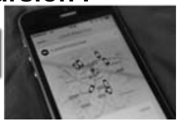
The Beauty and Joy of Computing

Lecture #9 Recursion I

UC Berkeley
EECS Lecturer
Pierce Vollucci

Use scratch paper in lab!



MICROTRANSACTIONS: YAY OR NAY?

The FTC is currently in negotiations with several app providers (Apple, Amazon, Google) regarding the truth behind "free" games. Industry sees both good and bad effects of microtransactions.

http://www.nytimes.com/2014/07/06/technology/free-video-games-say-pay-up-or-wait-testing-players-patience.html?ref=technology

TAXI COMPETITOR USES SMARTPHONES OVER METERS

Uber is now allowed to operate in London despite stringent requirements of the famous black taxis due to their use of smartphones for meters. Pros and cons?

http://bits.blogs.nytimes.com/2014/07/03/london-transport-regulator-says-uber-can-legally-operate/?ref=technology

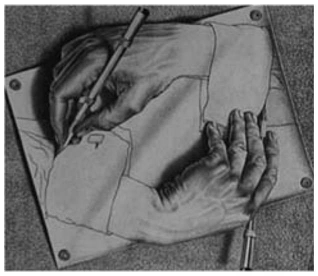
---

www.worldofescher.com/gallery/A13.html

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*

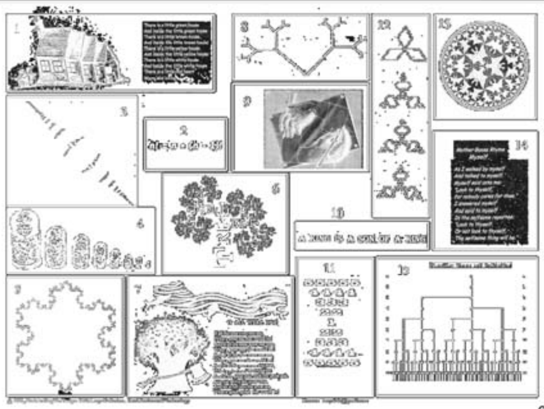
---

www.catb.org/~esr/jargon/html/R/recursion.html
www.nist.gov/dads/HTML/recursion.html

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!



Trust the Recursion

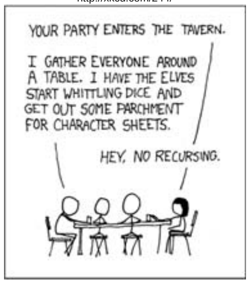
- When authoring recursive code:
  - The base is usually easy: "when to stop?"
  - In the recursive step
    - How can we break the problem down into two:
      - A piece I can handle right now
      - The answer from a smaller piece of the problem
    - Assume your self-call does the right thing on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

Sanity Check…

- Recursion is ■ Iteration (i.e., loops)
- Almost always, **writing a recursive solution is ◆ than an iterative one**

a) more powerful than, easier
b) just as powerful as, easier
c) more powerful than, harder
d) just as powerful as, harder

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 can sometimes lead to more concise & better code

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