

LEC 07

CSE 123

Recursive Programming

Questions during Class?
Raise hand or send here

sli.do #cse123



BEFORE WE START

Talk to your neighbors:

What is your perfect temperature?

Respond on [sli.do](#)!

Instructor: Trien Vuong

TAs:	Cora	Nhan
	Jonah	Nichole
	Maitreyi	Shiven

Music: [CSE 123 26su Lecture Tunes](#)

Announcements

- Yay Quiz 1 is done!
 - Again, grades before Quiz 2 but we have makeups to take of
- Programming Assignment 1 due tonight (7/22) at 11:59 pm
- Creative Project 1 released tomorrow due next Wednesday (7/29)
 - Focused on recursion!
 - Experimenting with giving y'all the choice between two assignments
- Resubmission Cycle 2 cycle out due Friday (7/24) at 11:59pm
 - P1 is not eligible. Only P0 and C0

Recursive Methods [Review]

- 2 components of every recursive method:
- Recursive case
 - Decompose problem into subproblem
 - Make the actual recursive call
 - Combine results meaningfully
- Base case
 - Simplest version of the problem
 - No subproblems to break into
 - Return known answer



If decomposing moves you closer to the base, no infinite recursion!

Why Recursion?

- Generally, anything you can write iteratively you can write recursively
 - So why write anything recursively?

Recursion is particularly useful when dealing with something that's recursively defined

- Math examples:
 - Factorial: $n! = n * (n - 1)!$
 - Exponent: $x^n = x * x^{n-1}$
 - Fibonacci: $fib(n) = fib(n - 1) + fib(n - 2)$
- Non-math examples?
 - ListNodes (int data, ListNode next)
 - Other ideas?

Public / Private Pairs

- Used when we need additional information between recursive calls
- Private helper method hides additional info
 - Clients shouldn't have to worry about it
- All public method does is kick-start the private one
 - What's the correct starting value(s) for additional param(s)?

Question to ask: "Do I need to keep track of any additional information?"

Files

- We'll say that computer files fall into one of the following categories:



Standard file (.txt, .csv, .java)

```
f.isDirectory() -> false
```



Directory w/ subfiles

```
f.isDirectory() -> true  
File[] subFiles = f.listFiles()
```

This is a recursive definition! A File is either normal, or a directory with a File[] of subFiles