Day 8
14/8/22

→ A tuple is a collection which is ordered and unchangeable and allow duplicate values.

→ Difference b/w tuples and the lists is that the tuples cannot be changed unlike lists.

→ Tuples uses parentheses (round bracket) lists use square brackets.

→ Tuples are sequences, just like lists.

→ Tuples are immutable - "never-changing"

→ Tuples items are indexed, the first item has index [0], the second item has index [1] etc.

1) Allow Duplicates :-
Since tuples are indexed, they can have items with the same value.

ex:- tuple1 = ("apple", "banana", "apple", "cherry")
print (tuple1)

2) Tuple length :-
To determine how many items a tuple has, use len() function.

ex:- print (len(tuple1))

3) tuple() constructor -

note the double round-brackets.

ex: tuple1 = tuple(("apple", "banana", "cherry"))
print(tuple1)

4) tuple items - Data Types :-

It can be of any data type with strings, integers and boolean values:

ex:- tuple1 = ("abc", 34, True, 40, "male");

5) type() -

tuples are defined as objects with the data type 'tuple':

ex: mytuple = ("apple", "banana", "cherry")
print(type(mytuple))

6) Create Tuple with one item -

To create tuple with one item you have to add a comma after item.

ex:- thistuple = ("apple",)
print(type(thistuple))

7) Access Tuple items -

you can access tuple items by referring to the index number, inside square brackets.

ex:- thistuple = ("apple", "banana")
print(thistuple[1])

8) Negative indexing -

Negative indexing means start from the end. -1 refers to last item

ex:- tuple2 = ("apple", "banana", "cherry")
print (tuple2 [-1])

9) Range of indexes -

Specify a range of indexes by specifying where to start & where to end the range.

ex:- thistuple = ("apple", "banana", "cherry", "kiwi", "mango")
print (thistuple [2:4])

10) Add items -

Since tuples are immutable, they do not have a build-in append() method.

~~you~~ way to add items to a tuple is: convert it into list, add your item and convert it back to tuple.

ex:- i) thistuple = ("apple", "banana", "cherry")
y = list (thistuple)
y.append ("orange")
thistuple = tuple (y)

ii) thistuple = ("apple", "banana", "cherry")
y = ("orange",)
thistuple += y
print (thistuple)

11) delete the tuple completely.

ex:- thistuple = ("apple", "kiwi", "cherry")
del thistuple
print(thistuple)

Sets.

→ Sets are used to store multiple items in a single variable.

→ A set is a collection which is unordered, unchangeable, and unindexed.

→ Each element in the set must be unique, immutable

→ collection of unique values.

→ set has no duplicate elements

→ The major advantage of using a set, as opposed to list, is that it has a highly optimized method for checking whether a specific element is contained in the set.

→ Sets are unordered, we cannot access items using indexes like we do in lists