CSC 101

Today

- import
- Tracing functions
- Writing functions

Have you read any good books lately?

import

data.py

```
class X
...
```

tests.py

```
import data

data.X
```

module

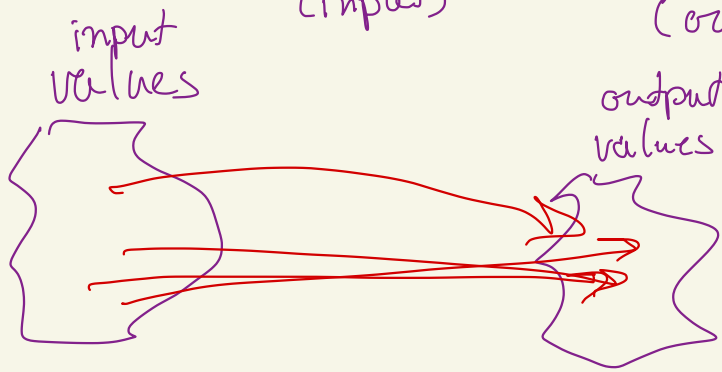
definition w/ module

```
from data import X

X
```

```
import math

math.sqrt(16)
```

Functions

- mapping from domain (input) to range (output)



- define calculations
- parameterize a calculation

$$dist(x1, y1, x2, y2) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

$f(x) = x$

$f(2) = 2$

$f(7) = 7$

Programming
- define calculations
- parameterize calculations

a "good" function
does "one thing"
- easier to reason about
- easier to write
- easier to test
- debug
reuse
replace

$f(x) = x$

type? float

input

output type

```
def f(x : float) -> float :
    return x
```

compute x and "return" value as result

- returned to caller

```
def f(x : float) -> float :
    return x

y = f(2)
```

function call

y = f(2)

call f w/ 2 as x

```
def f(x) :
    return x
```

with this knowledge

y = 2

Known values for f

x | 2

Known values (state)

def f(x) ---

```
def f(x : float) -> float :
    return x

y = f(2)
```

→ f(2)

return x

return 2

←

```
y = 2
```

Known values in f

x | 2 |

Known values

def f(x --

y | 2 |

$f(x) = x * 2$

```
def double(x : float) ->float :
    return x * 2
a = 4
result = double(a)
```

```
def double(x : float) ->float :
    return x * 2

a = 4
result = double(a)
```

double(double(4))

result = double(4)

→ double(4) (x gets 4)

return x * 2

return 4 * 2

return 8

← result = 8

Known values w/in double

x | 4

Known Values

def double ...

a | 4

result | 8

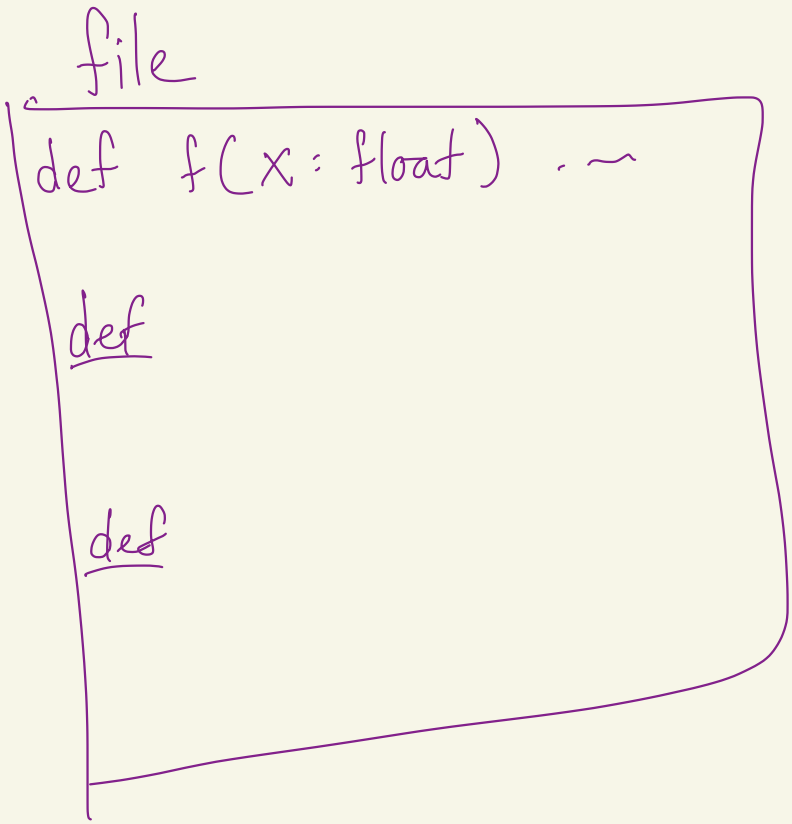
```
# Compute distance between two points
# input: two point objects
# result: float

def distance(pt1: data.Point,
             pt2: data.Point) -> float:
    return math.sqrt((pt1.x - pt2.x)**2 + (pt1.y - pt2.y)**2)
```

$$= \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

```
def distance(pt1: data.Point,
             pt2: data.Point) -> float:

    dx = pt1.x - pt2.x
    dy = pt1.y - pt2.y

    return math.sqrt(dx**2 + dy**2)
```

```
point1 = data.Point(4.2, 7.9)
point2 = data.Point(9.4, -1.7)
result = distance(point1, point2)
result = distance(point2, point1)
```

pt1 pt2 pt1 pt2


file
def f(x: float) . ~
def
def