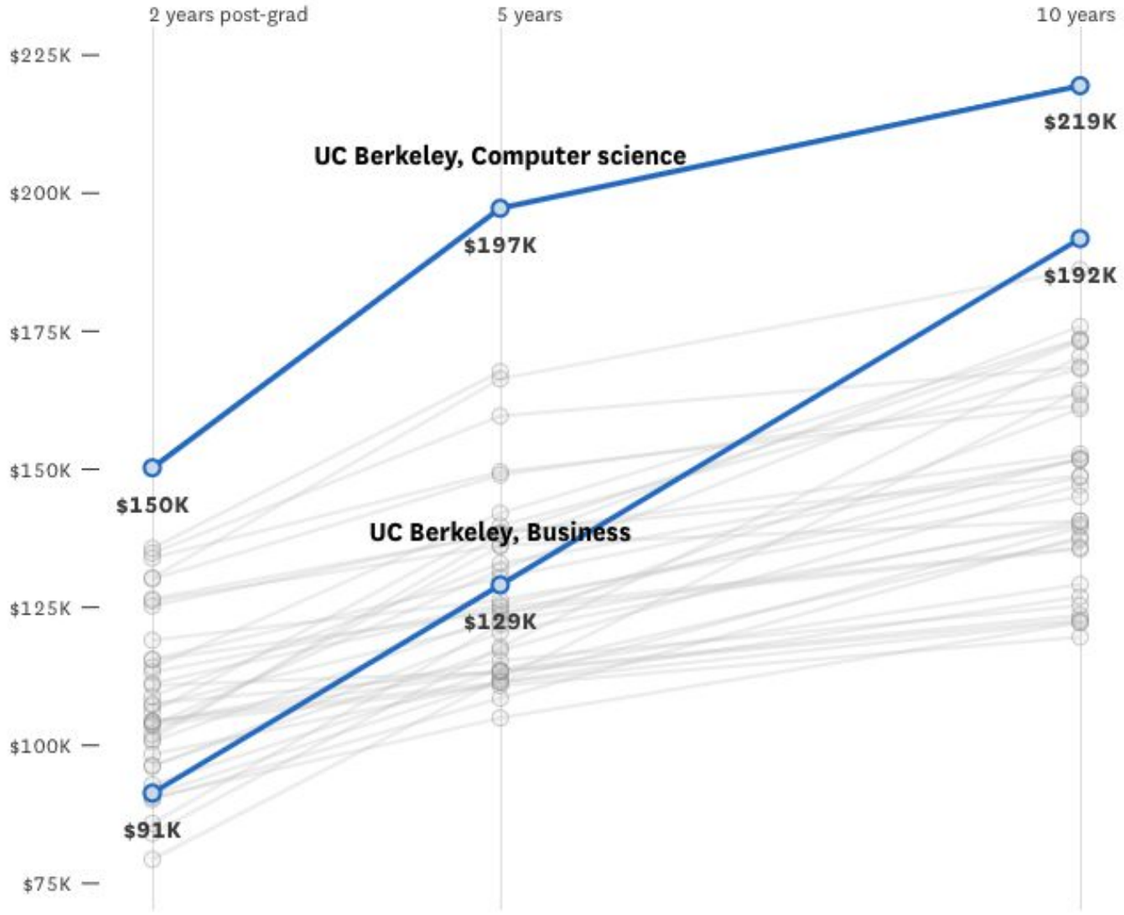
These are the college majors that lead to the best-paying jobs for UC and CSU graduates

Fields of study and campuses that yield the highest median annual earnings after graduation among UC and CSU undergraduate alumni working in California

Filter by campus: All campuses



Put your notes here

CS10 NEWS

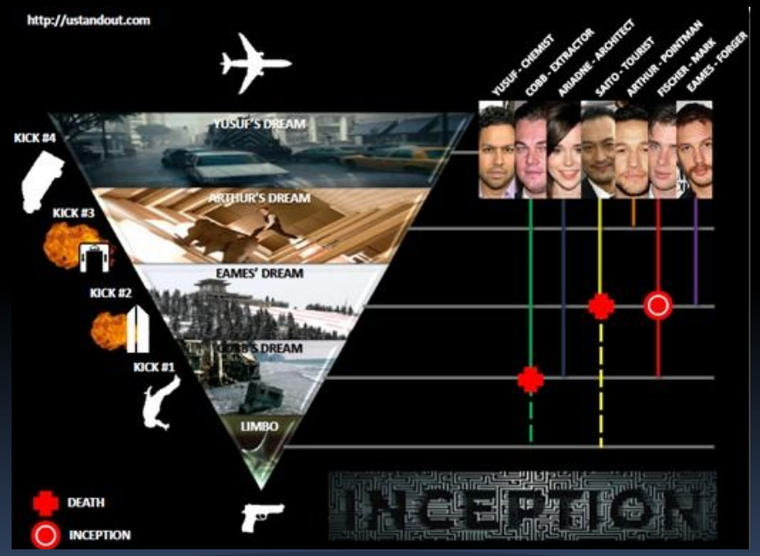
UC Berkeley
Teaching Professor
Dan Garcia

The Beauty and Joy of Computing

Recursion I

Go see “Inception”!

