

## Week 7 Practical Class Exercises

### Fractals

---

#### Objectives

This week's exercises will give you some more exposure to the graphics package discussed in lectures. You are to develop definitions to draw some different fractals. The emphasis is not so much on graphics *per se*, but rather on the user-defined algebraic types of the graphics package and on the recursive structure of the functions corresponding to the fractal definitions.

#### Assignment 1 Demonstration

During this week's practical class, you are to demonstrate your ppm image manipulation functions to your tutor. This is a part of the assessment for Assignment 1, so it is essential that you attend your practical class to obtain any marks for this component.

#### Using the ANU Graphics Library

The ANU Graphic Library is not installed with the GHC libraries, so it is most convenient if you need to put it in the same directory as the scripts you are developing. For example, if your working directory is `comp1100/week7/`, then you should copy the entire contents of the directory `/dept/dcs/comp1100/public/ANUPlot/Examples/`, including all subdirectories (`Graphics/` and `textures/`) into your working directory.

There are various examples in the directory, including the snowflake fractal `Snowflake.hs`. You can compile the programs, or load them into GHCi and evaluate `main`. The graphics library uses some non-standard extensions to Haskell, so the command must be

```
ghci -fglasgow-exts Snowflake.hs
```

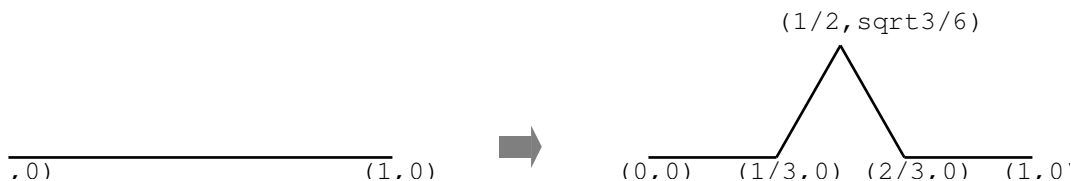
or

```
ghc -fglasgow-exts --make Snowflake.hs -o snowflake
```

#### Snowflake Fractal

The snowflake curve is an example of a *fractal* shape. A fractal is defined by a rule that defines how to construct the shape from smaller, simpler copies of itself. The *degree* of a fractal indicates the complexity of the fractal. Degree 0 is the basic shape and degree  $(n+1)$  is derived from degree  $n$  by replacing each 'part' by another copy of the degree  $n$  fractal, possibly scaled down in some way. To see what I mean, run the `Snowflake` program and try degrees 0, 1 and 2.

The rule for a side of the snowflake curve can be shown by the following figure:

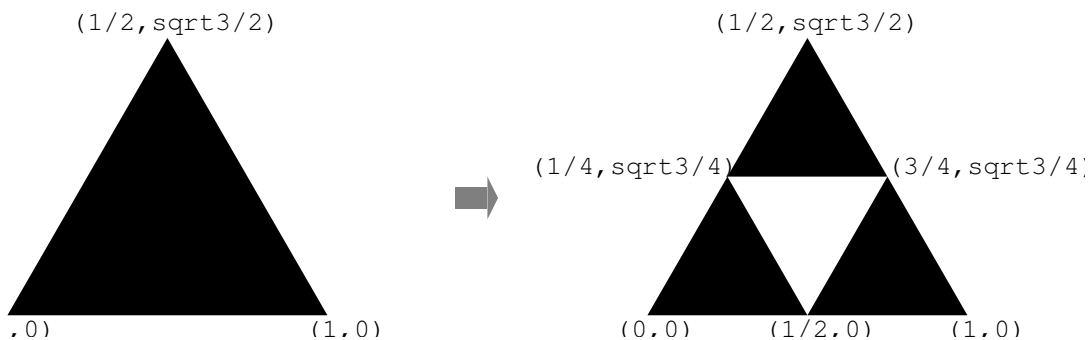


Snowflake Curve Fractal Rule

That is, each side is drawn by making four smaller copies of itself and arranging them as shown above.

**Exercise 1 (Sierpinski Triangle)**

The *Sierpinski triangle* is a fractal constructed from an equilateral triangle. The starting point is simply a filled-in equilateral triangle. The construction of the degree  $(n + 1)$  Sierpinski triangle is to arrange 3 copies of the degree  $n$  triangle as shown in the following diagram.

