iteraciones);**

**void keyboard(unsigned char key, int x, int y);**

**void special(int key, int x, int y);**

**void mouse(int button, int state, int x, int y);**

**void menu(int item);**

**void display();**

**void doZoom();**

**void init();**

**GLfloat pyramid[4][3] = {{-1.0, -1.0, -1.0},**

**{ 1.0, -1.0, -1.0},**

**{ 0.0, -1.0, 0.732},**

**{ 0.0, 1.0, -0.134}};**

**double colors[1000] = {0};**

**double angle = 0;**

**int iterations = 5;**

**double zoom = 0;**

**int shading = GL\_SMOOTH;**

**void generateColors() {**

**for (int i = 0; i < 1000; i++) {**

**colors[i] = rand() / (double) RAND\_MAX;**

**}**

**}**

**double random(bool reset) {**

**static int curr = 0;**

**if (reset) {**

**curr = 0;**

**return 0.0;**

**} else {**

**if (curr >= 1000) curr = 0;**

**return colors[curr++];**

**}**

**}**

**void drawPyramid(GLfloat \*a, GLfloat \*b, GLfloat \*c, GLfloat \*d) {**

**glPolygonMode(GL\_FRONT\_AND\_BACK,GL\_FILL);**

**glShadeModel(shading);**

**// bottom**

**glBegin(GL\_TRIANGLES);**

**//en este caso los vértices están dados en el sentido de las manecillas del reloj**

**glColor3f(random(), random(), random());**

**glVertex3fv(a);**

**glColor3f(random(), random(), random());**

**glVertex3fv(b);**

**glColor3f(random(), random(), random());**

**glVertex3fv(c);**

**glEnd();**

**glBegin(GL\_TRIANGLES);**

**glColor3f(random(), random(), random());**

**glVertex3fv(a);**

**glColor3f(random(), random(), random());**

**glVertex3fv(b);**

**glColor3f(random(), random(), random());**

**glVertex3fv(d);**

**glEnd();**

**glBegin(GL\_TRIANGLES);**

**glColor3f(random(), random(), random());**

**glVertex3fv(a);**

**glColor3f(random(), random(), random());**

**glVertex3fv(c);**

**glColor3f(random(), random(), random());**

**glVertex3fv(d);**

**glEnd();**

**glBegin(GL\_TRIANGLES);**

**glColor3f(random(), random(), random());**

**glVertex3fv(b);**

**glColor3f(random(), random(), random());**

**glVertex3fv(c);**

**glColor3f(random(), random(), random());**

**glVertex3fv(d);**

**glEnd();**

**}**

**void dividePyramid(GLfloat \*a, GLfloat \*b, GLfloat \*c, GLfloat \*d, int iteraciones) {**

**GLfloat v[6][3];**

**int j;**

**if (iteraciones > 0) {**

**//encontrar los puntos medios de cada arista del triángulo**

**for (j = 0; j < 3; j++) {**

**v[0][j] = (a[j] + b[j]) / 2;**

**}**

**for (j = 0; j < 3; j++) {**

**v[1][j] = (a[j] + c[j]) / 2;**

**}**

**for (j = 0; j < 3; j++) {**

**v[2][j] = (b[j] + c[j]) / 2;**

**}**

**for (j = 0; j < 3; j++) {**

**v[3][j] = (a[j] + d[j]) / 2;**

**}**

**for (j = 0; j < 3; j++) {**

**v[4][j] = (b[j] + d[j]) / 2;**

**}**

**for (j = 0; j < 3; j++) {**

**v[5][j] = (c[j] + d[j]) / 2;**

**}**

**dividePyramid( a , v[0], v[1], v[3], iteraciones-1);**

**dividePyramid(v[0], b , v[2], v[4], iteraciones-1);**

**dividePyramid(v[1], v[2], c , v[5], iteraciones-1);**

**dividePyramid(v[3], v[4], v[5], d , iteraciones-1);**

**//divideTriangle(v[0], v[1], v[2], iteraciones-1);**

**} else {**

**//dibujar el triángulo de la iteración 0**

**drawPyramid(a, b, c, d);**

**}**

**}**

**void keyboard(unsigned char key, int x, int y) {**

**switch (key) {**

**case '+':**

**if (iterations < 10) iterations += 1;**

**display();**

**break;**

**case '-':**

**if (iterations > 0) iterations -= 1;**

**display();**

**break;**

**case 'q':**

**exit(0);**

**break;**

**}**

**}**

**void special(int key, int x, int y) {**

