Week 4

*Talk is cheap. Show me the code.*

- Questions
- Example - Function Design
- Python Modules/Turtle Graphics
- Midterm Topics/Tricky Concepts
- Midterm

---

### Functions Concepts Checklist

- Python built-in functions
- combining functions
- user-defined functions
- function keywords (def, return)
- defining v. calling functions
- parameters v. arguments
- local variables
- memory model/namespace
- function calls are expressions/return a value
  (sometimes the None value)
- function design recipe

---

### Programming example:

If \( n \) is the number of seconds between lightning and thunder, the storm is \( n/5 \) miles away. Write a program that requests the number of seconds between lightning and thunder and reports [prints] the distance from the storm rounded to two decimal places.

---

### Python modules

collection of related variables and functions grouped together in a single file

**Python Standard Library**

---

```
>>> import random

>>> type(random)
<class 'module'>

>>> random
module 'random' from '/Library/Frameworks/Python.framework/Versions/3.3/lib/python3.3/random.py'
```
>>> help(random)

>>> random.

>>> help(random.randint)

Python turtle module

>>> turtle.setpos(50, 50)
>>> turtle.setheading(180)
>>> turtle.color('green', 'yellow')
>>> turtle.shape('turtle')
>>> turtle.speed(0)
>>> turtle.forward(100)
>>> turtle.right(90)

Python modules

>>> import turtle
>>> from turtle import pos
>>> from turtle import heading
>>> from turtle import *

Summary of Topics, 1.

- Programming = CT + Coding
- High level languages: Python, etc.
- Python primitive elements
  Objects (id/type/value)
- Primitive elements can be combined
- Expressions evaluate to a value
- Assignment statements (are not expressions) associate name/object
- Variables are expressions. Function calls are expressions.
Summary of Topics, 2.

- Python built in data types – values and operations:
  - numbers – integers and floating point
  - strings
- Python built in functions and operators and methods:
  - +, *, /, //, **
  - round, pow, abs, min, max, len, float, str, print, input, type, help
- User-defined functions:
  - defining and calling
  - parameters and arguments
  - local variables
  - return a value
  - function design recipe
  - memory model

Summary of Topics, 3.

- Python input and output
- Python modules – e.g., random, turtle

Tricky Concepts Checklist

- assignment statement v. expression stmt
- defining v. calling a function
- parameters v. arguments
- return v. print output
- functions always return a value (incl. None)
- variables v. strings
- parameters v. local variables v. input

Defining v. Calling a Function

```python
>>> def eom(days):
    count_days = days - 1
    return .01 * 2 ** count_days
>>> eom
>>> eom(3) + eom(2)
>>> result = eom(3)
```

Parameters v. Arguments

```python
>>> def best_route(a, b, c, d, e):
    road1 = maxtrans(a, b, c)
    road2 = min(d, e)
    route_max = max(road1, road2)
    return route_max
>>> best_route(29, 30, 31, 50, 55)
```
def skittles():
    b = 7
    ttl = 10
    g = ttl - b
    return g

>>> x = skittles()
>>> x
7

>>> x = skittles()
>>> x
7

variables v. strings

>>> 'hello'
'hello'

>>> hello
hello

>>> greeting = 'hello'

>>> greeting
'hello'

>>> hello = 'hello'

>>> hello
'hello'

parameters v. local vars v. input()

def race(time):
    time = 8.7
    time = input('what is your time?')
    return time