Class 6: Recursion
Recursive Structures

- A recursive structure is one in which part of the structure resembles the whole thing.

Examples:
Recursive Function Definitions

A recursive function definition is a function definition in which an *application of the function itself* makes up part of its definition, i.e. the function is defined in terms of itself.
Computing Factorial

- **Example:** What is \( n \) factorial?
  - \( n! = 1 \times 2 \times 3 \times \ldots \times n \)

- **Recursive definition of factorial:**
  - \( n! = \begin{cases} 
    1 & \text{if } n = 0 \\
    n \times (n-1)! & \text{if } n > 0
  \end{cases} \)

  - base case
  - recursive case

  \( n! \) is defined in terms of \((n-1)!\)!
A recursive algorithm is an algorithm whose definition involves calling itself (with “simpler” or “smaller” parameters)

Example:

```python
def factorial(n):
    if n == 0:
        return 1  # base case
    else:
        return n * factorial(n-1)  # recursive case
```
Example: Computing 6!

- 6!
- 6 * 5!
- 6 * (5 * 4!)
- 6 * (5 * (4 * 3!))
- 6 * (5 * (4 * (3 * 2!)))
- 6 * (5 * (4 * (3 * (2 * 1!)))))
- 6 * (5 * (4 * (3 * (2 * (1 * 0!)))))
- 6 * (5 * (4 * (3 * (2 * (1 * 1)))))
- 6 * (5 * (4 * (3 * (2 * 1))))
- 6 * (5 * (4 * (3 * 2)))
- 6 * (5 * (4 * 6))
- 6 * (5 * 24)
- 6 * 120
- 720
Creating a recursive solution

Base case:
- A trivial and easily solvable instance of the problem

Recursive case:
- Break the problem up into solvable problems and smaller versions of the same problem [*must make progress toward the base case*]
- Make the problem smaller by looking at smaller numbers, less data, or fewer choices
- Figure out how to combine the solutions to smaller problems to get the solution to the overall problem
Some Turtle pictures...
Koch Curves
The Koch Snowflake
Towers of Hanoi

Move $n$ disks from pole A to pole B:

- move 1 disk at a time
- never place a larger disk on top of a smaller one
- use the extra pole for “temporary storage”