Maktab Sharif

Python | Main course

Session 4

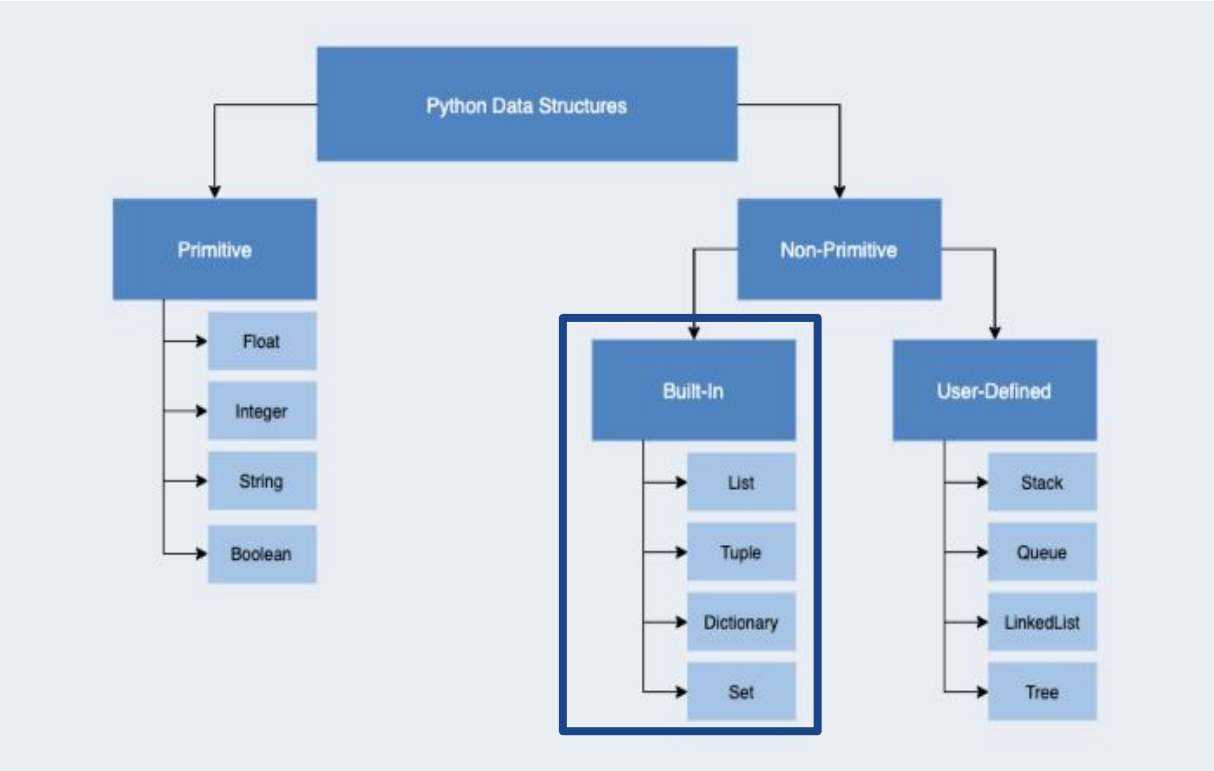
Python data structures
Dictionary
Tuple
(Short-hands)

by Mohammad Amin H.B. Tehrani - Reza Yazdani

www.maktabsharif.ir

Python data structure

Maktab Sharif



Python has **primitive** (or basic) data structures such as floats, integers, strings, and Booleans. Python also has **non-primitive** data structures such as lists, tuples, dictionaries, and sets. Non-primitive data structures store a collection of values in various formats rather than a single value. Some can hold data structures within the data structure, creating depth and complexity in data storage capabilities.

