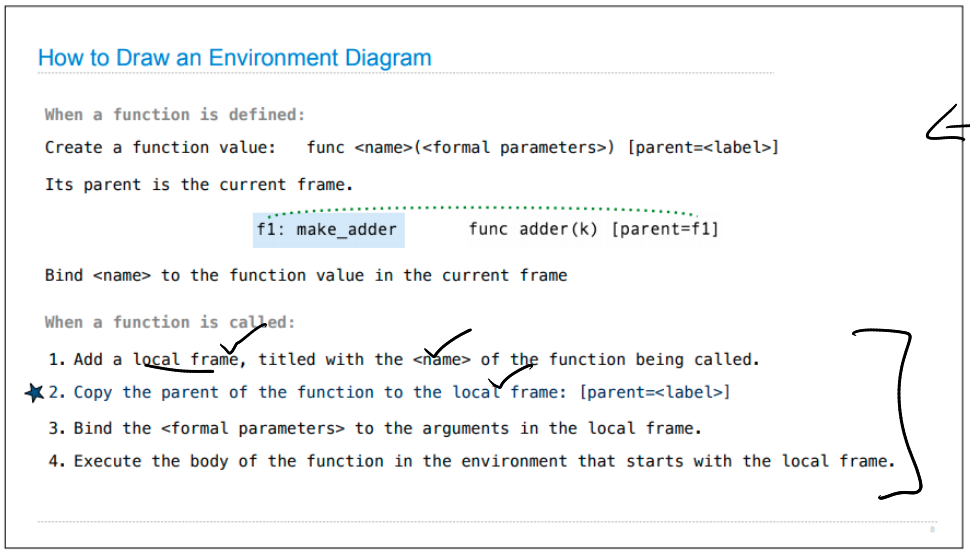
Welcome to CS61A Small-Group Tutoring

Tutor: Tarang Srivastava (he/him)



ENVIRONMENTS

s CS 61A

1 Environment Diagrams

1. When do we make a new frame in an environment diagram?

[everytime we call a function]

2. Draw the environment diagram that results from running the following code.

```
def swap(x, y):
    x, y = y, x
    return print("Swapped!", x, y)
x, y = 60, 1
a = swap(x, y)
swap(a, y)
```

print(...) → None

a = swap(x, y) → None

swap(a, y) → 1

Global Frame

swap → func swap(x, y) [parent: global]

x → 60

y → 1

a → None

F1: swap [parent: global]

x → 60

y → 1

rv → None

F2: swap [parent: global]

x → None

y → 1

rv → None ✓

Output

Swapped! 1 60

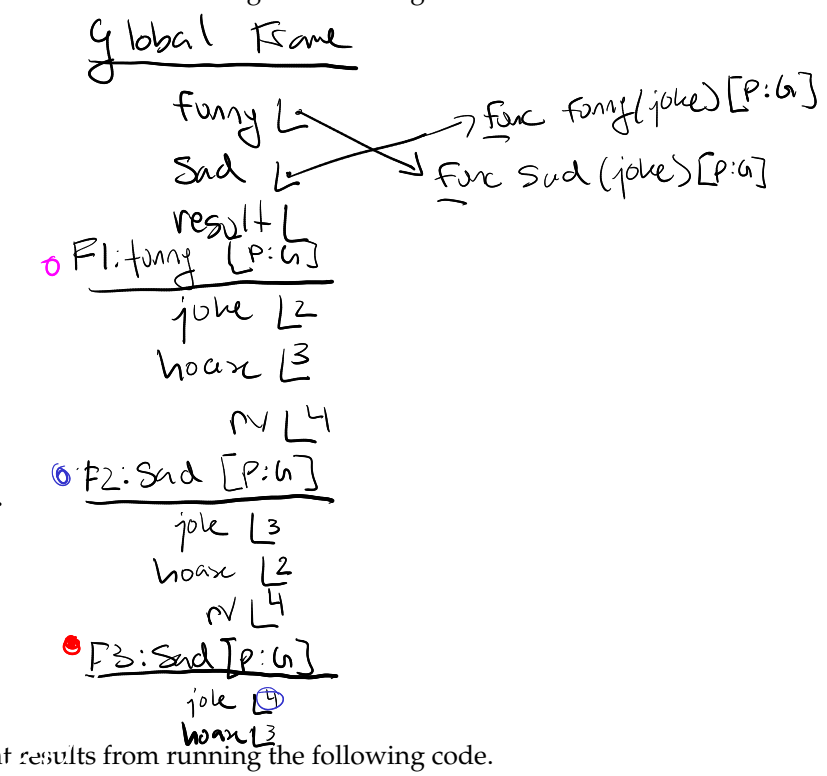
Swapped! 1 None

3. Draw the environment diagram that results from running the following code.

```
def funny(joke):
    hoax = joke + 1
    return funny(hoax)

def sad(joke):
    hoax = joke - 1
    return hoax + hoax

funny, sad = sad, funny
result = funny(sad(2))
```



4. Draw the environment diagram that results from running the following code.

```
a = 1
c = 2
def b(b):
    def d():
        return b + c
    return d()
c = b(a)
a = b(c)
```

2 Control

1. Write a function that returns true if a number is divisible by 4 and false otherwise.

2. Write a function, `is_leap_year`, that returns true if a number is a leap year and false otherwise. Recall that a *leap year* is divisible by 4 unless the year is not divisible by 400.

3. Implement `fizzbuzz(n)`, which prints numbers from 1 to `n` (inclusive). However, for numbers divisible by 3, print "fizz". For numbers divisible by 5, print "buzz". For numbers divisible by both 3 and 5, print "fizzbuzz".

```
def fizzbuzz(n):
    """
    >>> result = fizzbuzz(16)
    1
    2
    fizz
    4
    buzz
    fizz
    7
    8
    fizz
    buzz
    11
    fizz
    13
    14
    fizzbuzz
    16
    >>> result is None
    True
    """
```

3 Challenge Questions

1. Fill out the function `digit_div` which returns an integer that contains in any order all the digits of `k` that divide `n` evenly. If no such digit of `k` exists, the function should return 0. Assume that both `n` and `k` are positive integers.

```
def digit_div (n, k):
    >>> digit_div(4, 1234567890)
    421
    >>> digit_div(4, 2323)
    22
    >>> digit_div(7, 2323)
    0
    """
    Val = 0
    while n > 0:
        curr_digit = k % 10
        if k % curr_digit == 0:
            Val = (Val * 10) + curr_digit
        curr
    return
```