# BASICS, CONTROL STRUCTURES, AND HIGHER ORDER FUNCTIONS

### COMPUTER SCIENCE 61A

June 23 to July 3, 2015

# **1** Expressions and Functions

#### What would Python print?

```
1. Order of evaluation:
```

```
>>> def jurassic(park, world):
... print(world)
... return park - world
>>> def big(dino):
... print(dino)
... return 2 * dino
... print(dino + 1)
>>> closed = jurassic(jurassic(5, 4), big(7))
```

#### Solution:

4 7 14

>>> closed

```
Solution:
-13
```

```
Page 2
```

Solution: 42

>>> x

**Solution:** Nothing shows up. This is because x is assigned to None (the the return value of print)

```
>>> def foo(y):
... return y * y
>>> def bar(y):
... print(y * y)
>>> a = foo(4)
>>> a == 16
```

Solution: True

>>> b = bar(4)

Solution: 16

>>> b == 16

#### Solution:

False

Since bar does not have a return value, it implicitly returns None. Thus, b is assigned to None.

```
>>> def garply(y):
... print(y * y)
... return 3
>>> c = garply(4)
```

Solution: 16

>>> c

Solution: 3

CS 61A Summer 2015: Albert Wu and Robert Huang, with

Andrew Blum, Anish Balaji, Anthony de la Paz, Diana Advani, Delphine Ho, Fahad Kamran, Gunjan Baid, Joseph Simonian, Ken Katagiri, Meha Bakshi, Peter Xu, Raymond Chan, Tiffany Perumpail, and Zhen Qin

### **2** Control structures

1. Implement factorial(n), which takes a non-negative n and returns all the numbers from 1 to n multiplied together. For example, factorial(5) = 1 \* 2 \* 3 \* 4 \* 5 = 120.

*Note*: Your function should be able to compute factorial(0) to be 1, as defined in mathematics.

```
def factorial(n):
    """Returns the product of numbers from 1 to n.
    >>> factorial(0)
    1
    >>> factorial(1)
    1
    >>> factorial(5)  # 1 * 2 * 3 * 4 * 5
    120
    """
```

```
Solution:
```

```
i, total = 1, 1
while i <= n:
    total = total * i
    i += 1
return total</pre>
```

## **3** Higher order functions

1. Draw an environment diagram for the following code:

```
x = 5
def illum(nati):
    y = nati + x
    return nati - x
def files(x):
    return illum(x) - x
```

x = files(6)illum(4)

	Fra	ames Objec	ts	
	Global frame X illum files	-5	c illum(nati) [parent=Global] c files(x) [parent=Global]	
	f1: files [parent=Glob X Return value	al] 6 -5		
	f2: illum [parent=Glob nati y Return value	al] 6 11 1		
Solution:	f3: illum [parent=Glob nati Y Return value	al] 4 -1 9		

```
2. Draw an environment diagram for the following code:
y = 1
def cons(piracy):
    def confirmed(x):
        return piracy(x + y)
        y = 4
        return confirmed
```

cons(**lambda** a: a + y)(5)

