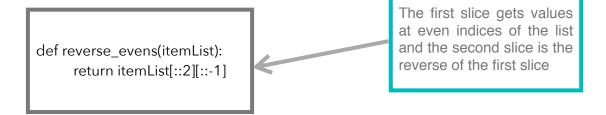
Python Collections 1 of 4

LISTS

FUNCTION NAME	DESCRIPTION
.remove(value)	Removes a value from a list
.pop()	Removes final value from a list
.pop(<index>)</index>	Removes item at index, assuming something exists at the index
.insert(<index>, <value>)</value></index>	Inserts <value> at <index> of list</index></value>

SLICES

```
>>> numbers = [1,2,3,4,5]
>>> numbers[0:2]
[1,2]
>>> numbers[2:10] # Index out of range
[3,4,5] # Will still print out the entire list even though out of range
>>> numbers = list(range(20))
>>> numbers
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
>>> numbers[2::2]
[2, 4, 6, 8, 10, 12, 14, 16, 18]
>>> numbers[-2: -5:-1]
[18, 17, 16]
```



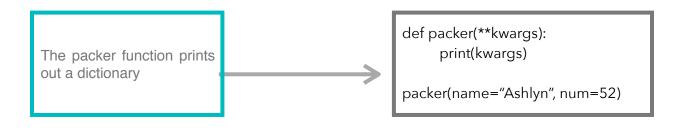
Python Collections 2 of 4

```
rainbow = ['red', 'orange', 'yellow', 'green', 'blue', 'pink']
rainbow[-2:] = "violet"
rainbow[-6:] = ["".join(rainbow[-6:])]

This writes each character in "violet" as its own list item. Must concatenate into a single list item.
```

DICTIONARIES

FUNCTION NAME	DESCRIPTION
.update()	Pass in dictionary of keys and values to create or update in a single step
.pop(<key>)</key>	Deletes the key and returns the values
.popitem()	Returns a random key/value pair
.keys()	Returns iterable of all keys in dictionary
.values()	Returns iterable of all values in dictionary
.items()	Returns iterable of key value pairs



```
def unpacker(name=None, num=None):
    if name and num:
        print "Hi {}".format(name)

unpacker(**{name: "Ashlyn", num=40)
```

Python Collections 3 of 4

=> TUPLES

-> Declaration:

```
my_tuple = (1,2,3)
my_tuple = 1,2,3
my_tuple = tuple([1,2,3])
```

- Tuples do not support item assignment
- -> Packing Tuples:

```
>>> a = 5

>>> b = 20

>>> a,b = b,a

>>> a

20

>>> c = b,a

>>> c

(5, 20)

>>>
```

-> Enumerating Tuples

```
>>> list(enumerate("abc"))
[(0, 'a'), (1, 'b'), (2, 'c')]
>>> for index,letter in enumerate("abcde"):
...     print("{} : {}".format(index, letter))
...
0 : a
1 : b
2 : c
3 : d
4 : e
```

-> Function that takes a sing string but returns a tuple of uppercase, lowercase, titlecase and reversed versions of the string.

```
def cases(string):
    return string.upper(), string.lower(), string.title(), string[::-1])
```

Python Collections 4 of 4

=> SETS

- Sets are used to make a iterable unique

pages = list(set(pages))

- Union = combination of two sets
- Difference = Everything unique in a set
- Intersection = elements in both sets

Union of two sets is a combination of both sets

```
>>> set1 = set(range(10))
>>> set2 = {1,2,3,5,7,11,13,17,19,23}
>>> set1 | set2
set([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 13, 17, 19, 23])
>>> set1 - set2
set([0, 8, 4, 6, 9])
>>> set1 & set2
set([1, 2, 3, 5, 7])

Difference of two sets returns everything unique to the first set
```

Intersection returns everything present in both sets