Function Examples

Announcements

Hog Contest Rules

- Up to two people submit one entry; Max of one entry per person
- Your score is the number of entries against which you win more than 50.00001% of the time
- Strategies are time-limited
- All strategies must be deterministic, pure functions of the players' scores
- Winning entries will receive a paltry amount of extra credit
- The real prize: honor and glory
- See website for detailed rules

Fall 2011 Winners

Kaylee Mann
Yan Duan & Ziming Li
Brian Prike & Zhenghao Qian
Parker Schuh & Robert Chatham

Fall 2012 Winners

Chenyang Yuan
Joseph Hui

Fall 2013 Winners

Paul Bramsen
Sam Kumar & Kangsik Lee
Kevin Chen

Fall 2014 Winners

Alan Tong & Elaine Zhao
Zhenyang Zhang
Adam Robert Villaflor & Joany Gao
Zhen Qin & Dian Chen
Zizheng Tai & Yihe Li

Hog Contest Winners

Spring 2015 Winners

Sinho Chewi & Alexander Nguyen Tran
Zhaoxi Li
Stella Tao and Yao Ge

Fall 2015 Winners

Micah Carroll & Vasilis Oikonomou
Matthew Wu
Anthony Yeung and Alexander Dai

Spring 2016 Winners
Michael McDonald and Tianrui Chen
Andrei Kassiantchouk
Benjamin Krieges

Fall 2016 Winners
Will Gan & Robert Quitt
Eric Sheng & Sachin Kesiraju
Mingwei Samuel
Simon Zhuang & Vaikunth Balaji
Fanyu Meng & Zekai Fan

Spring 2017 Winners
Cindy Jin and Sunjoon Lee
Anny Patino and Christian Vasquez
Asana Choudhury and Jenna Wen
Michelle Lee and Nicholas Chew

Fall 2017 Winners
Alex Yu and Tanmay Khattar
James Li
Justin Yokota

Spring 2018 Winners
Eric James Michaud
Ziyu Dong
Xuhui Zhou

Fall 2018 Winners
Rahul Arya
Jonathan Bodine
Sumer Kohli and Neelesh Ramachandran

Hog Contest Winners

Your name could be here FOREVER!

Fall 2019 Winners
Jet Situ and Lucas Schaberg
Anthony Han and Hongyi Huang
Arthur Pan and Qingyuan Liu

Spring 2020 Winners
Andy Dong
Theodor Sion and Anish Kar
Shaun Diem-Lane

Fall 2020 Winners
Timothy Guo
Shomini Sen
Samuel Berkun
Mitchell Zhen
Lucas Clark
Dominic de Bettencourt
Allen Gu
Alec Li
Aaron Janse

Fall 2021 Winners

Describing Functions

Boolean Favorites

```python
def likes(n):
    """Returns whether George Boole likes the non-negative integer n."""
    ...

def mystery1(n):            likes = is_prime
    k = 1                   n = 8
    while k < n:
        if likes(n):
            print(k)
        k = k + 2
```

One approach:
  1. Read the code
  2. Read the description options
  3. Consider an example

all odd numbers   but only if George likes n

mystery1 prints _____ less than n _____ .

~~**mystery1** prints all odd numbers less than n that George likes.~~

Boolean Favorites

```python
def likes(n):
    """Returns whether George Boole likes the non-negative integer n."""
    ...

def mystery2(n):
    i, j, k = 0, None, None
    while i < n:
        if likes(i):
            if j != None and (k == None or i - j < k):
                k = i - j
            j = i
        i = i + 1
    return k
```

One approach:
1. Read the code
2. Read the description options
3. Consider an example

the smallest difference between
two positive integers below n
that George likes

There are no two
such integers

mystery 2 returns _____ or returns None if _____ .

Generating Environment Diagram

A Day at the Beach

```
def flip(flop):
    if flop>2 :            not true for flop == 1
        return None            true for flop == 3
    flip = lambda flip: 3
    return flip


def flop(flip):
    return flop


flip, flop = flop, flip


flip(____)(3)
```

flop(1)(2)

Frames

Global frame

⭐ flip •
flop •

f1: flip [parent=Global]
⭐ flop | 1
flip •
Return value •

f2: λ <line ?> [parent=f1]
⭐ flip | 2
⭐ Return value | 3

f3: flop [parent=Global]
⭐ flip | 3
Return value •

f4: flip [parent=Global]
⭐ flop | 3
Return value | None ⭐

Objects

func flip(flop) [parent=Global]

func flop(flip) [parent=Global]

