In class Practice 2/7/2020

Part I

1. A(n) _____ is an unordered collection of items with no duplicates.
   a. set
   b. file
   c. dictionary
   d. tuple

2. Elements of a set are delimited with _____.
   a. { }
   b. [ ]
   c. ( )
   d. < >

3. The statement set1.union(set2) is:
   a. the set containing the elements that are in either set1 and set2 without duplicates
   b. the set containing the elements that are in both set1 and set2
   c. the set containing the elements that are in set1 with the elements of set2 removed
   d. the set containing the elements that are in set2 with the elements of in set1 removed

4. The statement set1.intersection(set2) is:
   a. the set containing the elements that are in both set1 and set2
   b. the set containing the elements that are in either set1 and set2 without duplicates
   c. the set containing the elements that are in set1 with the elements of set2 removed
   d. the set containing the elements that are in set2 with the elements of in set1 removed

5. The statement set1.difference(set2) is:
   a. the set containing the elements that are in set1 with the elements of set2 removed
   b. the set containing the elements that are in set2 with the elements of in set1 removed
   c. the set containing the elements that are in both set1 and set2
   d. the set containing the elements that are in either set1 and set2 without duplicates

6. What is the output of the following Python statement?

   print (set("bookkeeper"))

   a. {'b', 'o', 'k', 'e', 'p', 'r'}
   b. {'b', 'o', 'o', 'k', 'k', 'e', 'e', 'p', 'e', 'r'}
   c. {'o', 'k', 'e'}
   d. {'b', 'p', 'r'}

7. In a dictionary, a pair such such as “dog” : “rover” is called a(n) _____.
    a. item
    b. pair
    c. key
    d. couple

8. In a dictionary, keys must be immutable objects.
    a. true
    b. false

9. It is common to create dictionaries from text files.
    a. true
    b. false

10. Dictionaries cannot have other dictionaries as values.
    a. true
    b. false

11. A dictionary is an ordered structure that can be sorted.
    a. true
    b. false

12. Dictionaries cannot be created with comprehension.
    a. true
    b. false

Part II Short Answers

1. Use dict comprehension to generate the following Python dict. Note asci value of 'a' is 97. Hint use chr() function.

```
{97: 'a', 98: 'b', 99: 'c', 100: 'd', 101: 'e', 102: 'f', 103: 'g'}
```

2. Use list comprehension to create the similar one as follows

```
['A', 'B', 'C', 'D', 'E', 'F', 'G']
```

3. Write a single Python statement to convert the list ['spring', 'summer', 'fall', 'winter'] to a set called *seasons*.
4. Why can't elements of a set be indexed?
5. Why can't lists and sets serve as keys for dictionaries?

1.{i:chr(i) for i in range(97,104)}
2.[chr(i) for i in range(65,72)]
3.season = set(['spring','summer','fall','winter'])
4.You cannot access items in a set by referring to an index, since sets are unordered the items has no index. But you can loop through the set items using a for loop, or ask if a specified value is present in a set, by using the in keyword.
5. because lists are mutable, dict keys (and set members) need to be hashable, and hashing mutable objects is a bad idea because hash values should be computed on the basis of instance attributes. ... After mutating stupid , it cannot be found in the dict any longer because the hash changed.

Part III

1. Determine the set operations (op1 to op5) so that one can generate the solutions after the # in the right.

union
intersection
difference
symmetric_difference
issubset

```
{1, 2, 4, 8, 16} op1 {1, 4, 16, 64, 256} # {1,2,4,8,16,64,256}
{1, 2, 4, 8, 16} op2 {1, 4, 16, 64, 256} # {1,4,16}
{1, 2, 4, 8, 16} op3 {1, 4, 16, 64, 256} # {2,8}
{1, 2, 4, 8, 16} op4 {1, 4, 16, 64, 256} # {2,8,64,256}
{1, 2, 4, 8, 16} op5 {1, 4, 16, 64, 256} # False
```

2. Write a Python function that receives a list of words, then determines and display ni alphabetical order only the unique words. Treat upper and lower case the same.

3. Write a python program that uses a dict as the following.

```
tlds = {'Canada': 'ca', 'United States': 'us', 'Mexico': 'mx'}
```

   (a) Check if this dict contain 'Canada'
   (b) Check if this dict contains 'France'
   (c) Iterate the key-value pairs and output in two-column format as the following

```
Canada              ca
United States       us
Mexico              mx
```

   (d) Add key value pair 'Sweden' and 'sw'
   (e) Update the value of 'Sweden' to 'se'

```
{'Canada': 'ca', 'United States': 'us', 'Mexico': 'mx', 'Sweden': 'se'}
```

   (f) Use dict comprehension to reverse keys and values as the following

```
{'ca': 'Canada', 'us': 'United States', 'mx': 'Mexico', 'se': 'Sweden'}
```

   (g) Change the country names to all upper case by using result of (f)

```
{'ca': 'CANADA', 'us': 'UNITED STATES', 'mx': 'MEXICO', 'se': 'SWEDEN'}
```

```
def f(x):
    res = {}
    final = []
    for i in x:
        i = i.lower()
        if i in res:
            res[i] +=1
        else:
            res[i] = 1
    for k,v in res.items():
        if v == 1:
            final.append(k)
    return sorted(final)
```

```
tlds = {"Canada" : "ca","United States": 'us',"Mexico":'mx'}
"Canada" in tlds
"France" in tlds
for k,v in tlds.items():
    print(k,v)
tlds["Sweden"] = 'sw'
tlds["Sweden"] = 'se'
f = {v:k for k,v in tlds.items()}
{k:v.upper() for k,v in f.items()}
```