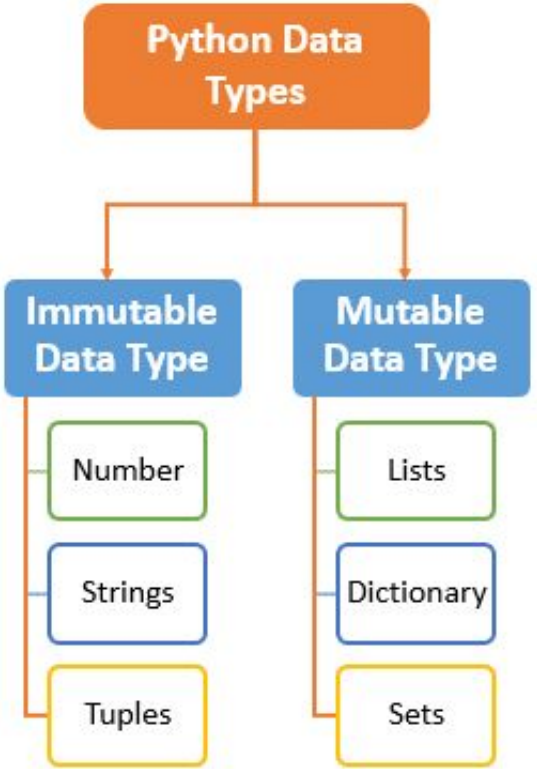
Maktab Sharif

Mutability

Maktab Sharif

Mutability means that the data in the data structure can be modified (added, deleted, or changed) after creation. Mutability is an important factor to consider when choosing your data structure. If you know that you won't need to change the internal state, consider using an immutable object to ensure that it is thread-safe and that nothing can overwrite your data.

```
l = [1, 2, 3]   # List is MUTABLE!
t = (1, 2, 3)   # Tuple is IMMUTABLE!

l[3] = 'New item'
t[3] = 'New item'  # ???
```



Chapter 11

Dictionary

Maktab Sharif

Dictionary (dict)

Maktab Sharif

Dictionaries are used to store data values in KEY:VALUE pairs.

A dictionary is a collection which is **ordered***, **changeable** and **does not allow duplicates**.
As of Python version 3.7, dictionaries are *ordered*. In Python 3.6 and earlier, dictionaries are *unordered*.

Syntax:

`my_dict = {key1:value1, key2:value2, key3:value3 , … }`

```
my_dict = {
    'a_str_key': 'Anything can be a value in python dicts...',
    85: 32,
    True: 15.67,
    (1, 2, 3): 1,
    [1, 2, 3]: 2,  #???
}
print(my_dict)
```

Access to Dictionary Items

Maktab Sharif

1) Use access operator: ur_dict[key]

Dictionary items are presented in key:value pairs, and can be referred to by using the key name.

2) Use .get() method: ur_dict.get(key, ...)

You can call .get() method with a default value. if key not found in the dict, default value will return.

```
print(my_dict[85])
print(my_dict['a_str_key'])
print(my_dict[True])
print(my_dict[(1, 2, 3)])
print(my_dict['Name'])

print(my_dict.get(85))
print(my_dict.get('a_str_key'))
print(my_dict.get('Name', 'Akbar'))
```

Some Dictionary methods

Maktab Sharif

clear()	Removes all the elements from the dictionary
copy()	Returns a copy of the dictionary
fromkeys()	Returns a dictionary with the specified keys and value
get()	Returns the value of the specified key
items()	Returns a list containing a tuple for each key value pair
keys()	Returns a list containing the dictionary's keys
pop()	Removes the element with the specified key
popitem()	Removes the last inserted key-value pair
update()	Updates the dictionary with the specified key-value pairs
values()	Returns a list of all the values in the dictionary

Dictionary Example

Maktab Sharif

```
>>> tel = {'jack': 4098, 'sape': 4139}
>>> tel['guido'] = 4127
>>> tel                          # 1?
>>> tel['jack']                  # 2?
>>> del tel['sape']
>>> tel['irv'] = 4127
>>> tel                          # 3?
>>> list(tel)                    # 4?
>>> sorted(tel)                  # 5?
>>> 'guido' in tel               # 6?
>>> 'jack' not in tel            # 7?
```

Dictionary Example

Maktab Sharif

```
>>> tel = {'jack': 4098, 'sape': 4139}
>>> tel['guido'] = 4127
>>> tel
{'jack': 4098, 'sape': 4139, 'guido': 4127}
>>> tel['jack']
4098
>>> del tel['sape']
>>> tel['irv'] = 4127
>>> tel
{'jack': 4098, 'guido': 4127, 'irv': 4127}
>>> list(tel)
['jack', 'guido', 'irv']
>>> sorted(tel)
['guido', 'irv', 'jack']
>>> 'guido' in tel
True
>>> 'jack' not in tel
 False
```

Example: *args , **kwargs

Maktab Sharif

What's result of code below

```
def save_information(first_name, last_name, phone, *marks, **extra_info):
    print(first_name)
    print(last_name)
    print(phone)
    print(type(marks), marks)
    print(type(extra_info), extra_info)
    ...


save_information('Reza',
                 'Bahadori',
                 '099999999999',
                 20, 12, 3, 5,
                 email='reza@bahadori...', username='R.BAHADOR'
                 )
```

Chapter 12

Tuples

Maktab Sharif

Tuples

Maktab Sharif

Tuples are used to store multiple items in a single variable.