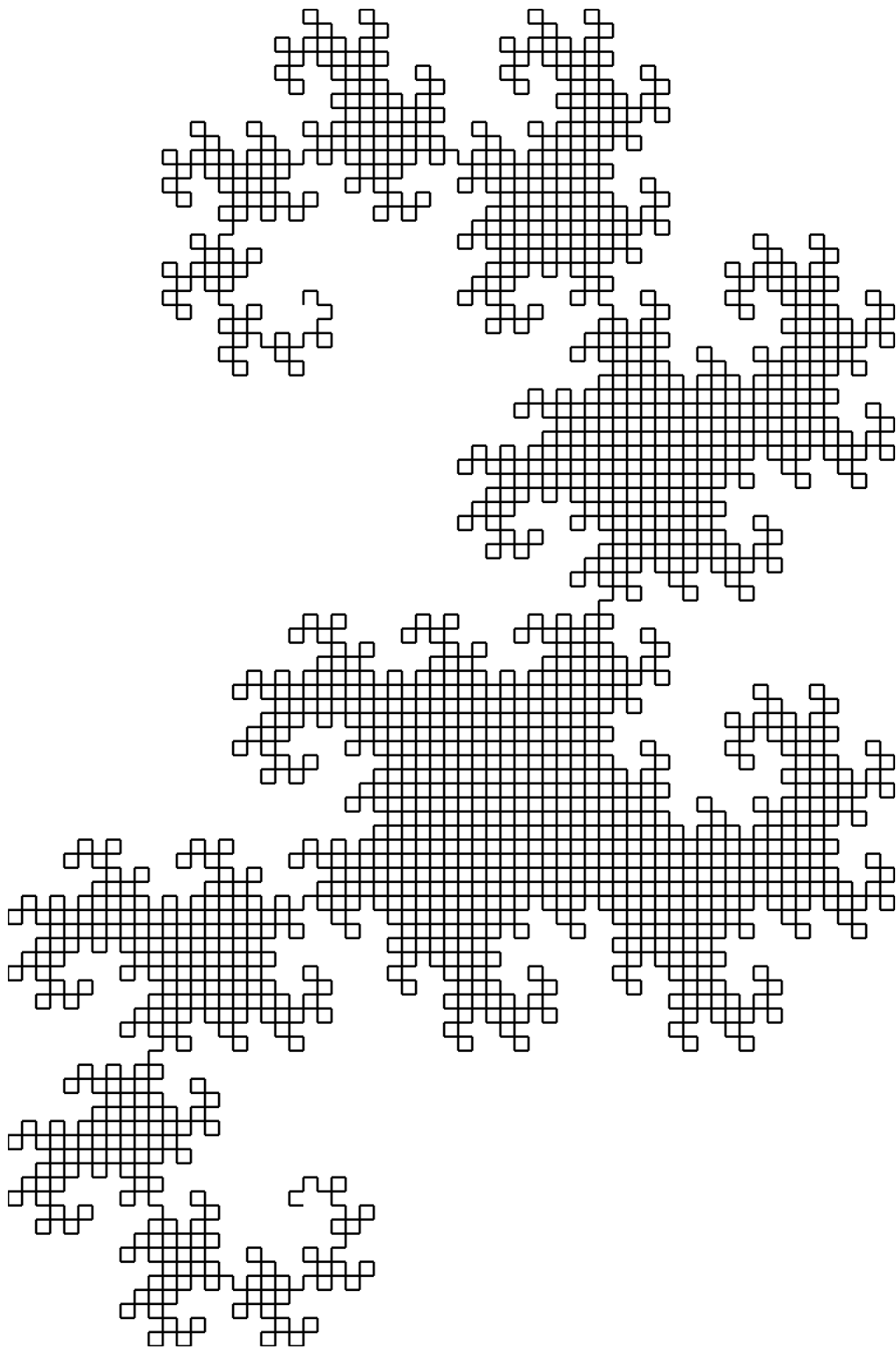


Sierpinski Triangle Rule

The degree 5 Sierpinski triangle can be viewed from the course website.

Based on the `Snowflake.hs` program, write a similarly structured program to draw Sierpinski triangles. Your fractal function, say `sierpinski :: Int -> Picture` should take as argument a degree and produces as result a picture of the Sierpinski triangle of that degree. To make it a reasonable size, begin with a triangle whose side length is about 400.





Dragon Curve — Degree 6