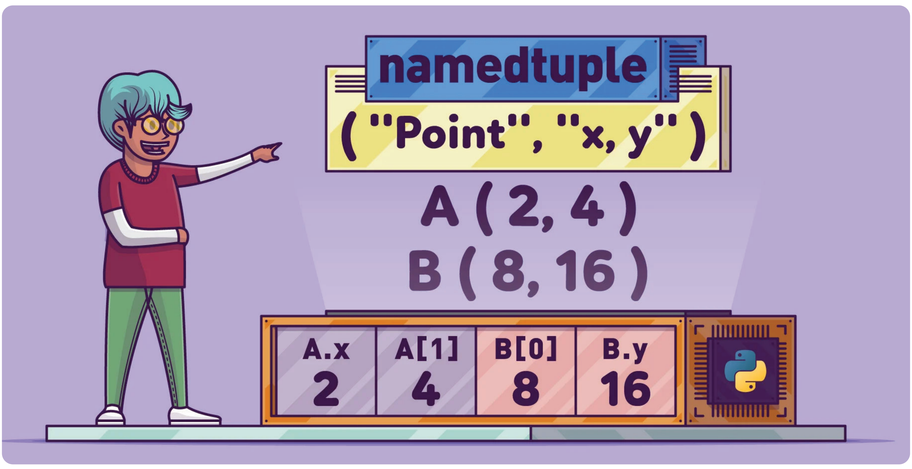
Day 5

Python for Data Science

Tuples




```
t1 = (22,45,67,89)
t2 = (1,3)
t3 = t1+t2
t3*2
```

(22, 45, 67, 89, 1, 3, 22, 45, 67, 89, 1, 3)

Tuples are immutable lists and cannot be changed in any way once it is created.

- Tuples are defined in the same way as lists.
- They are enclosed within parenthesis and not within square braces.
- Tuples are ordered, indexed collections of data.
- Similar to string indices, the first value in the tuple will have the index [0], the second value [1]
- Negative indices are counted from the end of the tuple, just like lists.
- Tuple also has the same structure where commas separate the values.
- Tuples can store duplicate values.
- Tuples allow you to store several data items including string, integer, float in one variable.

```
# Take a tuple
tuple_1 = ('Hello', 'Python', 3.14, 1.618, True, False, 32,
[1,2,3], {1,2,3}, {'A': 3, 'B': 8}, (0, 1))
tuple_1
```

('Hello',
'Python',
3.14,
1.618,
True,
False,
32,
[1, 2, 3],
{1, 2, 3},
{'A': 3, 'B': 8},
(0, 1))

```
print(type(tuple_1))
print(len(tuple_1))
```

<class 'tuple'>
11

Indexing

```
# Printi ng the each value in a tuple using both positi ve and
negati ve indexing
tuple_1 = ('Hello', 'Python', 3.14, 1.618, True, False, 32,
[1,2,3], {1,2,3}, {'A': 3, 'B': 8}, (0, 1))
print(tuple_1[0])
print(tuple_1[1])
print(tuple_1[2])
print(tuple_1[-1])
print(tuple_1[-2])
print(tuple_1[-3])
```

```
Hello
Python
3.14
(0, 1)
{'A': 3, 'B': 8}
{1, 2, 3}
```

```
# Printing the type of each value in the tuple
tuple_1 = ('Hello', 'Python', 3.14, 1.618, True, False, 32,
[1,2,3], {1,2,3}, {'A': 3, 'B': 8}, (0, 1))
print(type(tuple_1[0]))
print(type(tuple_1[2]))
print(type(tuple_1[4]))
print(type(tuple_1[6]))
print(type(tuple_1[7]))
print(type(tuple_1[8]))
print(type(tuple_1[9]))
print(type(tuple_1[10]))
```

```
<class 'str'>
<class 'float'>
<class 'bool'>
<class 'int'>
<class 'list'>
<class 'set'>
<class 'dict'>
<class 'tuple'>
```

Concatenation of tuples

To concatenate tuples, + sign is used

```
tuple_2 = tuple_1 + ('Hello World!', 2022)
tuple_2
```

```
('Hello',
'Python',
3.14,
1.618,
True,
False,
32,
[1, 2, 3],
{1, 2, 3},
{'A': 3, 'B': 8},
(0, 1),
'Hello World!',
2022)
```

Repetition of a tuple

```
rep_tup = (1,2,3,4)
rep_tup*2
```

```
(1, 2, 3, 4, 1, 2, 3, 4)
```

Membership

```
rep_tup = (1,2,3,4)
print(2 in rep_tup)
print(2 not in rep_tup)
print(5 in rep_tup)
print(5 not in rep_tup)
```

True
False
False
True

Iteration

```
rep_tup = (1,2,3,4)
for i in rep_tup:
  print(i)
```

1
2
3
4

Slicing

To obtain a new tuple from the current tuple, the slicing method is used.

```
# Obtaining a new tuple from the index 2 to index 6
tuple_1 = ('Hello', 'Python', 3.14, 1.618, True, False, 32,
[1,2,3], {1,2,3}, {'A': 3, 'B': 8}, (0, 1))
tuple_1[2:7]
```

(3.14, 1.618, True, False, 32)

```
# Obtaining tuple using negati ve indexing
tuple_1 = ('Hello', 'Python', 3.14, 1.618, True, False, 32,
[1,2,3], {1,2,3}, {'A': 3, 'B': 8}, (0, 1))
tuple_1[-4:-1]
```

