**Dictionaries Tuples & List**

**1.** A block of publicly traded stock has a variety of attributes, we'll look at a few of them. A stock has a ticker symbol and a company name. Create a simple dict with ticker symbols and company names.

For example:

stockDict = { 'GM': 'General Motors',

'CAT':'Caterpillar', 'EK':"Eastman Kodak" }

Create a simple list of blocks of stock. These could be tuples with ticker symbols, prices, dates and number of shares. For example:

purchases = [ ( 'GE', 100, '10-sep-2001', 48 ),

( 'CAT', 100, '1-apr-1999', 24 ),

( 'GE', 200, '1-jul-1998', 56 ) ]

Create a purchase history report that computes the full purchase price (shares times dollars) for each block of stock and uses the stockDict to look up the full company name. This is the basic relational database join algorithm between two tables.

Create a second purchase summary that which accumulates total investment by ticker symbol. In the above sample data, there are two blocks of GE. These can easily be combined by creating a dict where the key is the ticker and the value is the list of blocks purchased. The program makes one pass through the data to create the dict. A pass through the dict can then create a report showing each ticker symbol and all blocks of stock.

2. Write a function called accept login(users, username, password) with three parameters: users a dictionary of username keys and password values, username a string for a login name and password a string for a password. The function should return True if the user exists and the password is correct and False otherwise. Here is the calling code, test your code with both good and bad passwords as well as non-existent login names:

users = { "user1" : "password1", "user2" : "password2", "user3" : "password3" }

if accept\_login(users, "wronguser", "wrongpassword") :

print("login successful!")

else :

print("login failed...")

3. Write a function to invert a dictionary. It should accept a dictionary as a parameter and return a dictionary where the keys are values from the input dictionary and the values are lists of keys from the input dictionary.

For example, this input: { "key1" : "value1", "key2" : "value2", "key3" : "value1" }

Should return this

Dictionary: { "value1" : ["key1", "key3"], "value2" : ["key2"] }

4. Write a function called word frequencies(mylist) that accepts a list of strings called mylist and returns a dictionary where the keys are the words from mylist and the values are the number of times that word appears in mylist:

word\_list = list("aaaaabbbbcccdde")

word\_freq = { ’a’ : 5, ’b’ : 4, ’c’ : 3, ’d’ : 2, ’e’ : 1 }

if word\_frequencies(word\_list) == word\_freq :

print("correct")

else :

print("wrong")

**5.**Given the following dictionary:

inventory = {

'gold' : 500,

'pouch' : ['flint', 'twine', 'gemstone'],

'backpack' : ['xylophone','dagger', 'bedroll','bread loaf']

}

Try to do the followings:

* Add a key to inventory called 'pocket'.
* Set the value of 'pocket' to be a list consisting of the strings 'seashell', 'strange berry', and 'lint'.
* .sort()the items in the list stored under the 'backpack' key.
* Then .remove('dagger') from the list of items stored under the 'backpack' key.
* Add 50 to the number stored under the 'gold' key.

**6.** Folow the steps bellow: -Create a new dictionary called prices using {} format like the example above.

* Put these values in your prices dictionary:
* "banana": 4,
* "apple": 2,
* "orange": 1.5,
* "pear": 3
* Loop through each key in prices. For each key, print out the key along with its price and stock information. Print the answer in the following format:
* apple
* price: 2
* stock: 0
* Let's determine how much money you would make if you sold all of your food.
  + Create a variable called total and set it to zero.
  + Loop through the prices dictionaries.For each key in prices, multiply the number in prices by the number in stock. Print that value into the console and then add it to total.
  + Finally, outside your loop, print total.

**7.** Follow the steps:

* First, make a list called groceries with the values "banana","orange", and "apple".
* Define this two dictionaries:

stock = {

"banana": 6,

"apple": 0,

"orange": 32,

"pear": 15

}

prices = {

"banana": 4,

"apple": 2,

"orange": 1.5,

"pear": 3

}

* Define a function compute\_bill that takes one argument food as input. In the function, create a variable total with an initial value of zero.For each item in the food list, add the price of that item to total. Finally, return the total. Ignore whether or not the item you're billing for is in stock.Note that your function should work for any food list.
* Make the following changes to your compute\_bill function:
  + While you loop through each item of food, only add the price of the item to total if the item's stock count is greater than zero.
  + If the item is in stock and after you add the price to the total, subtract one from the item's stock count.

**8.** This exercise is a bit more complicate. We will review all about list and dictionaries. The aim of this exercise is to make a gradebook for teacher's students.

Try to follow the steps:

* Create three dictionaries: lloyd, alice, and tyler.
* Give each dictionary the keys "name", "homework", "quizzes", and "tests".Have the "name" key be the name of the student (that is, lloyd's name should be "Lloyd") and the other keys should be an empty list. Look in solutions, the "solution 1". Chechk if you have done it rigth.
* Now copy this code:
* lloyd = {
* "name": "Lloyd",
* "homework": [90.0,97.0,75.0,92.0],
* "quizzes": [88.0,40.0,94.0],
* "tests": [75.0,90.0]
* }
* alice = {
* "name": "Alice",
* "homework": [100.0, 92.0, 98.0, 100.0],
* "quizzes": [82.0, 83.0, 91.0],
* "tests": [89.0, 97.0]
* }
* tyler = {
* "name": "Tyler",
* "homework": [0.0, 87.0, 75.0, 22.0],
* "quizzes": [0.0, 75.0, 78.0],
* "tests": [100.0, 100.0]
* }
* Below your code, create a list called studentsthat contains lloyd, alice, and `tyler.
* for each student in your students list, print out that student's data, as follows:
  + print the student's name
  + print the student's homework
  + print the student's quizzes
  + print the student's tests
* Write a function average that takes a list of numbers and returns the average.
  + Define a function called average that has one argument, numbers.
  + Inside that function, call the built-in sum() function with the numbers list as a parameter. Store the result in a variable called total.
  + Use float() to convert total and store the result in total.
  + Divide total by the length of the numbers list. Use the built-in len() function to calculate that.
  + Return that result.
* Write a function called get\_average that takes a student dictionary (like lloyd, alice, or tyler) as input and returns his/her weighted average.
  + Define a function called get\_average that takes one argument called student.
  + Make a variable homework that stores the average() of student["homework"].
  + Repeat step 2 for "quizzes" and "tests".
  + Multiply the 3 averages by their weights and return the sum of those three. Homework is 10%, quizzes are 30% and tests are 60%.
* Define a new function called get\_letter\_grade that has one argument called score. Expect score to be a number.
  + Inside your function, test score using a chain of if: / elif: / else: statements, like so:
  + If score is 90 or above: return "A"
  + Else if score is 80 or above: return "B"
  + Else if score is 70 or above: return "C"
  + Else if score is 60 or above: return "D"
  + Otherwise: return "F"
  + Finally, test your function. Call your get\_letter\_grade function with the result of get\_average(lloyd). Print the resulting letter grade.
* Define a function called get\_class\_average that has one argument, students. You can expect students to be a list containing your three students.
  + First, make an empty list called results.
  + For each student item in the class list, calculate get\_average(student) and then call results.append() with that result.
  + Finally, return the result of calling average() with results.
* Finally, print out the result of calling get\_class\_averagewith your students list. Your students should be [lloyd, alice, tyler].
* Then, print the result of get\_letter\_grade for the class's average.