Iterators
Announcements
List Practice
def chain(s):
    return [s[0], s[1:]]
silver = [2, chain([3, 4, 5])]
gold = [silver[0], silver[1].pop()]
silver[0] = 1
platinum = chain(chain([6, 7, 8]))

Reminder: s.pop() removes and returns the last item in list s.

>>> silver
[1, [3]]

>>> gold
[2, [4, 5]]

>>> platinum
[6, [[7, 8]]]
Tuples

(Demo)
Iterators
Iterators

A container can provide an iterator that provides access to its elements in order

iter(iterable): Return an iterator over the elements of an iterable value

next(iterator): Return the next element in an iterator

```python
>>> s = [3, 4, 5]
>>> t = iter(s)
>>> next(t)
3
>>> next(t)
4
>>> u = iter(s)
>>> next(u)
3
>>> next(t)
5
>>> next(u)
4
```
Discussion Question

What will be printed?

```python
a = [1, 2, 3]
b = [a, 4]
c = iter(a)
d = c
print(next(c))
print(next(d))
print(b)
```
Map Function

(Demo)
Discussion Question

`all(s)` iterates through `s` until a false value is found (or the end is reached).

What's printed when evaluating:

```python
x = all(map(print, range(-3, 3)))
```

Why?

- `print(-3)` returns `None` after displaying `-3`
- `None` is a false value
- `all([None, ...])` is False for any ...
- The map iterator never needs to advance beyond `-3`