Announcements

• Midterm 1 is on Monday 9/23 from 7pm to 9pm
  • 2 review sessions on Saturday 9/21 2pm–4pm and 4pm–6pm in 1 Pimentel
  • HKN review session on Sunday 9/22 from 4pm to 7pm in 2050 Valley LSB
  • Extra weekend office hours announced on Piazza
  • Cannot attend? Fill out the conflict form by Friday 9/20 @ 11:59pm!
• No lab next week: Monday 9/23, Tuesday 9/24, or Wednesday 9/25
• Homework 3 due Tuesday 10/1 @ 11:59pm
• Optional Hog strategy contest ends Thursday 10/3 @ 11:59pm
Abstraction
Functional Abstractions

```
def square(x):
    return mul(x, x)

def sum_squares(x, y):
    return square(x) + square(y)
```

What does `sum_squares` need to know about `square`?

- Square takes one argument.  
  Yes

- Square has the **intrinsic** name square.  
  No

- Square computes the square of a number.  
  Yes

- Square computes the square by calling `mul`.  
  No

```
def square(x):
    return pow(x, 2)

def square(x):
    return mul(x, x-1) + x
```

If the name “square” were bound to a built-in function, `sum_squares` would still work identically.
### Choosing Names

Names typically *don’t* matter for correctness

**but**

...they matter a lot for composition

<table>
<thead>
<tr>
<th>From:</th>
<th>To:</th>
</tr>
</thead>
<tbody>
<tr>
<td>true_false</td>
<td>rolled_a_one</td>
</tr>
<tr>
<td>d</td>
<td>dice</td>
</tr>
<tr>
<td>play_helper</td>
<td>take_turn</td>
</tr>
<tr>
<td>my_int</td>
<td>num_rolls</td>
</tr>
<tr>
<td>l, I, 0</td>
<td>k, i, m</td>
</tr>
</tbody>
</table>

Names should convey the *meaning* or *purpose* of the values to which they are bound.

The type of value bound to the name is best documented in a function's docstring.

Function names typically convey their effect (print), their behavior (triple), or the value returned (abs).
Which Values Deserve a Name

Repeated compound expressions:

```python
if sqrt(square(a) + square(b)) > 1:
    x = x + sqrt(square(a) + square(b))
```

```python
hypotenuse = sqrt(square(a) + square(b))
if hypotenuse > 1:
    x = x + hypotenuse
```

Meaningful parts of complex expressions:

```python
x = (-b + sqrt(square(b) - 4 * a * c)) / (2 * a)
```

```python
discriminant = sqrt(square(b) - 4 * a * c)
x = (-b + discriminant) / (2 * a)
```

More Naming Tips

- Names can be long if they help document your code:

```python
average_age = average(age, students)
```

is preferable to

```python
# Compute average age of students
aa = avg(a, st)
```

- Names can be short if they represent generic quantities: counts, arbitrary functions, arguments to mathematical operations, etc.

```python
n, k, i - Usually integers
x, y, z - Usually real numbers
f, g, h - Usually functions
```
Testing
Test-Driven Development

Write the test of a function before you write the function.

A test will clarify the domain, range, & behavior of a function.

Tests can help identify tricky edge cases.

Develop incrementally and test each piece before moving on.

You can't depend upon code that hasn't been tested.

Run your old tests again after you make new changes.

Run your code interactively.

Don't be afraid to experiment with a function after you write it.

Interactive sessions can become doctests. Just copy and paste. (Demo)
Decorators
Function Decorators

(demo)

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

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

is identical to

Why not just use this?
Review
What Would Python Print?

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

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

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

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

# Names in nested def statements can refer to their enclosing scope
```

<table>
<thead>
<tr>
<th>This expression</th>
<th>Evaluates to</th>
<th>And prints</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>print(5)</td>
<td>None</td>
<td>5</td>
</tr>
<tr>
<td>print(add(3, 4), print(5))</td>
<td>None</td>
<td>5 7 None</td>
</tr>
<tr>
<td>delay(delay)()(6)()</td>
<td>6</td>
<td>delayed delayed</td>
</tr>
<tr>
<td>print(delay(print)()(4))</td>
<td>None</td>
<td>delayed delayed 4 None</td>
</tr>
</tbody>
</table>
The print function returns None. It also displays its arguments (separated by spaces) when it is called.

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

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

A function that always returns the identity function

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

Example: [http://goo.gl/NdrVqr](http://goo.gl/NdrVqr)
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)