The Beauty & Joy of Computing

Lecture #8
Recursion

GO SEE INCEPTION!
The coolest movie last year highlights recursion, and it was up for best picture. If you haven’t seen it yet, you should, because it will help you understand recursion!!

You can clobber Quest with Midterm!

New Rule: Use scratch paper in lab! The problems there are hard enough that you won’t be able to keep it in your head!
Overview

- Recursion
  - Demo
    - Vee example & analysis
    - Downup
  - You already know it
  - Definition
  - Trust the Recursion!
  - Conclusion

M. C. Escher: Drawing Hands
”I understood Vee & Downup“

a) Strongly disagree
b) Disagree
c) Neutral
d) Agree
e) Strongly agree

M. C. Escher : Fish and Scales
Definition

- Recursion: (noun) See recursion. 😊

- An algorithmic technique where a function, in order to accomplish a task, calls itself with some part of the task.

- Recursive solutions involve two major parts:
  - Base case(s), the problem is simple enough to be solved directly
  - Recursive case(s). A recursive case has three components:
    - Divide the problem into one or more simpler or smaller parts
    - Invoke the function (recursively) on each part, and
    - Combine the solutions of the parts into a solution for the problem.

- Depending on the problem, any of these may be trivial or complex.
You already know it!

There is a little gray house.
And inside the little gray house
There is a little yellow house.
And inside the little yellow house
There is a little white house.
And inside the little white house
There is a little red heart.
And inside the little red heart
There is a little yellow house.
And inside the little yellow house
There is a little white house.
And inside the little white house
There is a little red heart.
And inside the little red heart
There is a little yellow house.
And inside the little yellow house
There is a little white house.
And inside the little white house
There is a little red heart.
And inside the little red heart
There is a little yellow house.
And inside the little yellow house
There is a little white house.
And inside the little white house
There is a little red heart.
And inside the little red heart
There is a little yellow house.
And inside the little yellow house
There is a little white house.
And inside the little white house
There is a little red heart.

\[ n! = n \times (n-1)! \]

A King is a Son of a King

If all were one
What a great tree would be!
And if all were one
What a great man would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
And if all were one
What a great tree would be!
And if all were one
What a great king would be!
And if all were one
What a great man would be!
Trust the Recursion

- **When authoring recursive code:**
  - The base is usually easy: “when to stop?”
  - In the recursive step
    - How can we break the problem down into two:
      - A piece I can handle right now
      - The answer from a smaller piece of the problem
    - Assume your self-call does the right thing on a smaller piece of the problem
    - How to combine parts to get the overall answer?

- Practice will make it easier to see idea
Recursion is not iteration (i.e., loops)

Almost always, writing a recursive solution is more powerful than an iterative one.

- More powerful than, easier
- Just as powerful as, easier
- More powerful than, harder
- Just as powerful as, harder

Sanity Check...

http://xkcd.com/244/

YOUR PARTY ENTERS THE TAVERN.

I GATHER EVERYONE AROUND A TABLE. I HAVE THE ELVES START WHITTLING DICE AND GET OUT SOME PARCHMENT FOR CHARACTER SHEETS.

HEY, NO RECURSING.
Summary

- Behind Abstraction, Recursion is probably the 2nd biggest idea about programming in this course.
- It’s tremendously useful when the problem is self-similar.
- It’s no more powerful than iteration, but often leads to more concise & better code.

http://www.dominiek.eu/blog/?m=200711