The coolest movie a few years ago highlights <u>recursion</u>, and it was up for best picture. If you haven’t seen it yet, you should, because it will help you understand recursion!!



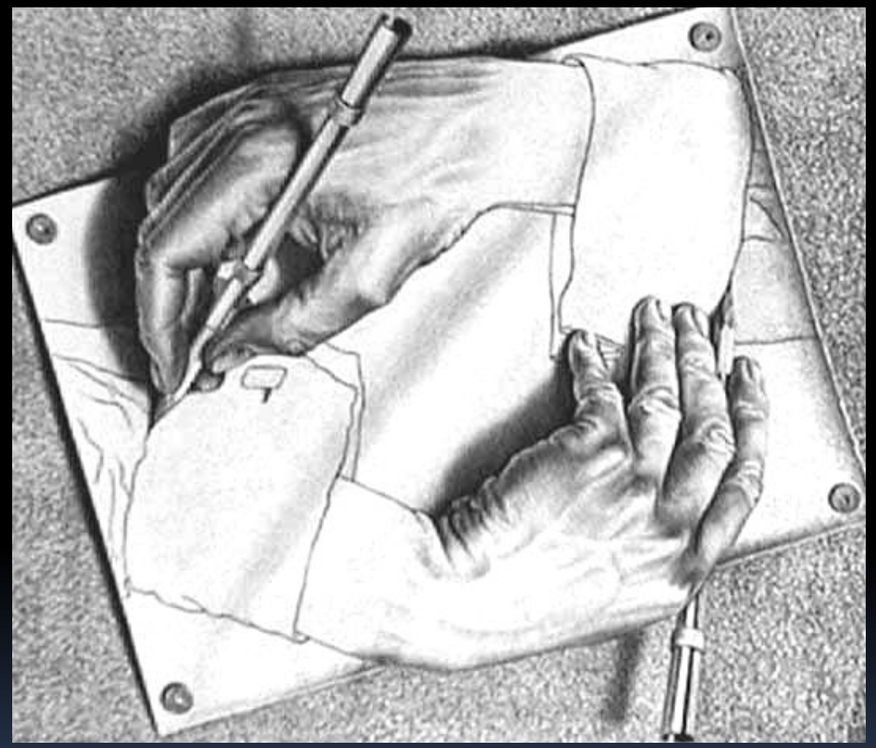
New Rule: Use scratch paper in lab!
The problems are hard enough that you won’t be able to keep it in your head!

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*



Recursion: Vee Demo

tinyurl.com/veeapp

Recursion: Downup Demo

“I Understood Vee & Downup”

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

M. C. Escher : Fish and Scales

When poll is active, respond at **pollev.com/ddg**

Text **DDG** to **22333** once to join

L10a "I understood Vee and Downup"

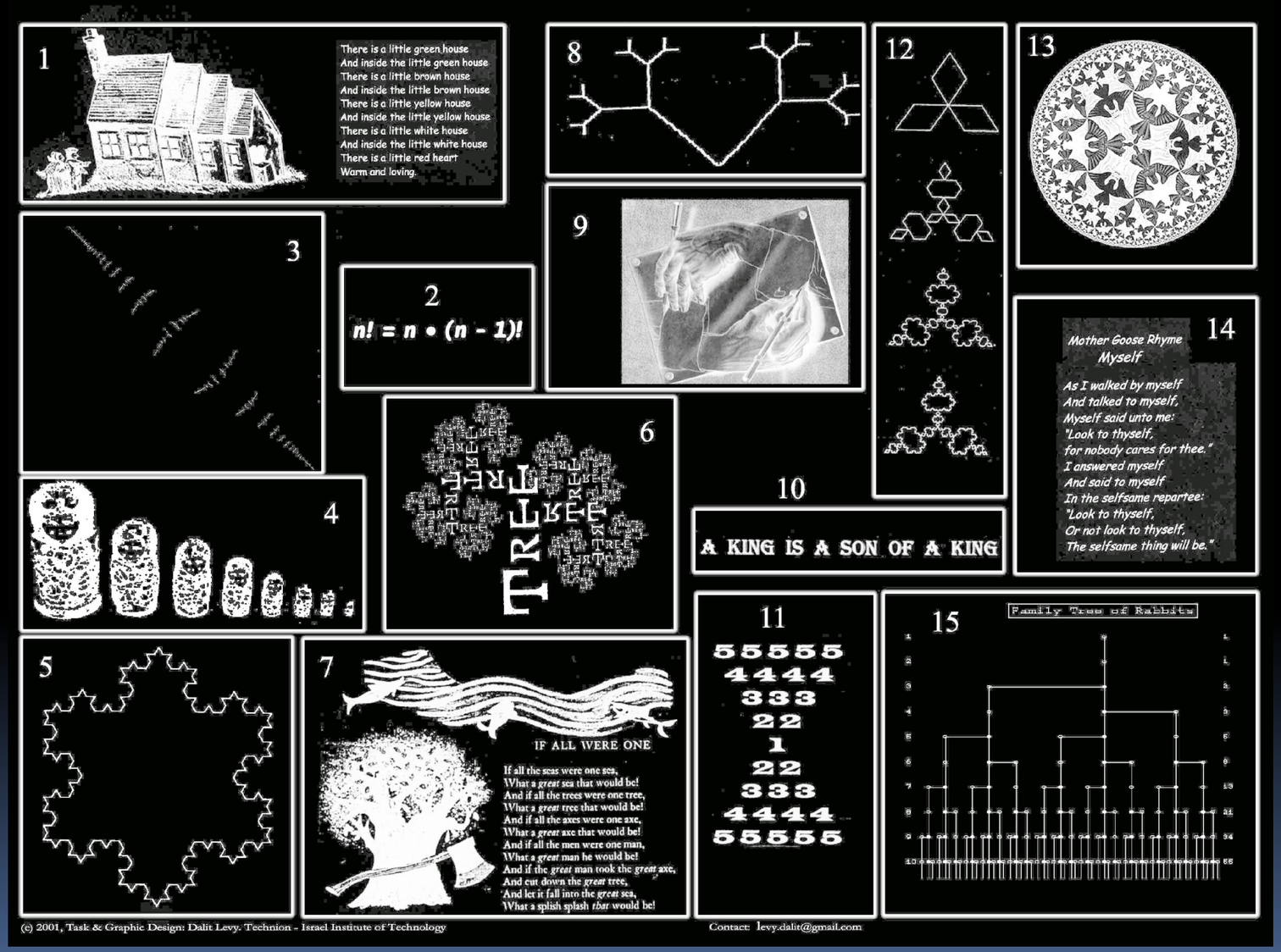
- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Recursion: Definition, You Know It, Trust It