Tuple is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Set, and Dictionary, all with different qualities and usage.

A tuple is a collection which is ordered and **immutable (unchangeable)**.

Tuples are written with **round brackets.**

```
my_tuple = ('akbar', 'asqar', 'reza')

print(my_tuple)
```

Tuple is:

Maktab Sharif

- **Ordered**

  When we say that tuples are ordered, it means that the items have a defined order, and that order will not change.

- **Immutable**

  Tuples are unchangeable, meaning that we cannot change, add or remove items after the tuple has been created.

- **Allow duplicates**

  Since tuples are indexed, they can have items with the same value:

```
my_tuple = ('akbar', 'akbar', 'akbar', 'reza')
```

Tuple example

Maktab Sharif

```
>>> t = 12345, 54321, 'hello!'
>>> t[0]                # 1?
>>> t                   # 2?
>>> u = t, (1, 2, 3, 4, 5)
>>> u                   # 3?
>>> t[0] = 88888        # 4?
>>> v = ([1, 2, 3], [3, 2, 1])
>>> v                   # 5?
```

Tuple example

Maktab Sharif

```
>>> t = 12345, 54321, 'hello!'
>>> t[0]
12345
>>> t
(12345, 54321, 'hello!')
>>> # Tuples may be nested:
... u = t, (1, 2, 3, 4, 5)
>>> u
((12345, 54321, 'hello!'), (1, 2, 3, 4, 5))
>>> # Tuples are immutable:
... t[0] = 88888
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
>>> # but they can contain mutable objects:
... v = ([1, 2, 3], [3, 2, 1])
>>> v
 ([1, 2, 3], [3, 2, 1])
```

Chapter 11

(Short-Hands)

Maktab Sharif

Conditional expression (ternary operator)

Maktab Sharif

Ternary operators also known as conditional expressions are operators that evaluate something based on a condition being true or false.

```
variable = first_value if condition else second_value
```

```
a, b = 10, 20

minimum = a if a < b else b

print(minimum)
```

```
a, b = 10, 20

if a < b:
    minimum = a
else:
    minimum = b

print(minimum)
```

List Comprehension (inline for)

Maktab Sharif

List comprehension offers a shorter syntax when you want to create a new list based on the values of an existing list.

```
newlist = [expression for item in iterable]
newlist = [expression for item in iterable if condition == True]
```

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

newlist = [x for x in fruits if "a" in x]

print(newlist)
```

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]
newlist = []

for x in fruits:
  if "a" in x:
    newlist.append(x)

print(newlist)
```

List Comprehension Examples

Maktab Sharif

```
l = [x**2 for x in range(1, 21)]
print(l)
```

```
s = 'Akbar neshan'
l = [x.upper() for x in s if x.lower() in 'aoieu']
print(l)
```

```
n = 10

for i in range(1, n+1):
    print(*[i*j for j in range(1, n+1)], sep='\t')
```

Lambda (inline function)

Maktab Sharif

A lambda function is a small anonymous function.
A lambda function can take any number of arguments, but can only have one expression.

`lambda` *`arguments`* : *`expression`*

```
f = lambda x: 2*x + 10
print(f(2))
```

```
def f(x):
    return 2*x + 10

print(f(2))
```

Lambda Examples

Maktab Sharif

```
f = lambda a, b, c : a + b + c
print(f(5, 6, 2))
```

```
l = list(map(lambda x: x ** 2, range(1, 21)))
print(l)
```

```
n = int(input('Enter a number: '))

x = lambda n: not [i for i in range(2, n) if not (n % i)]
y = lambda N: [i for i in range(2, N+1) if x(i)]

print(y(n))
```

Examples

Example 1

Maktab Sharif

What's result of code below

```
def is_primal(n):
    n = int(n)
    assert n > 0
    for i in range(2, int(n**0.5) +1):
        if not (n%i):
            return False
    return True

l = list(map(lambda x:is_primal(x), range(2, 100)))
print(any(l), all(l))
```

