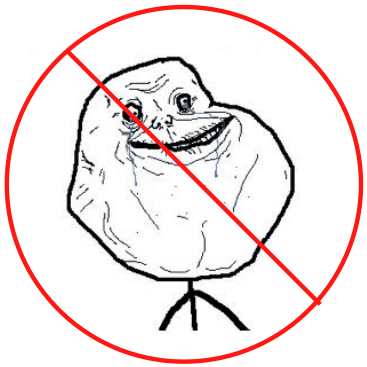
Iteration

Announcements

Office Hours: You Should Go!

You are not alone!

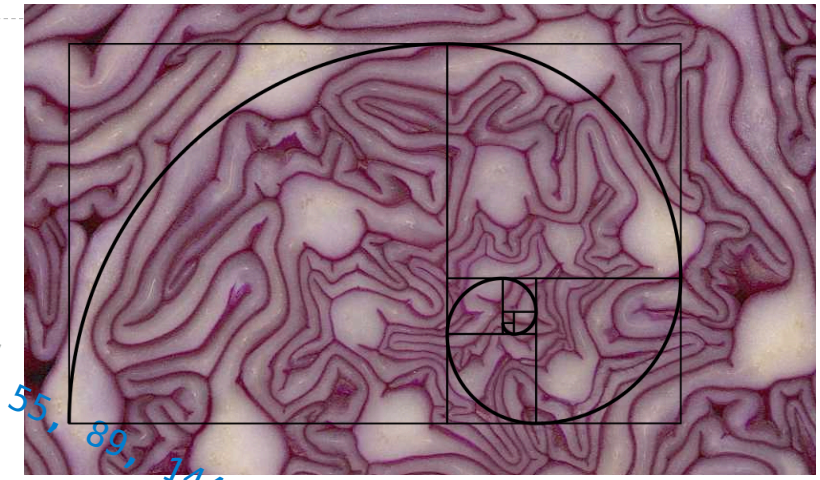


http://cs61a.org/office-hours.html

Iteration Example: Fibonacci Numbers

The Fibonacci Sequence

0 1 2 3 4 5 6 7 8 ...
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, ...
1 0 1 1 2 3 5 8 ...



```
def fib(n):
    """Compute the nth Fibonacci number.

    >>> fib(0)
    0
    >>> fib(8)
    21
    """
    k, kth, difference = 0, 0, 1
    while k < n:
        kth, difference = kth + difference, kth
        k = k + 1
    return kth
```

```
old_diff = difference
difference = kth
kth = kth + old_diff
```

```
kth = kth + difference
difference = kth
```

to write the statement in either order is incorrect,
you have to introduce a third variable, here old_diff

Return

Return Statements

A return statement completes the evaluation of a call expression and provides its value

f(x) for user-defined function f: switch to a new environment; execute f's body

return statement within f: switch back to the previous environment; f(x) now has a value

Only one return statement is ever executed while executing the body of a function

```
def end(n, d):
    """Print the final digits of N in reverse order until D is found.

    >>> end(34567, 5)
    7
    6
    5
    """
    while n > 0:
        last, n = n % 10, n // 10
        print(last)
        if d == last:
            return None
```

(Demo)

return None as a process of ending the statement

Define An Inverse Funtion

Stare at what you wrote and think
if you can simplify the logic

```
def search(f):
    x = 0
    while not f(x):
        x += 1
    return x
```

A simplified version

```
Terminal  Shell  Edit  View  Window  Help
lec — Python -i ex.py — 67×36
~/lec$ python3 -i ex.py
>>> sqrt
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'sqrt' is not defined
>>> sqrt = inverse(square)
>>> square(16)
256
>>> sqrt(256)
16
>>> sqrt(16)
4
>>> sqrt(4)
2
>>> sqrt(2)
^CTraceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "ex.py", line 19, in <lambda>
    return lambda y: search(lambda x: f(x) == y)
  File "ex.py", line 6, in search
    x += 1
KeyboardInterrupt
>>>
```

```
ex.py (~/Documents/workspace/61a_lectures/sp18) -
def search(f):
    x = 0
    while True:
        if f(x):
            return x
        x += 1

def is_three(x):
    return x == 3

def square(x):
    return x * x

def positive(x):
    return max(0, square(x) - 100)

def inverse(f):
    """Return g(y) such that g(f(x)) -> x."""
    return lambda y: search(lambda x: f(x) == y)
~
~
~
~
~
```

this way of defining sqrt only applies to integers,
see Newton's method on how to define sqrt