**switch (key) {**

**case GLUT\_KEY\_UP:**

**if (zoom - 0.1 > -1) zoom -= 0.05;**

**doZoom();**

**break;**

**case GLUT\_KEY\_DOWN:**

**zoom += 0.05;**

**doZoom();**

**break;**

**}**

**}**

**void mouse(int button, int state, int x, int y) {**

**if (state == GLUT\_UP) {**

**generateColors();**

**display();**

**}**

**}**

**void menu(int item) {**

**switch (item) {**

**case MENU\_FLAT:**

**shading = GL\_FLAT;**

**display();**

**break;**

**case MENU\_SMOOTH:**

**shading = GL\_SMOOTH;**

**display();**

**break;**

**}**

**}**

**void doZoom() {**

**glMatrixMode(GL\_PROJECTION);**

**glLoadIdentity();**

**glOrtho(-1.0 - zoom, 1.0 + zoom, -1.0 - zoom, 1.0 + zoom, -20.0, 20.0);**

**glMatrixMode(GL\_MODELVIEW);**

**display();**

**}**

**void display() {**

**glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);**

**glPushMatrix();**

**glRotatef(angle, 0, 1, 0);**

**random(true);**

**dividePyramid(pyramid[0], pyramid[1], pyramid[2], pyramid[3], iterations);**

**glPopMatrix();**

**glFlush();**

**glutPostRedisplay();**

**}**

**void init() {**

**glClearColor(0.0, 0.0, 0.0, 1.0);**

**glColor3f(0.0, 0.0, 0.0);**

**glMatrixMode(GL\_PROJECTION);**

**glLoadIdentity();**

**glOrtho(-2.0, 2.0, -2.0, 2.0, -20.0, 20.0);**

**glEnable(GL\_DEPTH\_TEST);**

**}**

**void idle() {**

**angle += 0.01;**

**if (angle > 360) angle = 0;**

**}**

**int main(int argc, char\*\* argv) {**

**srand(time(NULL));**

**generateColors();**

**glutInitDisplayMode(GLUT\_RGB);**

**glutInitWindowSize(600, 600);**

**glutInitWindowPosition(0, 0);**

**glutCreateWindow("Sierpinski Triangle");**

**glutPositionWindow(100, 100);**

**glutKeyboardFunc(keyboard);**

**glutSpecialFunc(special);**

**glutMouseFunc(mouse);**

**glutCreateMenu(menu);**

**glutAddMenuEntry("Smooth shading", MENU\_SMOOTH);**

**glutAddMenuEntry("Flat shading", MENU\_FLAT);**

**glutAttachMenu(GLUT\_RIGHT\_BUTTON);**

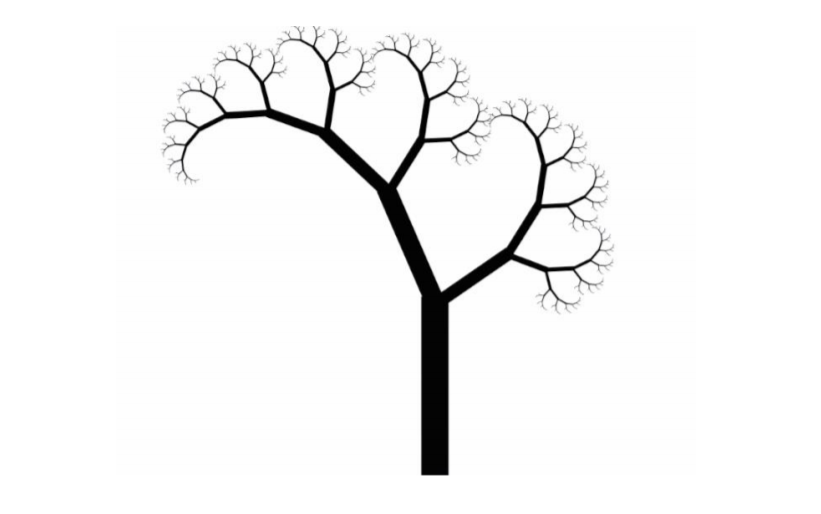
**init();**

**glutDisplayFunc(display);**

**glutIdleFunc(idle);**

**glutMainLoop();**

**}**

**2.** 

**#include <GL/glut.h>**

**#include <stdlib.h>**

**void drawtree(int n)**

**{**

**if(n>0)**

**{**

**glPushMatrix();**

**glTranslatef(-0.5,1.0,0);**

**glRotatef(45, 0.0, 0.0, 1.0);**

**glScalef(0.707,0.707,0.707);**

**drawtree(n-1);**

**glPopMatrix();**

**glPushMatrix();**

**glTranslatef(0.5,1.0,0);**

**glRotatef(-45, 0.0, 0.0, 1.0);**

**glScalef(0.707,0.707,0.707);**

**drawtree(n-5);**

**glPopMatrix();**

**glutSolidCube(1);**

**}**

**}**

**void treeInit(int n)**

**{**

**glColor3f(0.0, 0.0, 0.0);**

**drawtree(n);**

**}**

**void display (void)**

**{**

**glClearColor (1.0,1.0,1.0,1.0);**

**glClear (GL\_COLOR\_BUFFER\_BIT);**

**glLoadIdentity();**

**gluLookAt (0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);**

**treeInit(15);**

**glFlush();**

**}**

**void reshape (int w, int h)**

**{**

**glViewport (0, 0, (GLsizei)w, (GLsizei)h);**

**glMatrixMode (GL\_PROJECTION);**

**glLoadIdentity ();**

**gluPerspective (80, (GLfloat)w / (GLfloat)h, 1.0, 100.0);**

**glMatrixMode (GL\_MODELVIEW);**

**}**

**int main (int argc, char \*\*argv)**

**{**

**glutInit (&argc, argv);**

**glutInitDisplayMode (GLUT\_SINGLE);**

**glutInitWindowSize (900, 700);**

**glutInitWindowPosition (300, 100);**

**glutCreateWindow ("ABHISHEK SHARMA");**

**glutDisplayFunc (display);**

**glutReshapeFunc (reshape);**

**glutMainLoop ();**

**return 0;**

**}**