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!

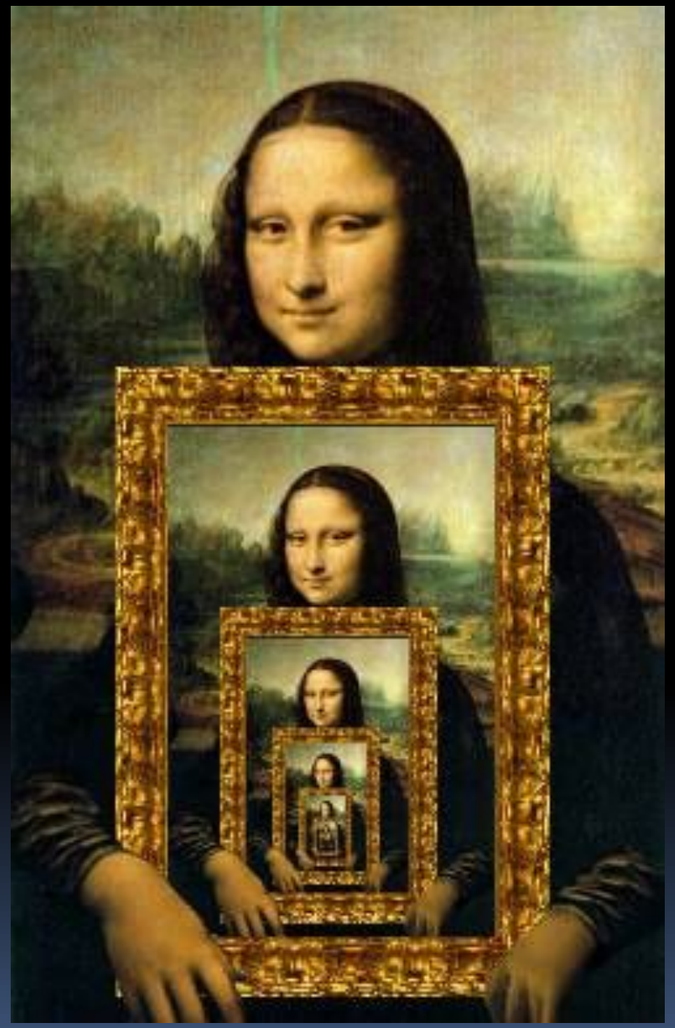


Trust the Recursion

- 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)
- For self-similar problems, **writing a recursive solution is ◆ than an iterative one**
  - more powerful than, easier
  - just as powerful as, easier
  - more powerful than, harder
  - just as powerful as, harder



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

When poll is active, respond at **pollev.com/ddg**

Text **DDG** to **22333** once to join

L10b Recursion is _____ Iteration. For self-similar problems, writing a recursive solution is _____ than writing an iterative one.

- MORE powerful than; EASIER
- MORE powerful than; HARDER
- LESS powerful than; EASIER
- LESS powerful than; HARDER
- JUST AS powerful as; EASIER
- JUST AS powerful as; HARDER

Summary

- Behind Abstraction, Recursion is the 2nd biggest idea about programming in this course
- Format (usually) is 2 cases:
  - Base Case
  - Recursive case
    - Divide, Invoke, Combine
- It's no more powerful than iteration, but often leads to more concise & better code
- It's most useful when the problem is self-similar

xkcd.com/244/