Quest1

August 27, 2018

1 CSE 5345 Quest 1

Basic Python Quest
CLASS = CSE5345
NAME = Lee, SeungKi
STUDENT_ID = 35460312
CODING_NAME = MonsterPeach

1.1 1. `isSorted(list)`

Iterate through the list and see if value of the left is ever greater than the right

```
In [1]: def isSorted(list):

            for i in range (len(list)-1):
                if list[i] > list[i+1]:
                    return False
            return True
```

```
In [2]: print(isSorted([2,4,7,7,99,122]))
        print(isSorted(['a', 'b', 'c']))
        print(isSorted(['a', 'z', 'b', 'c']))
```

```
True
True
False
```

1.2 2. `hasUniqueValues(list)`

Put every element of a list into a dictionary as keys, then check the number of keys against number of elements.

```
In [3]: def hasUniqueValues(list):

            return len({}.fromkeys(list)) == len(list)
```

```
In [4]: print(hasUniqueValues([2,4,7,99,122, 7]))
        print(hasUniqueValues(['a', 'b', 'c']))
        print(hasUniqueValues(['a', 'z', 'b', 'c', 'c']))
```

```
False
True
False
```

1.3 3. isSortedAndUnique(list)

Just utilize two functions above

```
In [5]: def isSortedAndUnique(list):

            return isSorted(list) and hasUniqueValues(list)
```

```
In [6]: print(isSortedAndUnique([2,4,7,7,99,122]))
        print(isSortedAndUnique(['a','b','c']))
        print(isSortedAndUnique(['a','z','b','b','c']))
```

```
False
True
False
```

1.4 4. genSortedArrayUniqueValues(list)

use fromkeys of dictionary to eleminate duplicates, then sort and return the list

> Dr. Coyle, I am using the sorted() function here because it does not apprear in the constraints for this function. If you would like me to write my own sort functions, I would be happy to do it. Please let me know.

```
In [7]: def genSortedArrayUniqueValues(list):

            return_list = []

            for key in {}.fromkeys(list):
                return_list.append(key)

            return sorted(return_list)
```

```
In [8]: print(genSortedArrayUniqueValues([2,4,7,7,122,99]))
        print(genSortedArrayUniqueValues(['a','b','z','c', 'a']))
```

```
[2, 4, 7, 99, 122]
['a', 'b', 'c', 'z']
```

1.5 5. listToMapTwoByTwo(list)

check the size of the list, then even items in the list will be key, odd items will be value

```
In [9]: def listToMapTwoByTwo(list):

            return_dict = {}

            if len(list) > 1:
                for i in range (len(list)-1):
                    if i%2 == 0:
                        return_dict[list[i]] = list[i+1]

            return return_dict
```

```
In [10]: print(listToMapTwoByTwo(['foo','bar']))
         print(listToMapTwoByTwo(['a',2, 3,'foo']))
         print(listToMapTwoByTwo([]))
```

```
{'foo': 'bar'}
{'a': 2, 3: 'foo'}
{}
```

1.6 6. wordsInStringToDictWordCount(str)

Split the string by whitespace character and put them in a list. For every item in the list, if the item is already in the dictionary add to its count. If not, set its count to 1. Return the count.

```
In [11]: def wordsInStringToDictWordCount(str):

             return_dict = {}

             if len(str) > 1:
                 temp_list = str.split()

                 for i in temp_list:
                     if i in return_dict:
                         return_dict[i] += 1
                     else:
                         return_dict[i] = 1

             return return_dict
```

```
In [12]: print(wordsInStringToDictWordCount("foo bar   bar"))
         print(wordsInStringToDictWordCount(''))
```

```
{'foo': 1, 'bar': 2}
{}
```

1.7 7. reverseWordsInString(str)

Split by whitespace and put str into a list, reverse the list using [::-1], then join the list as char. Make sure to leave a space between the joined items

```
In [13]: def reverseWordsInString(str):

             temp_list = str.split()
             temp_list = temp_list[::-1]

             return ' '.join(temp_list)
```

```
In [14]: print(reverseWordsInString('foo bar bar baz'))
```

```
baz bar bar foo
```

1.8 8. def genListOfOverlaps(list1, list2)

Put unique items of list1 into a dict. Put unique items of list2 into the dict using word counter. Return a list with value = 2

Dr. Coyle,

If I can use set it would be much shorter... but I am not sure if set is one of the constraints for this function. If you would like a better big(o), please let me know and I will happily write the function using intersection.

```
In [15]: def genListOfOverlaps(list1, list2):

             return_list = []
             temp_dict = {}

             for i in list1:
                 if i not in temp_dict:
                     temp_dict[i] = 1
             for i in {}.fromkeys(list2):
                 if i in temp_dict:
                     temp_dict[i] += 1
                 else:
                     temp_dict[i] = 1

             for key in temp_dict:
                 if temp_dict[key] == 2:
                     return_list.append(key)

             return sorted(return_list)
```

```
In [16]: print(genListOfOverlaps([2,4,6,8],[6,2,2,9,7]))
         print(genListOfOverlaps([2,4,6,8],[2,4,6,8]))
         print(genListOfOverlaps([2,4,6,8],[1,1,9,7]))
```

```
[2, 6]
[2, 4, 6, 8]
[]
```

1.9 9. `def removeDupsNoSet(list):`

use {}.fromkeys(list) and append them to a new list

```
In [17]: def removeDupsNoSet(list):

             return_list = []

             for i in {}.fromkeys(list):
                 return_list.append(i)

             return return_list
```

```
In [18]: print(removeDupsNoSet([1,1,2,2,5,6]))
```

```
[1, 2, 5, 6]
```

1.10 10. `removeDupsUseSet(list1):`

One Liner

```
In [19]: def removeDupsUseSet(list1):

             return list(set(list1))
```

```
In [20]: removeDupsUseSet([1,1,2,2,5,6])
```

```
Out[20]: [1, 2, 5, 6]
```