Day4: Sierpinski2D OpenGl

#ifdef \_\_APPLE\_CC\_\_

#include <GLUT/glut.h>

#else

#include <GL/glut.h>

#endif

#include <cstdlib>

// A simple two-dimensional point class to make life easy.  It allows you to

// reference points with x and y coordinates instead of array indices) and

// encapsulates a midpoint function.

struct Point {

  GLfloat x, y;

  Point(GLfloat x = 0, GLfloat y = 0): x(x), y(y) {}

  Point midpoint(Point p) {return Point((x + p.x) / 2.0, (y + p.y) / 2.0);}

};

// Draws a Sierpinski triangle with a fixed number of points. (Note that the

// number of points is kept fairly small because a display callback should

// NEVER run for too long.

void display() {

  glClear(GL\_COLOR\_BUFFER\_BIT);

  static Point vertices[] = {Point(0, 0), Point(200, 500), Point(500, 0)};

  // Compute and plot 100000 new points, starting (arbitrarily) with one of

  // the vertices. Each point is halfway between the previous point and a

  // randomly chosen vertex.

  static Point p = vertices[0];

  glBegin(GL\_POINTS);

  for (int k = 0; k < 100000; k++) {

    p = p.midpoint(vertices[rand() % 3]);

    glVertex2f(p.x, p.y);

  }

  glEnd();

  glFlush();

}

// Performs application-specific initialization. Sets colors and sets up a

// simple orthographic projection.

void init() {

  // Set a deep purple background and draw in a greenish yellow.

  glClearColor(0.25, 0.0, 0.2, 1.0);

  glColor3f(0.6, 1.0, 0.0);

  // Set up the viewing volume: 500 x 500 x 1 window with origin lower left.

  glMatrixMode(GL\_PROJECTION);

  glLoadIdentity();

  glOrtho(0.0, 500.0, 0.0, 500.0, 0.0, 1.0);

}

// Initializes GLUT, the display mode, and main window; registers callbacks;

// does application initialization; enters the main event loop.

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

  glutInit(&argc, argv);

  glutInitDisplayMode (GLUT\_SINGLE | GLUT\_RGB);

  glutInitWindowSize(500, 500);

  glutInitWindowPosition(40, 40);

  glutCreateWindow("Sierpinski Triangle");

  glutDisplayFunc(display);

  init();

  glutMainLoop();

}

Output:

