CS315 – Introduction to Computer Graphics
Winter, 2021

Assignment 1

Assigned Date: Friday, January 22, 2021
Due Date: Friday, February 5, 2021

1. (Problem 2.1, page 115, with slight modification)

A slight variation on generating the Sierpinski gasket with triangular polygons yields the fractal mountains used in computer-generated animations. After you find the midpoint of each side of the triangle, perturb this location before subdivision.

[Later if you are interested in it, you can do it in three dimensions and add shading. After a few subdivisions, you may generate sufficient details that your triangles look like a mountain.]

Hint:

First, you should have a good understanding to "gasket2.html" and "gasket2.js", one of the examples in Chapter 2. The two files should be available from your Lab 1. I also provide them in UR Courses/CS315 for your convenience. This question is based on this example.

Perturbing a point is to displace the point in a random direction with a random distance. Given a point $P = (x, y)$, this can be achieved by generating two random values, say $(dx, dy) = D$, and use it as the displacement:

$$P' = P + D = (x + dx, y + dy).$$

In JavaScript, the Math object has a method, called random( ), that can be used to generate a random number from 0 (inclusive) up to but not including 1 (exclusive). That is, the statement

```
x = Math.random( );
```

could, for example, return a result like 0.3974578890176147.

A small JavaScript testing demo for this method is a good tutorial that can be found at:

http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_random

If you would like to generate an integer random number between 1 and 100, the following statement will do it:

*i = Math.floor((Math.random() * 100) + 1);*

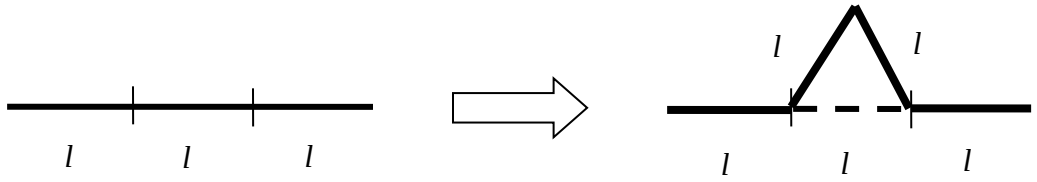
where the random number (in the range [0.0, 1.0]) is scaled by 100 (and plus 1), then being truncated by *Math.floor( )* to become an integer.

In your program for this problem, you may need a float random value whose magnitude is less than *a,* a pre-specified positive float value, or in other words, a random value in the range of (-a, a).

In this problem, the amount of displacement should also be controlled within a proper range at each recursion level respectively. That is, when you find the midpoint between the two points *A* and *B*, you should compute the distance, *L*, between *A* and *B*. The maximum displacement should be proportional to *L*. I suggest you set the range of the random value to be *L/K*, where *K* is between 4 and 8. You may try different values for *K* to see their effects on the generated pictures. Please also note that, *dx* and *dy* could be positive or negative; otherwise, the displacement will be always in the same direction.

2. (Problem 2.6, page 116)

Space-filling curves have interested mathematicians for centuries. In the limit, these curves have infinite length, but they are confined to a finite rectangle and never cross themselves. Many of these curves can be generated iteratively. Consider the "rule" pictured in Figure 2.48 (page 111) that replaces a single line segment with four short segments. Write a program that starts with a triangle and iteratively applies the replacement rule to all the line segments. The object that you generate is called the Koch snowflake.



The initial triangle is the start level 0. Try 1, 3, and 5 levels of recursive replacements respectively.

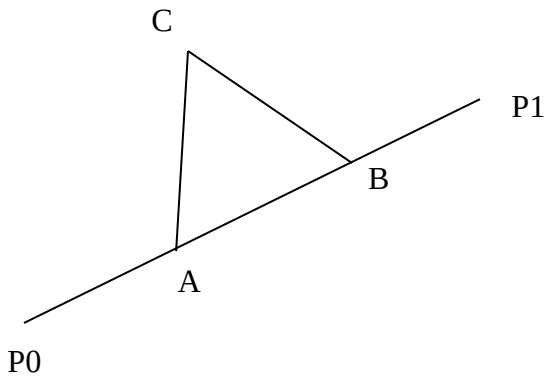
Hint:

To get started with this question, I suggest you first to modify "gasket2.html" and "gasket2.js" such that it simply draws the initial triangle in line style.

If you have difficulty to do this, I have provided them as "gasket2a.html" and "gasket2a.js" for your reference.

Next, you consider replacing each line with 4 short lines using the rule provided in the question. Each line is defined by two points. Therefore, this step requires you to find 3 new points in addition to the 2 given points.

Specifically, given two points, P0 = (x0, y0), and P1 = (x1, y1):



(a) We need first to find the coordinates for the points A and B respectively, such that A and B divide the line segment (P0 –> P1) evenly into three sections.

(b) Then, we find the coordinate of the point C, such that the triangle A-B-C has an equal side length. (The point C should be on the left-hand side if you "walk" from P0 to P1.)

To derive the points A, B, and C, you need some knowledge learned from Calculus. If you do not know how to derive them now, you can click the link "findABC" which provides two solutions to this problem.

General Rules for the Assignments:

1. Discussion among the students is encouraged for effective learning. However, sharing answers (including program codes in this case) is strictly prohibited.
2. Assignments should be submitted in electronic form to the UR Courses – CS 315 before the closing time on the due day. Any extension must have the instructor's permission in advance.
3. All programs must be well-documented. A general rule is that, the marker should be able to understand your program by reading only the comments. Any non-trivial statements must have in-line comments.
4. Your submission should include the source programs and the generated pictures (screen shots).
5. All documents should be zipped into a single file for submission.