• Questions/Announcements

• data collections: lists, cont’d
• repetition, part 2

• file processing

More than testing, designing tests is one of the best bug preventers known.
Recall: Updating a string

```python
>>> w1 = 'computer'
>>> w1 = w1.capitalize()
```
Updating a list

```python
>>> list1 = ['a', 'b', 99]
>>> list1 = list1.reverse()
>>> list1
```
Lists are mutable – be careful

```python
>>> list1 = ['a', 'b', 99]
>>> list1.reverse()
>>> list1
```
Assigning new string to a new variable

```python
>>> w1 = 'computer'
>>> w2 = w1.capitalize()
>>> w1
>>> w2
```
Assigning a new list to a new variable – Be careful

```python
>>> list1 = ['a', 'b', 99]
>>> list2 = list1.reverse()
>>> list1
>>> list2
```
Aliasing!

```python
>>> list1 = ['a', 'b', 99]
>>> list2 = list1
>>> list1.reverse()
>>> list1
>>> list2
```
Lists are mutable – Be careful

```python
>>> list1 = ['a', 'b', 99]
>>> list2 = list1.copy()  
   or  
>>> list2 = list1[:]
>>> list1.reverse()
>>> list1
>>> list2
```
Lists are mutable – Be careful

```python
def capital(s):
    '''(str) -> str

    returns input string s
    with first letter in uppercase

    >>> capital('abcefgij')
    'Abcdefghij'
    '''
    astring = s
    bstring = s.capitalize()
    return bstring

w1 = 'computer'
w2 = 'science'
z = capital(w1) + capital(w2)
```
Lists are mutable – Be careful

```python
def rev(li):
    """(li) -> li
    returns input list li reversed
    ""
    alist = li
    blist = li.reverse()
    return blist

>>> rev(['a', 'b', 'c'])
['c', 'b', 'a']
''
list1 = ['a', 'b', 'c']
list2 = [1, 2, 3]
y = rev(list1) + rev(list2)
```
Lists are mutable
so be very aware of what list methods do and what value they return:

- `append(item)`
- `sort()`
- `insert(i, item)`
- `reverse()`
- `extend(list)`
- `index(item)`
- `pop()`
- `count(item)`
- `copy()`
- `remove(item)`
Lists are mutable

- append(item)
- insert(i, item)
- extend(list)
- pop()
- copy()

- sort()
- reverse()
- index(item)
- count(item)
- remove(item)

Does the list method change the list? return a new list? another value? None?
>>> myL = [3.10, 2.30, 4.04, 5.20]
>>> myL = myL.sort()
>>> myL

>>> myL = [3.10, 2.30, 4.04, 5.20]
>>> myL.sort()
>>> myL
>>> myL = [3.10, 2.30, 4.04, 5.20]
>>> myL.sort()
>>> myL

>>> otherL = myL
>>> myL.reverse()
>>> myL

>>> otherL
>>> myL = [3.10, 2.30, 4.04, 5.20]
(a)
>>> myL = myL.sort()

(b)  
(c)  
>>> otherL = myL
>>> otherL = myL.copy()
>>> myL.sort()
>>> myL.reverse()
>>> myL
>>> otherL
>>> otherL
Python repetition, part 2.

recall: drawing a square

turtle.fd(100)
rt(90)
turtle.fd(100)
rt(90)
...

What kind of loop?

while loop

for loop
What kind of loop?

while loop – most general kind of loop
  indefinite # of iterations

for loop – operator for sequences
  definite number of iterations
drawing a square

ctr = 0

while ctr < 4:
    fd(100)
    rt(90)
    ctr += 1
>>> for ctr in <sequence>:
    fd(100); rt(90)
>>> for ctr in [1, 2, 3, 4]:
    fd(100); rt(90)
>>> range(4)
range(0, 4)

>>> list(range(4))
[0, 1, 2, 3]
>>> list(range(4))
[0, 1, 2, 3]

>>> list(range(1, 4))
[1, 2, 3]

>>> list(range(0, 11, 2))
[0, 2, 4, 6, 8, 10]

>>> list(range(10, 0, -2))
[10, 8, 6, 4, 2]
for ctr in <sequence>:
    fd(100); rt(90)

range(4)
for _ in range(4):
    fd(100); rt(90)
• Questions/Announcements

• Python file processing
Python file object

a text file from outside of the Python program

must be opened for use:
checkf = open('file_example.txt', 'r')

and closed when done:
checkf.close()
Python file object

checkf = open('file_example.txt', 'r')
contents = checkf.read()
checkf.close()

better:
with open('CIS122file_example.txt', 'r') as checkf:
    contents = checkf.read()
Python file objects

with open('file_example.txt', 'r') as checkf:
    contents = checkf.read()

opening a file sets a pointer to the start of the file.

file read methods move the pointer:
    read, readline, readlines
Python file objects

with open('file_example.txt', 'r') as checkf:
    contents = checkf.read()

file read methods move the pointer:
    read, readline, readlines

we can write to a file, too:
    write, writelines
Python file objects

files are sequences → can use Python for operator

with open('CIS122file_example.txt', 'r') as checkit:
    count = 0
    for line in checkit:
        count += 1