Access Tuple Items

You can access tuple items by referring to the index number, inside square brackets:

Example

Print the second item in the tuple:

thistuple = ("apple", "banana", "cherry")

print(thistuple[1])

Note: The first item has index 0.

Output:

banana

Negative Indexing

Negative indexing means start from the end.

-1 refers to the last item, -2 refers to the second last item etc.

Example

Print the last item of the tuple:

thistuple = ("apple", "banana", "cherry")

print(thistuple[-1])

output:

cherry

Range of Indexes

You can specify a range of indexes by specifying where to start and where to end the range.

When specifying a range, the return value will be a new tuple with the specified items.

Example

Return the third, fourth, and fifth item:

Note: The search will start at index 2 (included) and end at index 5 (not included).

Remember that the first item has index 0.

By leaving out the start value, the range will start at the first item:

Example

This example returns the items from the beginning to, but NOT included, "kiwi":

thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")

print(thistuple[:4])

output:

('apple', 'banana', 'cherry', 'orange')

By leaving out the end value, the range will go on to the end of the list:

Example

This example returns the items from "cherry" and to the end:

thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")

print(thistuple[2:])

output:

('cherry', 'orange', 'kiwi', 'melon', 'mango')

Range of Negative Indexes

Specify negative indexes if you want to start the search from the end of the tuple:

Example

This example returns the items from index -4 (included) to index -1 (excluded)

thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")

print(thistuple[-4:-1])

output:

('orange', 'kiwi', 'melon')

Check if Item Exists

To determine if a specified item is present in a tuple use the in keyword:

Example

Check if "apple" is present in the tuple:

thistuple = ("apple", "banana", "cherry")

if "apple" in thistuple:

print("Yes, 'apple' is in the fruits tuple")

output:

Yes, 'apple' is in the fruits tuple

Python - Update Tuples

Tuples are unchangeable, meaning that you cannot change, add, or remove items once the tuple is created.

But there are some workarounds.

Change Tuple Values

Once a tuple is created, you cannot change its values. Tuples are unchangeable, or immutable as it also is called.

But there is a workaround. You can convert the tuple into a list, change the list, and convert the list back into a tuple.

Example

Convert the tuple into a list to be able to change it:

x = ("apple", "banana", "cherry")

y = list(x)

y[1] = "kiwi"

x = tuple(y)

print(x)

output:

('apple', 'kiwi', 'cherry')

Add Items

Once a tuple is created, you cannot add items to it.

Example

You cannot add items to a tuple:

thistuple = ("apple", "banana", "cherry")

thistuple.append("orange") # This will raise an error

print(thistuple)

output:

Traceback (most recent call last):

File "D:/DEEPIKA/as.py", line 2, in <module>

thistuple.append("orange") # This will raise an error

AttributeError: 'tuple' object has no attribute 'append'

Just like the workaround for changing a tuple, you can convert it into a list, add your item(s), and convert it back into a tuple.

Example

Convert the tuple into a list, add "orange", and convert it back into a tuple:

thistuple = ("apple", "banana", "cherry")

y = list(thistuple)

y.append("orange")

thistuple = tuple(y)

Tuple operations:

Transversing a tuple:

Accessing and processing each element on it

for <item> in <Tuple>:

process each item here

for eg:

T=(“p”,”y”,”t”,”h”,”o”,”n”)

for a in T:

Print(T[a])

Output:

p

y

t

h

o

n

Joining tuples:

+operator it will join two tuples

Tuple1=(1,5,6)

Tuple2=(8,9,3)

Tuple1+Tuple2

Output:

(1, 5, 6, 8, 9, 3)

Repeating or Replicating Tuples:

Tuple1=(4,6,7)

Tuple1\*3

Output:

(4,6,7,4,6,7,4,6,7)

Slicing the tuples:

Seq=T[start:stop]

Tup1=(10,45,78,65,43,32,78,89)

Seq=tp1[4:-3]

Output:

(43,32)

Seq=T[start:stop:step]

Tp1=(45,32,67,21,34,89,90,43)

Tp1[0:10:2]

Output:

(45,67,34,90)

Tp1=(45,32,67,21,34,89,90,43,55)

Tp1[2:10:3]

Output:

(67,89,55)

Tuple functions and methods

Len()

Max()

Min()

Index()

Count() All these functions are similar to list.