Recursion

- The divide-and-conquer strategy uses this simple concept to solve hard problems.

- Recursion is where a function calls itself.

- Recursion is used when it makes the solution clearer. There’s no performance benefit to using recursion; in fact, loops are sometimes better for performance. I like this quote by Leigh Caldwell on Stack Overflow:

  > "Loops may achieve a performance gain for your program. Recursion may achieve a performance gain for your programmer. Choose which is more important in your situation!"

- Because a recursive function calls itself, it's easy to write a function incorrectly that ends up in an infinite loop:

  ```
  def countdown(i)
    print i
    countdown(i-1)
  end
  ```

- When you write a recursive function, you have to tell it when to stop recursing. That’s why every recursive function has two parts: the base case, and the recursive case. The recursive case is when the function calls itself. The base case is when the function doesn’t call itself again, so it doesn’t go into an infinite loop.

- Let’s add a base case to the countdown function:

  ```
  def countdown(i)
    print i

    if i <= 0
      return
    else
      countdown(i-1)
    end
  end
  ```