Week 11 - Friday

COMP 1800

Last time

- What did we talk about last time?
- Recursion

Questions?

Assignment 8

Exam 2 Post Mortem

Recursion

To understand recursion, you must first understand recursion.

Useful Recursion

Two parts:

- Base case(s)
  - Tells recursion when to stop
  - For factorial, $n = 1$ or $n = 0$ are examples of base cases
- Recursive case(s)
  - Allows recursion to progress
  - "Leap of faith"
  - For factorial, $n > 1$ is the recursive case

Complex shapes

- Many natural things have recursive shapes:
  - Trees
  - Spiral shells
  - Blood vessels
  - Mountains
  - Snowflakes
- Using recursion, we can draw some complex, organic-looking shapes with only a little code

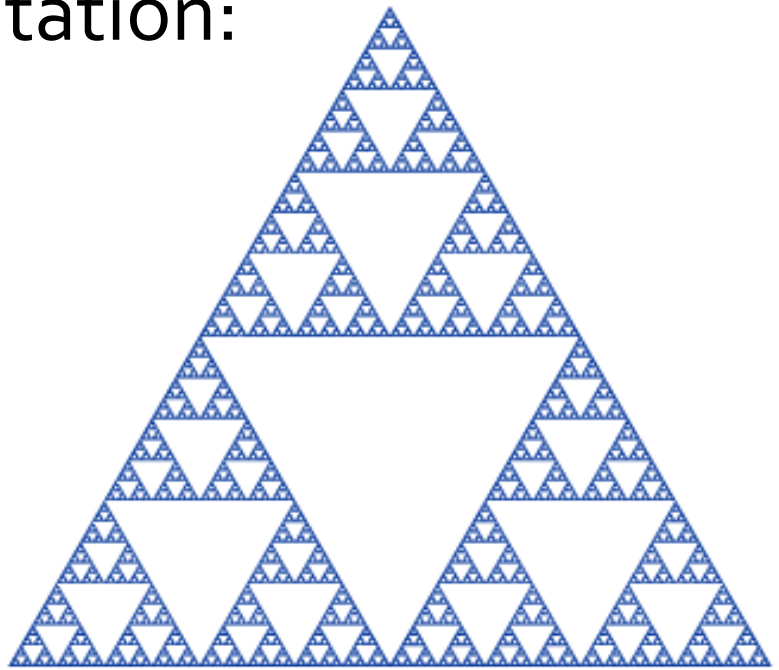
Drawing a triangle

- The following function draws a triangle, using the three points (which are given as lists that contain two elements each: an *x* and a *y* value)

```
def drawTriangle(yertle, p1, p2, p3):
    yertle.up()
    yertle.goto(p1)
    yertle.down()
    yertle.goto(p2)
    yertle.goto(p3)
    yertle.goto(p1)
```

Sierpinski triangle

- The Sierpinski triangle draws a triangle with other smaller triangles inside of it
- Here's a representation:



Steps along the way

- To make a Sierpinski triangle, we need another function that finds the midpoint between two points
- This function takes two points (both lists of two numbers) and makes a new point whose first value is the average of the first values from the original points and whose second value is the average of the second values from the original points

```
def mid(p1, p2):
    return ((p1[0] + p2[0])/2, (p1[1] + p2[1])/2)
```

Recursion for Sierpinski

- We will use a recursive depth that keeps getting smaller until we decide to stop drawing triangles
- Base case (Depth is 0):
  - Draw a triangle with the given corner points
- Recursive case (Depth > 0):
  - Make a Sierpinski triangle at point 1, the midpoint of point 1 and point 2, and the midpoint of point 1 and point 3, at a depth one level lower
  - Make a Sierpinski triangle at point 2, the midpoint of point 2 and point 3, and the midpoint of point 2 and point 1, at a depth one level lower
  - Make a Sierpinski triangle at point 3, the midpoint of point 3 and point 1, and the midpoint of point 3 and point 2, at a depth one level lower

Sierpinski function

- Here is that function implemented in Python:

```
def sierpinski(yertle, p1, p2, p3, depth):
    if depth > 0:
        sierpinski(yertle, p1, mid (p1, p2), mid(p1, p3), depth - 1)
        sierpinski(yertle, p2, mid(p2, p3), mid(p2, p1), depth - 1)
        sierpinski(yertle, p3, mid(p3, p1), mid(p3, p2), depth - 1)
    else:
        drawTriangle(yertle, p1, p2, p3)
```

- A nice looking call that covers most of the turtle drawing space is:

```
sierpinski(yertle, [-225, -250], [225, -250], [0, 225], 5)
```

Quiz

Work Time

Upcoming

Next time…

- Objects

Reminders

- Reader 10.1 through 10.5
- **Work on Assignment 8!**
  - **It's hard.**