1. What is the output of the following code
   atuple = (100, 200, 300, 400, 500)
   print(atuple[-2])
   print(atuple[-4:-1])

   1.IndexError: tuple index out of range
   2.400
   3. (200, 300, 400)

- ○ 1,2
- ◉ 2,3
- ○ 1,3
- ○ None

Explanation: Use the range of negative indexes to start a search from the end of the tuple.

2. What is the type of the following variable
   atuple = ("Orange")
   print(type(atuple))

- ○ list
- ○ tuple
- ○ array
- ◉ str

Explanation: Explanation: To create a tuple with a single item, you need to add a comma after the item. Otherwise, Python will not recognize the variable as a tuple, and it will treat it as a string type.

3. What is the output of the following tuple operation
   atuple = (100,)
   print(atuple * 2)

- ○ TypeError
- ◉ (100, 100)
- ○ (200)
- ○ a & b

Explanation: We can use * operator to repeat the tuple values n number of times.

4. What is the output of the following dictionary operation
   dict1 = {"name": "Mike", "salary": 8000}
   temp = dict1.get("age")
   print(temp)

- ○ KeyError: 'age'
- ◉ None

Explanation: The get() method returns a value of the key. If the key is not found, it returns None, instead of throwing a KeyError exception.

5. What is the output of the following code
   dict1 = {"key1":1, "key2":2}
   dict2 = {"key2":2, "key1":1}
   print(dict1 == dict2)

- ◉ True
- ○ False
- ○ All of the above
- ○ none

Explanation: We can use the == and != operators to check whether the dictionary contains the same items.

6. Select all the correct ways to copy two sets
   1.set2 = set1.copy()
   2.set2 = set(set1)
   3.set2.update(set1)
   4.set2 = set1

- ◉ 1&2
- ○ 1,2&3
- ○ 2,3&4
- ○ None of the above

Explanation: When you set set2= set11, you are making them refer to the same dict object, so when you modify one of them, all references associated with that object reflect the current state of the object. So don't use the assignment operator to copy the set, instead use the copy() method or set() constructor.

7. What is the output of the following set operation
   set1 = {"Yellow", "Orange", "Black"}
   set2 = {"Orange", "Blue", "Pink"}
   set3 = set2.difference(set1)
   print(set3)

- ○ {'Yellow', "Black', 'Pink', 'Blue'}
- ◉ {'Pink', 'Blue'}
- ○ {'Yellow', "Black'}
- ○ All of the above

Explanation: The difference() method returns a set that contains the difference between two sets. Here set3 = set2.difference(set1) so the returned set contains items that exist only in the first set, and not in both sets.

8. Select all which is true for Python set
   1.Sets are unordered
   set doesn't allow duplicate
   sets are written with curly brackets {}
   2.set object does support indexing
   set is mutable

- ○ Both 1&2
- ◉ 1
- ○ 2
- ○ None

Explanation: We mostly use sets for mathematical operations such as union and intersection. Sets are unordered; it means item order isn't fixed. So we cannot be sure in which order the items will appear. The set is mutable. We can add or remove items from it when required.