Self-Reference

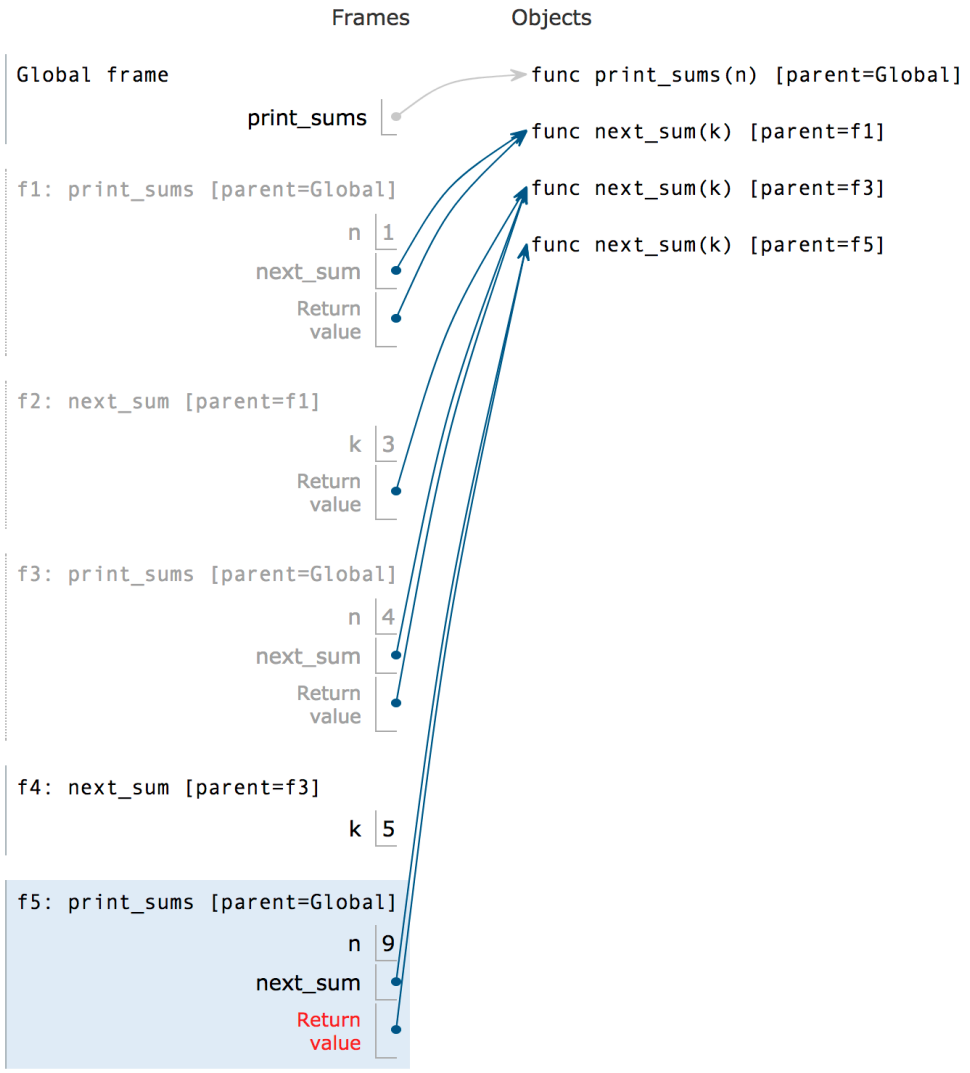
(Demo)

Returning a Function Using Its Own Name

```
1  def print_sums(n):
2      print(n)
3      def next_sum(k):
4          return print_sums(n+k)
5      return next_sum
6
7  print_sums(1)(3)(5)
```

next_sum does the summation,
print_sums prints the result

Note: only print_sums(1) calls print_sum,
print_sums(1) (3) & print_sums(1) (3) (5)
calls next_sum



