

# Discussion 03

## Recursion

Aditya Balasubramanian

`aditbala [at] berkeley [dot] edu`

# Announcements

- Midterm Discussion
  - [CS61A Final Studying Guide](#)
  - [Advising OH](#)
- Project 1: Hog is due Friday 2/10.
  - Project party Wednesday 5pm-7pm in 101B Warren Hall.
  - Earn an early submission bonus point for submitting by Thursday 2/9.

# Recursion



# An Exciting Activity

- The Problem
  - In line to get boba
  - Want to find out how many people are in front of you



# Recursion

- What is a recursive function?
  - A function that calls itself
  - Returns a function call of itself, not the object (different than HOF)
- Recursive Leap of Faith
  - The idea that the recursive function will work no matter what/how many test cases are passed in

# Solution in Formal Terms of Recursion

- Base Case
  - smallest problem with guaranteed answer, or smallest input
  - **One dish left**
- Recursive Call
  - A method of reducing the current problem into a smaller problem
  - **Making a clone to wash dishes**

# Recursion vs Iteration

- Recursion
  - Make problem smaller
  - Variables get reset
  - Many frames will open
    - Lot of memory taken up
  - Better for some problems
    - Recursive Data Structures (Trees, Linked Lists)
- Iteration
  - Loops happen in one frame
  - Easy to visualize
  - No additional function calls

# Q2: Recursion Environment Diagram

Draw an environment diagram for the following code:

```
def rec(x, y):  
    if y > 0:  
        return x * rec(x, y - 1)  
    return 1  
  
rec(3, 2)
```



# Q1: Recursive Multiplication

Write a function that takes two numbers m and n (only positive) and returns their product. Use recursion, not `mul` or `*`

```
def multiply(m, n):  
    """ Takes two positive integers and returns  
    their product using recursion.  
>>> multiply(5, 3)  
15  
"""  
  
    """ * * * YOUR CODE HERE * * *
```

# Q3: Find the Bug

Find the bug with this recursive function.

```
def skip_mul(n):  
    """Return the product of n * (n - 2) * (n - 4) * ...  
    >>> skip_mul(5) # 5 * 3 * 1  
    15  
    >>> skip_mul(8) # 8 * 6 * 4 * 2  
    384  
    """  
    if n == 2:  
        return 2  
    else:  
        return n * skip_mul(n - 2)
```

# Choose your own adventure !!!

Q3 , Q4 , Q5

# Thank you!

**Anon Feedback -> <https://tinyurl.com/adit-anon>**