Lab2_JakeDineen

July 13, 2018

In [64]:
```
#Load Dicts
stock = {"banana": 6, "apple": 0, "orange": 32, "pear": 15}
prices = {"banana": 4, "apple": 2, "orange": 1.5, "pear": 3}
```

0.1 a.

Show the expression that gets the value of the stock dictionary at the key 'orange'.

Show a statement that adds an item to the stock dictionary called 'cherry' with some integer value and that adds 'cherry' to the prices dictionary with a numeric value. (Or pick your own fruit name.)

In [83]:
```
#Checking the value for key 'orange' in stock dict
stock['orange']
print('The value of orange is: {}'.format( stock['orange']))

#Add cherry to the stock dict
stock['cherry'] = 4
prices['cherry'] = 1

#Simple check
if 'cherry' in stock:
    print('Cherry is now in the stock dict')
if 'cherry' in prices:
    print('Cherry is now in the prices dict')
```

```
The value of orange is: 32
Cherry is now in the stock dict
Cherry is now in the prices dict
```

0.2 b.

Write the code for a loop that iterates over the stock dictionary and prints each key and value.

In [84]:
```
for key, value in stock.items(): #This method returns a list of tuple pairs.
    print(key, value)
#https://www.tutorialspoint.com/python/dictionary_items.htm
```

```
banana 6
apple 0
orange 32
pear 15
cherry 4
```

0.3 c.

Suppose that we have a list:
groceries = ['apple', 'banana', 'pear']
Write the code that will sum the total number in stock of the items in the groceries list.

```
In [95]: #Load List
         groceries = ['apple', 'banana', 'pear']

         import numpy as np #For operations

         #Trying out mapping
         np.sum(list(map(stock.get, groceries)))
         print('Stock Count mapping: {}'.format(np.sum(list(map(stock.get, groceries)))))

         #Trying List Comprehension
         np.sum([stock[i] for i in groceries])
         print('Stock Count List Comprehension: {}'.format(np.sum([stock[i] for i in groceries]

         #Tried out two different methods here. Listcomp was more familiar and easier to under
         #Mapping is faster when scaling.
```

```
Stock Count mapping: 21
Stock Count List Comprehension: 21
```

0.4 d.

Write the code that can print out the total value in stock of all the items.

This program can iterate over the stock dictionary and for each item multiply the number in stock times the price of that item in the prices dictionary. (This can include the items for 'cherry' or not, as you choose.)

```
In [96]: #Creating a new dict with the stored keys/values of the multiplication
         print({key: price * stock[key] for key, price in prices.items()} )

         #Print out the total cost/rev of inventory in stock.
         print('Total Price*Stock of Inventory = ${}'.format(sum(price * stock[key] for key, pr
```

```
{'banana': 24, 'apple': 0, 'orange': 48.0, 'pear': 45, 'cherry': 4}
Total Price*Stock of Inventory = $121.0
```