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

cs61a.org/proj/hog_contest

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

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

Your name could be here FOREVER!

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

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

Describing Functions

Boolean Favorites

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

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

likes = is_prime
n = 8

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

```
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

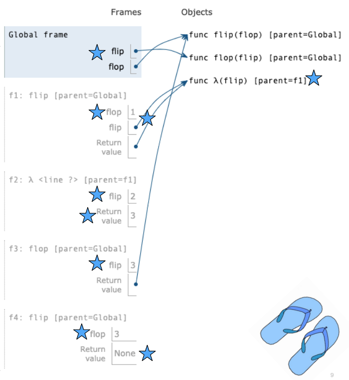
A Day at the Beach

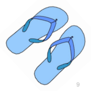
```
def flip(flop):
    if flop>2 :         <- not true for flop == 1
                           true for flop == 3
       return None
    flip = lambda flip: 3
    return flip

def flop(flip):
    return flop

flip, flop = flop, flip

flip(____)(3)
     flop(1)(2)
```





Implementing Functions

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       [illegible]are[illegible]IT, for some
       [illegible]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
            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

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       [illegible]are[illegible]IT, for some
       [illegible]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______
            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)
```