([1, 2, 3], {1, 2, 3}, {'A': 3, 'B': 8})

Different Tuple operation

min() function

```
rep_tup = (1,2,3,4)
min(rep_tup)
```

1

min() function

```
ep_tup = (1,2,3,4)
max(rep_tup)
```

4

tup(seq) function

It converts a specific sequence to a tuple

```
seq = 'ATGCGTATTGCCAT'
tuple(seq)
```

('A', 'T', 'G', 'C', 'G', 'T', 'A', 'T', 'T', 'G', 'C', 'C', 'A', 'T')

len() function

To obtain how many elements there are in the tuple, use len() function.

```
tuple_1 = ('Hello', 'Python', 3.14, 1.618, True, False, 32,
[1,2,3], {1,2,3}, {'A': 3, 'B': 8}, (0, 1))
len(tuple_1)
```

11

Sorting tuple

```
# Tuples can be sorted and save as a new tuple.
tuple_3 = (0,9,7,4,6,2,9,8,3,1)
sorted_tuple_3 = sorted(tuple_3)
sorted_tuple_3
```

[0, 1, 2, 3, 4, 6, 7, 8, 9, 9]

Nested tuple

In Python, a tuple written inside another tuple is known as a nested tuple.

```
# Take a nested tuple
nested_tuple =('biotechnology', (0, 5), ('fermentati on', 'etha-
nol'), (3.14, 'pi', (1.618, 'golden rati o')) )
nested_tuple
```

('biotechnology',
 (0, 5),
 ('fermentati on', 'ethanol'),
 (3.14, 'pi', (1.618, 'golden rati o')))

```
# Now printi ng the each element of the nested tuple
print('Item 0 of nested tuple is', nested_tuple[0])
print('Item 1 of nested tuple is', nested_tuple[1])
print('Item 2 of nested tuple is', nested_tuple[2])
print('Item 3 of nested tuple is', nested_tuple[3])
```

Item 0 of nested tuple is biotechnology
Item 1 of nested tuple is (0, 5)
Item 2 of nested tuple is ('fermentati on', 'ethanol')
Item 3 of nested tuple is (3.14, 'pi', (1.618, 'golden rati o'))

```
# Using second index to access other tuples in the nested tuple
print('Item 1, 0 of the nested tuple is', nested_tuple[1][0])
print('Item 1, 1 of the nested tuple is', nested_tuple[1][1])
print('Item 2, 0 of the nested tuple is', nested_tuple[2][0])
print('Item 2, 1 of the nested tuple is', nested_tuple[2][1])
print('Item 3, 0 of the nested tuple is', nested_tuple[3][0])
print('Item 3, 1 of the nested tuple is', nested_tuple[3][1])
print('Item 3, 2 of the nested tuple is', nested_tuple[3][2])
# Accesing to the items in the second nested tuples using a third
index
print('Item 3, 2, 0 of the nested tuple is', nested_tuple[3][2][0])
print('Item 3, 2, 1 of the nested tuple is', nested_tuple[3][2][1])
```

Item 1, 0 of the nested tuple is 0
Item 1, 1 of the nested tuple is 5
Item 2, 0 of the nested tuple is fermentati on
Item 2, 1 of the nested tuple is ethanol
Item 3, 0 of the nested tuple is 3.14
Item 3, 1 of the nested tuple is pi
Item 3, 2 of the nested tuple is (1.618, 'golden rati o')
Item 3, 2, 0 of the nested tuple is 1.618
Item 3, 2, 1 of the nested tuple is golden rati o

Tuples are immutable

```
# Take a tuple
tuple_4 = (1,3,5,7,8)
tuple_4[0] = 9
print(tuple_4)
# The output shows the tuple is immutable
```

TypeError Traceback (most recent call last)
Input In [21], in <cell line: 3>()
1 # Take a tuple
2 tuple_4 = (1,3,5,7,8)
---> 3 tuple_4[0] = 9
4 print(tuple_4)

TypeError: 'tuple' object does not support item assignment

Delete a tuple

An element in a tuple can not be deleted si nce it is immutable.
But a whole tuple can be deleted

```
tuple_4 = (1,3,5,7,8)
print('Before deleti ng:', tuple_4)
del tuple_4
print('Aft er deleti ng:', tuple_4)
```

Before deleti ng: (1, 3, 5, 7, 8)

NameError Traceback (most recent call last)
Input In [22], in <cell line: 4>()
2 print('Before deleti ng:', tuple_4)
3 del tuple_4
----> 4 print('Aft er deleti ng:', tuple_4)

NameError: name 'tuple_4' is not defined

count() method

This method returns the number of time an item occurs in a tuple.

```
tuple_5 = (1,1,3,3,5,5,5,5,6,6,7,8,9)
tuple_5.count(5)
```

4

index() method

It returns the index of the first occurrence of the specified value in a tuple

```
tuple_5 = (1,1,3,3,5,5,5,5,6,6,7,8,9)
print(tuple_5.index(5))
print(tuple_5.index(1))
print(tuple_5.index(9))
```

4
0
12

One element tuple

if a tuple includes only one element, you should put a comma after the element.
Otherwise, it is not considered as a tuple.

```
tuple_6 = (0)
print(tuple_6)
print(type(tuple_6))
# Here, you see that the output is an integer
```

0
<class 'int'>

```
tuple_7 = (0,)
print(tuple_7)
print(type(tuple_7))
# You see that the output is a tuple
```

(0,)
<class 'tuple'>