PW Skills - Assignment

(Tuples, Sets, Dictionary)

~Huzaifa Khan

Q1. What are the characteristics of the tuples? Is tuple immutable?

Ans. Tuples are ordered collections of elements, which can be of different data types such as integers, floats, strings, or even other tuples.

1. The main characteristics of tuples are:
    a. Immutable
    b. Heterogeneous
    c. Ordered
    d. Iterable

2. No Tuples are Mutable.

Q2. What are the two tuple methods in python? Give an example of each method. Give a reason why tuples have only two in-built methods as compared to Lists.

Ans.

1. The Two tuple methods in python are:
    a. Count ()
    b. Index ()

Examples:

Count()

```
In [1]: tup=(1,2,3,4,4,5)
        print(tup.count(4))
```

```
2
```

Index()

```
In [2]: tup=(1,2,3,4,5)
        print(tup.index(3))
```

```
2
```

2. The reason why tuples have only two in-built methods as compared to lists is that tuples are immutable, while lists are mutable. This means that tuples do not require as many methods as lists since tuples cannot be modified. The two methods provided for tuples are useful for accessing and manipulating the data in the tuple, without altering its contents. Lists, on the other hand, have more in-built methods since they can be modified and therefore require additional methods for appending, removing, and sorting elements.

Q3. Which collection datatypes in python do not allow duplicate items? Write a code using a set to remove duplicates from the given list.
List = [1, 1, 1, 2, 1, 3, 1, 4, 2, 1, 2, 2, 2, 3, 2, 4, 3, 1, 3, 2, 3, 3, 3, 4, 4, 1, 4, 2, 4, 3, 4, 4]

Ans. The collection of datatypes in Python that doesn't allow duplicate items is the 'Set'.

1. The required code is:

Question 3:

```
In [3]: my_list = [1, 1, 1, 2, 1, 3, 1, 4, 2, 1, 2, 2, 2, 3, 2, 4, 3, 1, 3, 2, 3, 3, 3, 4, 4, 1, 4, 2, 4, 3, 4, 4]

        my_set = set(my_list)
        my_list = list(my_set)
        print(my_list)

        [1, 2, 3, 4]
```

Q4. Explain the difference between the union () and update () methods for a set. Give an example of each method.

Ans. The 'Union ()' and 'Update ()'methods are used to combine sets in Python. However, there is a small difference between these two methods.

1. 'Union ()' : This method returns a new set containing all the elements from the original set and another set or iterable. The original sets remain unchanged. If there are duplicate elements in the sets, the resulting set will only contain one instance of each unique element.
   Eg:

Question 4 (Union)

```
In [6]: set_a = {1, 2, 3}
        set_b = {3, 4, 5}
        set_c = set_a.union(set_b)
        print(set_c)

        {1, 2, 3, 4, 5}
```

2. 'Update ()' : This method adds all the elements from another set or iterable to the original set. The original set is modified in place. If there are duplicate elements in the sets, the resulting set will only contain one instance of each unique element.
   Eg:

Question 4 (Update)

```
In [8]: set_a = {1, 2, 3}
        set_b = {3, 4, 5}
        set_a.update(set_b)
        print(set_a)

        {1, 2, 3, 4, 5}
```

Q5. What is a dictionary? Give an example. Also, state whether a dictionary is ordered or unordered.

Ans. A dictionary is a collection data type in Python that stores data as key-value pairs. Each key is unique, and the corresponding value can be any data type such as integers, strings, lists, or other dictionaries. Dictionaries are enclosed in curly braces '{}' and the key-value pairs are separated by a colon ':'

Eg:

Question 5

```
In [9]: d1 = {"Name":"Huzi","Age":19,"Mobile Num":92969420}

In [10]: print(d1)
{'Name': 'Huzi', 'Age': 19, 'Mobile Num': 92969420}
```

Dictionaries are unordered, which means that the order in which the items are inserted into the dictionary is not preserved. Meaning that if we iterate over the items in a dictionary, we cannot rely on the order of the items.

Q6. Can we create a nested dictionary? If so, please give an example by creating a simple one-level nested dictionary.

Ans. Yes, we can create a nested dictionary in Python, where a dictionary can contain other dictionaries as values.

Eg:

Question 6

```
In [12]: person = { "name": "John",
             "age": 30,
             "address": {
                 "street": "123 Main St",
                 "city": "New York",
                 "state": "NY",
                 "zipcode": "10001" }}

In [13]: print(person)
{'name': 'John', 'age': 30, 'address': {'street': '123 Main St', 'city': 'New York', 'state': 'NY', 'zipcode': '10001'}}
```

Q7. Using setdefault() method, create key named topics in the given dictionary and add the value of the key as this list ['Python', 'Machine Learning', 'Deep Learning']
dict1 = {'language' : 'Python', 'course': 'Data Science Masters'}
Ans: Code:

```
In [66]: dict1 = {'language' : 'Python', 'course': 'Data Science Masters'}
         print(dict1)

         {'language': 'Python', 'course': 'Data Science Masters'}

In [67]: dict1.setdefault("topics", ['Python', 'Machine Learning', 'Deep Learning'])
         print(dict1)

         {'language': 'Python', 'course': 'Data Science Masters', 'topics': ['Python', 'Machine Learning', 'Deep Learning']}
```

Q8. What are the three view objects in dictionaries? Use the three in-built methods in python to display these three view objects for the given dictionary.
dict1 = {'Sport': 'Cricket' , 'Teams': ['India', 'Australia', 'England', 'South Africa', 'Sri Lanka', 'New Zealand']}
Ans: The three view objects in dictionaries are:

1. Keys: a view object that contains the keys of the dictionary.
2. Values: a view object that contains the values of the dictionary.
3. Items: a view object that contains the key-value pairs of the dictionary as tuples.

Code:

Question 8

```
In [14]: dict1 = {'Sport': 'Cricket' , 'Teams': ['India', 'Australia', 'England', 'South Africa', 'Sri Lanka', 'New Zealand']}

         # Display dict_keys object
         keys = dict1.keys()
         print(keys)

         # Display dict_values object
         values = dict1.values()
         print(values)

         # Display dict_items object
         items = dict1.items()
         print(items)

         dict_keys(['Sport', 'Teams'])
         dict_values(['Cricket', ['India', 'Australia', 'England', 'South Africa', 'Sri Lanka', 'New Zealand']])
         dict_items([('Sport', 'Cricket'), ('Teams', ['India', 'Australia', 'England', 'South Africa', 'Sri Lanka', 'New Zealand'])])
```