Quiz 1 review, part 2

Tuples, Lists, Dictionaries, Assertions, Recursion
Assertions

- Excellent tool for debugging
- Allow you to check that your variables are what you think they should be
- Fail fast

assert <boolean condition>
Tuples

ex. months = ('Sept','Oct','Nov','Dec')

● Are like lists, but immutable

● Why are they useful?
  ○ As keys in dictionaries (lists can’t because they are are mutable)
  ○ Swapping two variables (x, y = y, x)
  ○ Returning multiple values from a function
List indexing

```python
>>> myList = [3,5,2,7]
>>> myList[0]
3
>>> myList[1] = 6
[3, 6, 2, 7]
>>> myList[1:15:2]
[6, 7]
>>> del myList[1:2]
[3, 6]
```

```python
>>> myComplicatedList = [0, [3,4,5], [1,2]]
>>> myComplicatedList[1]
[3, 4, 5]
>>> myComplicatedList[1][0] = 9
>>> myComplicatedList[2] = [8,7]
>>> myComplicatedList[1:2]
[9, 4, 5, 8, 7]
>>> del myComplicatedList[1]
[0, [8, 7]]
```
List functions

```python
>>> letters = ['a','b','d']
>>> len(letters)
3
>>> letters.append('e')
['a', 'b', 'd', 'e']
>>> letters.insert(2, 'c')
['a', 'b', 'c', 'd', 'e']
>>> letters.remove('a')
['b', 'c', 'd', 'e']
>>> letters.reverse()
['e', 'd', 'c', 'b']
>>> letters.pop()
'b'
>>> letters
['e', 'd', 'c']
>>> letters.extend(['b', 'a'])
['e', 'd', 'c', 'b', 'a']
```
Dictionaries

- Key, value pairs
- Keys can be integers, strings, tuples, etc.
  (anything immutable)
- Keys can’t be lists, dictionaries, etc.
  (anything mutable)
- Keys are unique, values don’t have to be
Using dictionaries

```python
>>> zoo = {'elephant' : 3, 'giraffe' : 4}
>>> len(zoo)
2
>>> zoo['elephant']
3
>>> zoo['frog']
KeyError: 'frog'
>>> if 'cheetah' not in zoo:
    zoo['cheetah'] = 5
>>> zoo
{'cheetah': 5, 'giraffe': 4, 'elephant': 3}
>>> zoo.keys()
['cheetah', 'giraffe', 'elephant']
>>> zoo.values()
[5, 4, 3]
>>> del zoo['elephant']
>>> zoo
{'cheetah': 5, 'giraffe': 4}
```
Mutability

Mutable
- Lists
- Dictionaries

Immutable
- Strings
- Tuples
- Dictionary keys
Aliasing

- Two variables are bound to the same object
- Copying (read more about it in the extra notes section of Stellar)
  - Shallow copy
  - copy()
  - deepcopy()
Recursion

A child couldn't sleep, so her mother told a story about a little frog, who couldn't sleep, so the frog's mother told a story about a little bear, who couldn't sleep, so bear's mother told a story about a little weasel...who fell asleep. ...and the little bear fell asleep; ...and the little frog fell asleep; ...and the child fell asleep.
Recursion

- When a function solves a problem by calling itself on a smaller version of the problem
- Requires 2 things
  - Base case
  - Recursive (inductive) step

Will be covered more in recitation tomorrow
Code examples