Example 1

Maktab Sharif

What's result of code below

```
def is_primal(n):
    n = int(n)
    assert n > 0
    for i in range(2, int(n**0.5) +1):
        if not (n%i):
            return False
    return True

l = list(map(lambda x:is_primal(x), range(2, 100)))
print(any(l), all(l))
```

```
True False
```

Example 2

Maktab Sharif

What's result of code below

```
s1 = list(range(10))
s2 = "Hello Ali!"
x1 = zip(s1, s2)
x2 = enumerate(s2)
print(list(x1) == list(x2))
```

Example 2

Maktab Sharif

What's result of code below

```
s1 = list(range(10))
s2 = "Hello Ali!"
x1 = zip(s1, s2)
x2 = enumerate(s2)
print(list(x1) == list(x2))
```

```
True
```

Example 3

Maktab Sharif

What's result of code below

```
s = "1 - Hello World /n2-  Hello Akbar +" \
    "3 Hello Reza"
x = list(filter(lambda x: not x.isalpha(), s))
print(x)
x = list(reversed(x))
print(x)
x = [str(ord(_)) if _.isnumeric() else _ for _ in x if _ != ' ']
print(x)
x = ''.join(x)
print(x)
x = eval(x)
print(x)
x = round(x, 3)
print(x)
x = hex(int(x * 10))
print(x)
```

Example 3

Maktab Sharif

Result:

```
['1', ' ', '-', ' ', ' ', ' ', '/', '2', '-', ' ', ' ', ' ', ' ', '+', '3', ' ', ' ']
[' ', ' ', '3', '+', ' ', ' ', ' ', ' ', '-', '2', '/', ' ', ' ', ' ', '-', ' ', '1']
['51', '+', '-', '50', '/', '-', '49']
51+-50/-49
52.02040816326531
52.02
0x208
```

Exercises

Exercise: Tuple-Dict convertor

Maktab Sharif

Tuple-Dict convertor

Write a **function (tuple_dict_converter)** that, gets an argument then performs the following action on that:

A. Convert it to **a list of Tuples** if it is a **Dict**
B. Convert it to **Dict** if it is **a list of Tuples**

Hint: Read about *'isinstance()`* function

Note*: You are NOT allowed to use built-in functions. (Except *`isinstance*)

Example 1:

```
a_dict = {'a': 1, 'b': 2}

print(tuple_dict_converter(a_dict))
# [('a', 1), ('b', 2)]
```

Example 2:

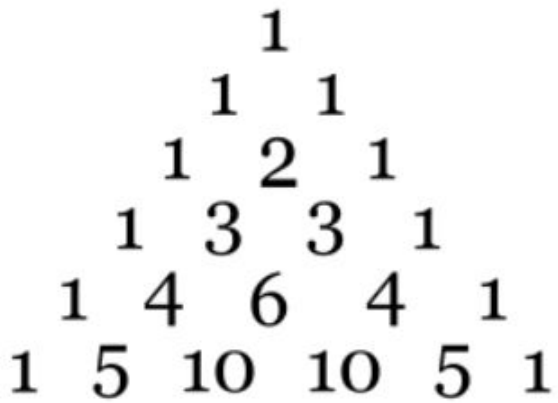
```
a_l_of_tuples = [('a', 1), ('b', 2)]

print(tuple_dict_converter(a_l_of_tuples))
# {'a': 1, 'b': 2}
```

Exercise: Pascal's Triangle

Maktab Sharif

برنامه‌ای بنویسید که عدد n را از ورودی بگیرد و n سطر اول مثلث خیام پاسکال را نمایش دهد.

مثلث خیام پاسکال به این صورت است که ابتدا تنها یک عدد ۱ در سطر اول وجود دارد. سپس در سطر i، i عدد وجود دارد که عدد اول و آخر آن ۱، و هر کدام از اعداد دیگر جمع دو عدد بالایی خود می‌باشند. تصویر زیر، شش سطر اول مثلث خیام پاسکال می‌باشد.



input:

```
5
```

output:

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
```

Pre-reading

Search about:

1. * List copy in python
2. * Dict comprehension in python
3. * Set in python (Set vs. list)
4. * args and kwargs in python (*args & **kwargs)

Maktab Sharif