

# CS3L: Introduction to Symbolic Programming

Lecture 20:  
Tree Recursion

Summer 2008

Colleen Lewis  
colleenL@berkeley.edu



## Announcements

- Midterm two Tuesday July 29<sup>th</sup>
  - This one will probably be harder than the first
- Colleen will be out of town Thurs-Monday
  - I'm leaving before 10am
- Miniproject 3
  - Due Friday July 25<sup>th</sup> at 11:59 pm



## Today

- Tree recursion
- Work on the mini-project
- Tomorrow you'll do the tree recursion lab



## Tree Recursion

more than one recursive call can  
be made within a recursive case.



## What will happen?

- What will `countem` return for  $n=1, 2, \dots$ ?

```
(define (countem n)
  (if (= n 0)
      '()
      (se (countem (- n 1))
          n
          (countem (- n 1)))))
```



## What will happen?

- What will `mystery` return for  $n=1, 2, \dots$ ?

```
(define (mystery n)
  (if (= n 0)
      '()
      (se (mystery (- n 1))
          n
          (mystery (- n 1)))))

(mystery 0) → '()
(mystery 1) → (se '() 1 '()) → '(1)
(mystery 2) → (se '(1) 2 '(1)) → '(1 2 1)
(mystery 3) → (se '(1 2 1) 3 '(1 2 1)) →
              '(1 2 1 3 1 2 1)
```



## Pascal's triangle

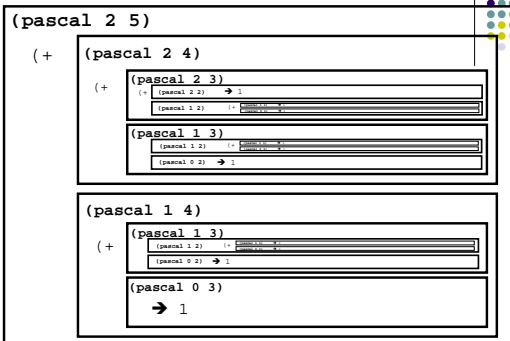
		columns (C)						
		0	1	2	3	4	5	...
r o w s  (R)	0	1						...
	1	1	1					...
	2	1	2	1				...
	3	1	3	3	1			...
	4	1	4	6	4	1		...
	5	1	5	10	10	5	1	...
...	...	...	...	...	...	...	...	

- Pascal's Triangle
- How many ways can you choose C things from R choices?
  - Coefficients of the  $(x+y)^R$ : look in row R
  - etc.

## Pascal's Triangle

```
(define (pascal C R)
  (cond
    ((= C 0) 1) ;base case
    ((= C R) 1) ;base case
    (else ;tree recurse
      (+ (pascal C (- R 1))
         (pascal (- C 1) (- R 1))
      )))
```

```
> (pascal 2 5)
```



## Pascal's Triangle

```
(define (pascal C R)
  (cond
    ((= C 0) 1) ;base case
    ((= C R) 1) ;base case
    (else ;tree recurse
      (+ (pascal C (- R 1))
         (pascal (- C 1) (- R 1))
      )))
```