func λ(flip) [parent=f1] ⭐

9
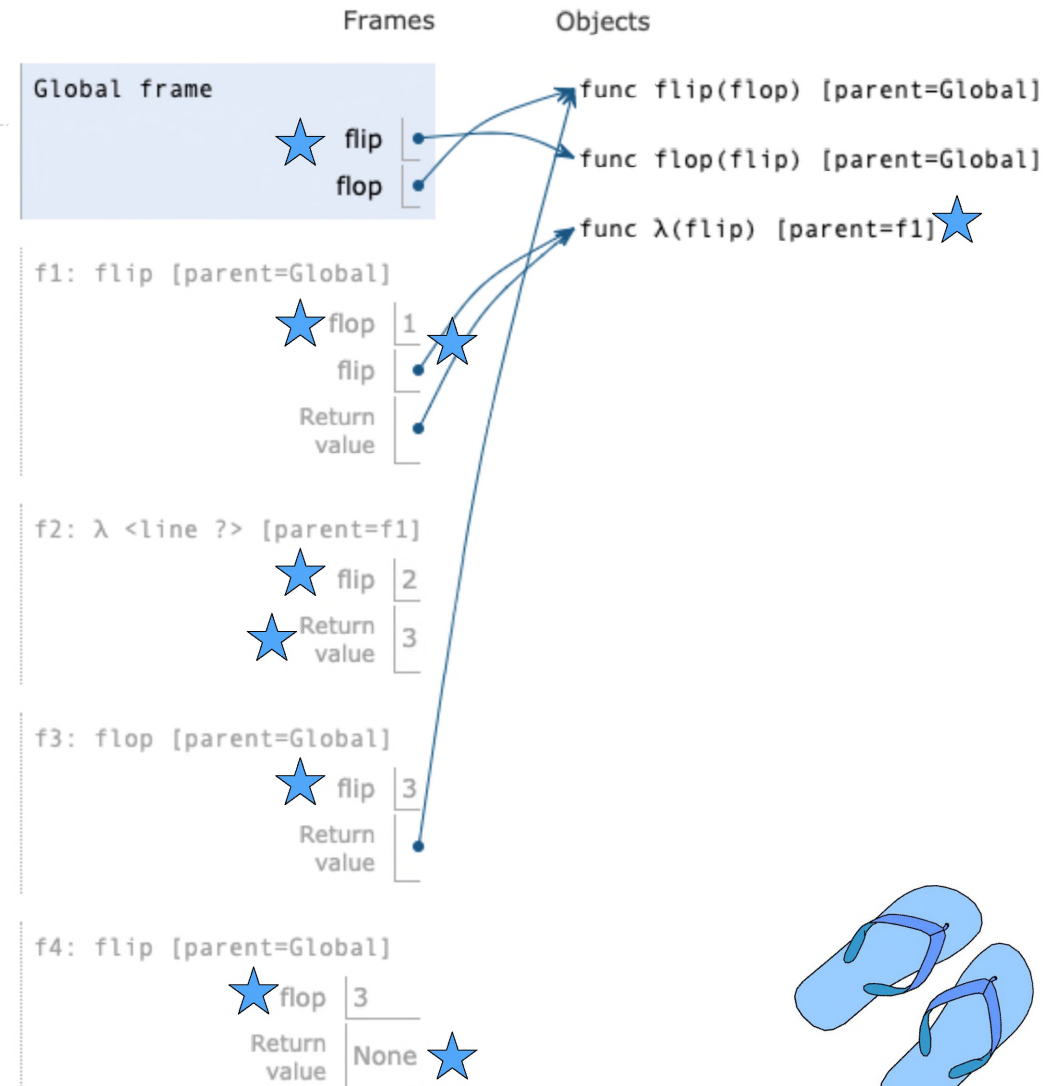
Review

What Would Python Display?

The print function returns None. It also displays its arguments
(separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any
argument and returns a
function that returns
that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def
statements can refer to
their enclosing scope

This expression	Evaluates to	Interactive Output
5	5	5
print(5)	None	5
print(print(5)) None	None	5 None
delay(delay)()(6)()	6	delayed delayed 6
print(delay(print)()(4))	None	delayed 4 None

What Would Python Print?

The print function returns None. It also displays its arguments
(separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that
always returns the
identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

