Programing in Python

Lecture 7a - Tuples

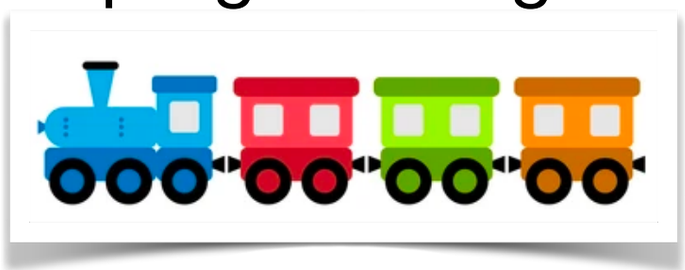
Instructor: Zhandos Yessenbayev

Outline

- Python Collections
- Python Tuples
- Access Tuple Items
- Updating a Tuple
- Unpacking a Tuple
- Tuple Methods
- Looping through Tuple

Python Collections

- There are **four** collection data types in the Python programming language:



  - **List** is a collection which is *ordered* and *changeable* (with *duplicates*).

  - **Tuple** is a collection which is *ordered* and *unchangeable* (with *duplicates*).

  - **Set** is a collection which is *unordered*, *unchangeable* (with *no duplicates*).

  - **Dictionary** is a collection which is *ordered* and *changeable* (with *no duplicates*).

Python Tuples

- **Tuple** is a collection which is *ordered* and *unchangeable* (with *duplicates*).
- We can initialize a tuple with some items
- Single item tuples use comma **(item,)**
- Items (elements) can be of any type
- *Empty* tuple can be set with **( )** or **tuple( )** function
- *Length* of the tuple can be found using **len( )** function

```
>>> mytuple = ( 25, 'hello', 'hello', [100, 200] )
>>> len(mytuple)
4
>>> single_item = (10,)
>>> empty_tuple1 = ()
>>> empty_tuple2 = tuple()
```

Access Tuple Items

```
>>> my_tuple = (‘C', 'H', ‘Tom’, 2)
>>>
>>> len(my_tuple)
>>> 4
>>>
>>> my_tuple[0]
‘C’
>>>
>>> mylist[-1]
‘Tom'
>>>
>>> my_tuple[1:-1]
(‘H’, ‘Tom’)
>>>
```

This is a tuple !!!

Updating a Tuple

- Tuples are **unchangeable !!!**

- We cannot change, add, or remove items once the tuple is created.

```
>>> my_tuple = (‘C', 'H', ‘Tom’)
>>>
>>> my_tuple[0] = ‘A’
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
>>>
>>> my_list = list( my_tuple )
>>> my_list[0] = 'A'
>>> my_list
['C', 'H', 'Tom']
>>> my_tuple = tuple( my_list )
>>> my_tuple
('A', 'H', ‘Tom')
```

Convert to list

Convert back to tuple

Unpacking a Tuple

- We can assign values to a tuple - **packing** a tuple
- We can extract the values back into variables - **unpacking** a tuple

```
>>> # Packing a tuple:
>>> fruits = ("apple", "banana", "cherry")
>>>
>>> # Unpacking a tuple:
>>> (green, yellow, red) = fruits
>>> print(green, yellow, red)
>>>
>>> (green, yellow) = fruits
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: too many values to unpack (expected 2)
```

Unpacking a Tuple

- We can unpack lists, function's return values, etc.

```
>>> m = [ 'have', 'fun' ]
>>> (x, y) = m
>>>
>>> addr = 'bob@gmail.com'
>>> username, domain = addr.split('@')
>>>
>>> a, b = b, a
>>>
>>> def func(x, y):
...   return 2*x, x+y
...
>>> s, t = func(2, 3)
>>> s
4
>>> t
5
```

List Methods

- Python has built-in methods to use on lists.

Method	Description
count()	Returns the number of elements with the specified value
index()	Returns the index of the first element with the specified value

```
>>> tup = (1, 2, 2)
>>> dir(tup)
>>>
>>> tup.count(2)
2
>>> tup.index(0)
0
>>> tup.index(2)
1
```

Source: https://www.w3schools.com/python/python_tuples_methods.asp

Looping through Tuple

- We use **for** to loop through the tuple.
- There are 2 use cases when you need:
  - access by value
  - access by index

```
>>> tup = (2, 4, 6, 8, 10, 12, 14)
>>>
>>> # Case 1
>>> for item in tup:
...     print(item)
...
>>>
>>> # Case 2
>>> n = len(tup)
>>> for i in range(n):
...     print(int(tup[i] / 2))
...
```

Unique Words

- *Exercise:* **Print the lines of a file in increasing order of their lengths.**

    1. Create a new file *input.txt*

    2. Fill the file with some text

    3. Open the file from a program

    4. Read the file line by line

    5. Find the length of each line

    6. Populate a list of tuples — (length, line)

    7. Print the list of tuples in increasing order of their length

Thanks!