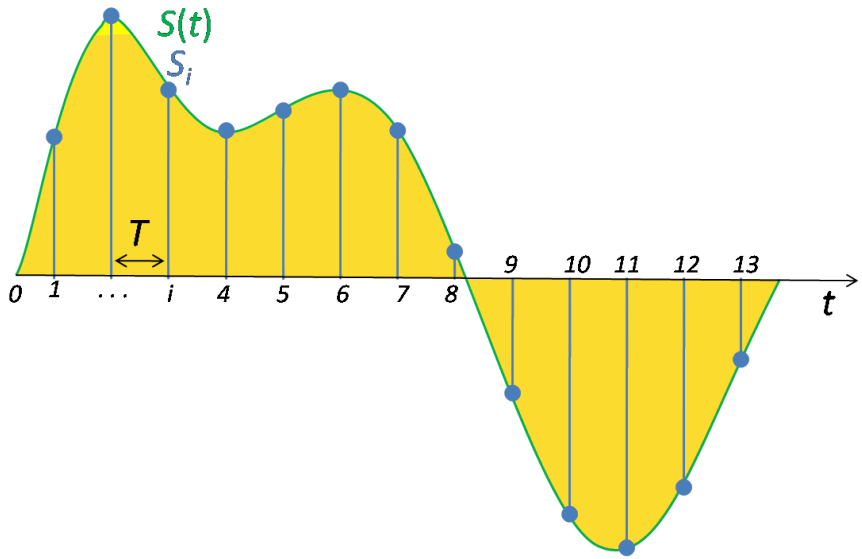
http://pythontutor.com/composingprograms.html#code=def%20print_all%28k%29%3A%0A%20%20%20%20print%28k%29%0A%20%20%20%20return%20print_all%0A%20%20%20%20%0Aprint_all%281%29%283%29%285%29&cumulative=true&curInstr=0&mode=display&origin=composingprograms.js&py=3&rawInputLstJSON=%5B%5D

http://pythontutor.com/composingprograms.html#code=def%20print_sums%28n%29%3A%0A%20%20%20%20print%28n%29%0A%20%20%20%20def%20next_sum%28k%29%3A%0A%20%20%20%20%20%20%20%20return%20print_sums%28n%2Bk%29%0A%20%20%20%20return%20next_sum%0A%0Aprint_sums%281%29%283%29%285%29&cumulative=true&curInstr=0&mode=display&origin=composingprograms.js&py=3&rawInputLstJSON=%5B%5D

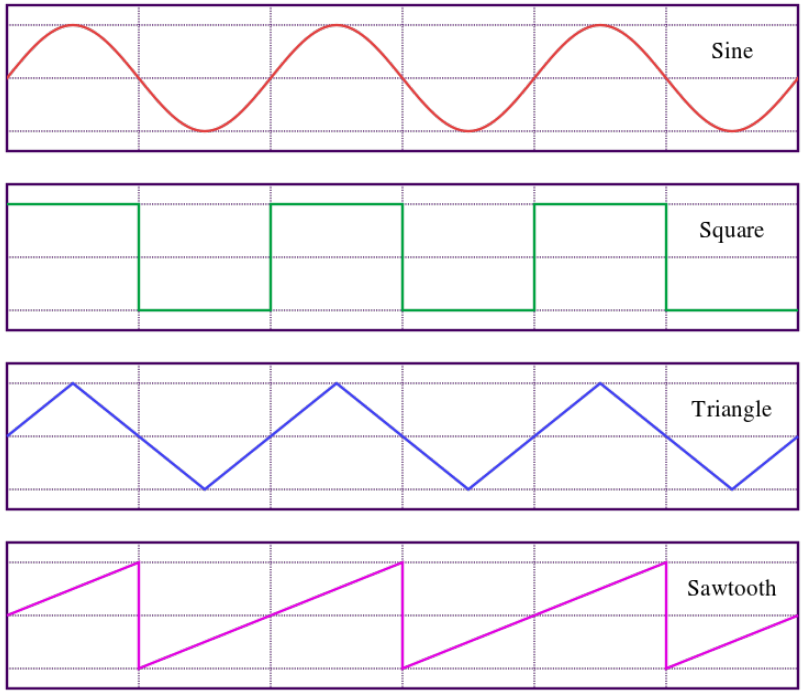
Function Example: Sounds

WAV Files

The Waveform Audio File Format encodes a sampled sound wave



A triangle wave is the simple wave form with the most pleasing sound



(Demo) notes in .py

https://en.wikipedia.org/wiki/Triangle_wave
https://en.wikipedia.org/wiki/Sampling_(signal_processing)