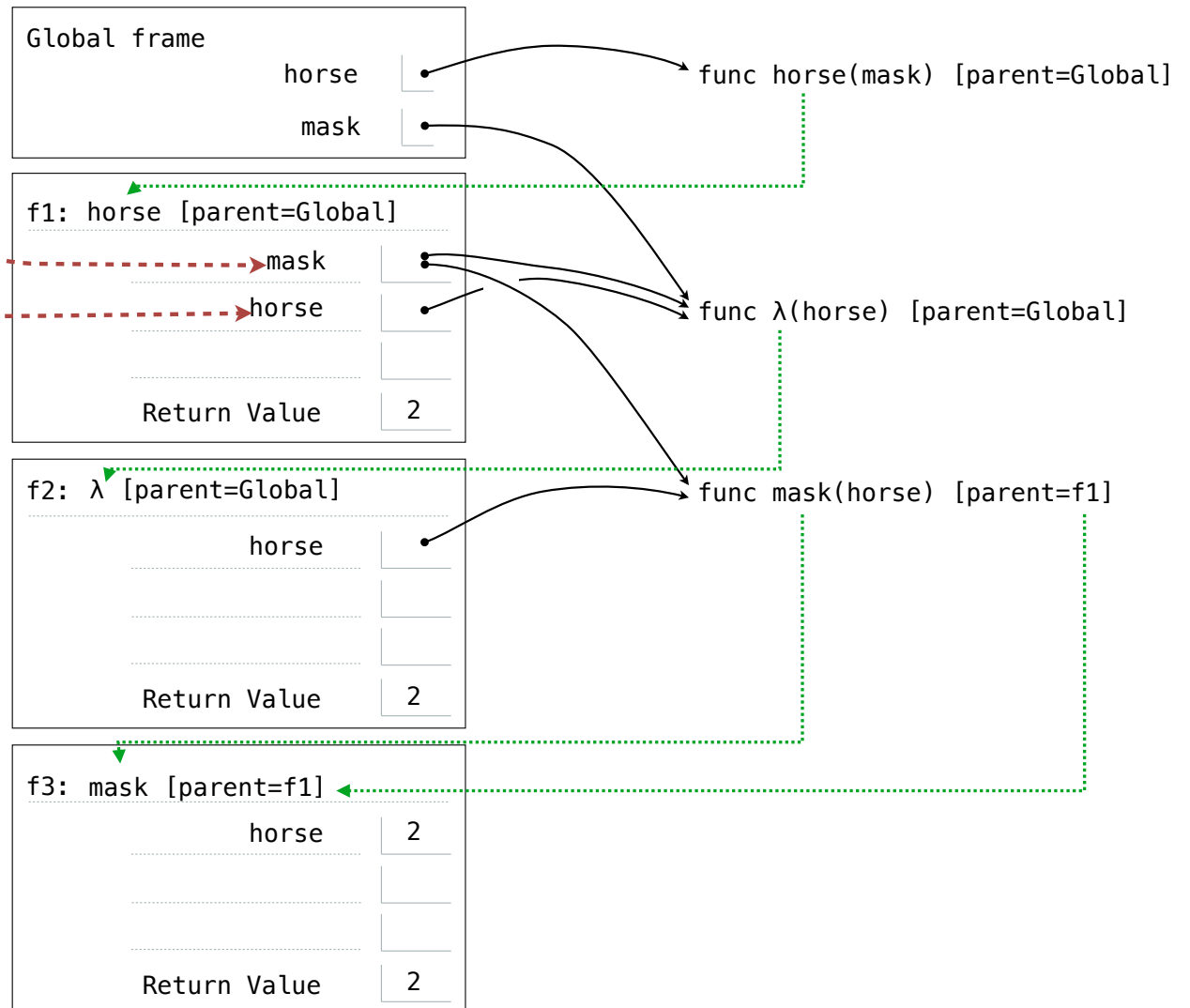| This expression	Evaluates to	Interactive Output
add(pirate(3)(square)(4), 1)	17	Matey 17
func square(x)		
16		
pirate(pirate(pirate))(5)(7)	Error	Matey Matey Error
Identity function		
5		

A name evaluates to the value bound to that name in the earliest frame of the current environment
in which that name is found.

```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```

Global frame

horse   func horse(mask) [parent=Global]

mask

f1: horse [parent=Global]

mask

horse   func λ(horse) [parent=Global]

Return Value   2

f2: λ [parent=Global]

horse   func mask(horse) [parent=f1]

Return Value   2

f3: mask [parent=f1]

horse   2

Return Value   2

Implementing Functions

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
```
[231] [4] IT, for some
IT less than 10.

```
    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while _____ n > 0 _____:

        n, last = n // 10, n % 10

        if _____ last != digit _____:

            kept = __~~10*~~kept + last*10**digits__

    [231]  digits = ___ digits + 1 ___

    return _____ kept _____
```

```
    1        1
  + 20     + 30
           + 200
  _____   _____
    21       231
```

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

11

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
```
231 are 3 IT, for some
non-negative IT less than **10**.

```
    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while _____n > 0_____:

        n, last = n // 10, n % 10

        if _____last != digit_____:

            kept = _____kept/10 +    last_____
```
21
```
        digits = _____digits + 1_____

    return _____round(kept * 10 ** (digits-1))_____
```

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example
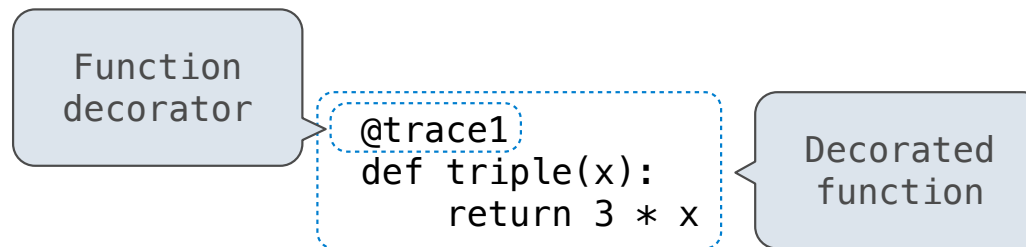
Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Decorators

Function Decorators

(Demo)

Function
decorator

```
@trace1
def triple(x):
    return 3 * x
```

Decorated
function

is identical to

Why not just
use this?

```
def triple(x):
    return 3 * x
triple = trace1(triple)
```