Python Data Type (Advanced)

Seaborn Kusto


HI, FRIENDS!
WE ARE
SEABORN KUSTO
TEAM

Official Members

Malka Rusyd Abdussalam

Project Leader
(Github Code)

Trianto Haryo Nugroho

Member
(Content Creator)

Maulana Ishaq Siregar

Member
(Presentation)

Fauzi Wardah Ali

Member
(Github Readme)

Rizki Afrinal

Member
(Linkedin Caption)

OUTLINES

01
LIST

02
TUPLE

03
SET

04
DICTIONARY

01. LIST

The list is a data type that contains ordered items. As with the string data type, each item (members) list have indexes according to the order. The index starts from 0 and not from 1. Lists can contain members of the same or different types. Brackets [ ] are used to declare a list, and a comma separates each member.

The method to access items from a list calls the list's name followed by the item's index in question, namely with the format name list [index]. In addition, access to a number of items from the index to the index can also be done. This matter
called slicing.

EXAMPLE

script.py IPython Shell

```
1   a = [5,10,15,20,25,30,35,40]
2
3   # a[2] = 15
4   print("a[2] = ", a[2])
5
6   # a[0:3] = [5, 10, 15]
7   print("a[0:3] = ", a[0:3])
8
9   # a[5:] = [30, 35, 40]
10  print("a[5:] = ", a[5:])
```

Creating List

```
# list kosong
my_list = []

# list berisi integer
my_list = [1,2,3,4,5]

# list berisi tipe campuran
my_list = [1, 3.5, "Hello"]
```

```
# list bersarang
my_list = ["hello", [2,4,6], ['a','b']]
```

Accessing Members List

script.py | IPython Shell

```
1   my_list = ["I", "love","python","programming",2017]
2   # output: I
3   my_list[0]
4
5   #output: python
6   my_list[2]
7
8   # list dalam list
9   your_list = ["hello", [1,2,3], "python"]
10
11  # output 1
12  print(your_list[1][0])
13
14  # output 3
15  print(your_list[1][2])
16
17  # IndexError
18  my_list[6]
```

List with Negative Index

script.py IPython Shell

```
1  my_list = ['p','y','t','h','o','n']
2
3  # output: n
4  my_list[-1]
5
6  # output: h
7  my_list[-3]
```

Slicing List

script.py IPython Shell

```
1  my_list = ['p','y','t','h','o','n','i','n','d','o']
2
3  # anggota list dari 3 s/d 5 (dari h s/d n)
4  print(my_list[3:6])
5
6  # anggota list dari 4 s/d yang terakhir
7  print(my_list[4:])
8
9  # anggota list dari 0 s/d 4
10 print(my_list[:5])
11
12 # indeks dari belakang dari -1 s/d -4
13 print(my_list[-1:-5])
```

Changing List Members

script.py IPython Shell

```
1
2   # misal ada nilai yang salah
3   ganjil = [1,3,4,7,9]
4
5   # ubah item ke 3 (indeks ke 2)
6   ganjil[2] = 5 print(ganjil)
7
8   # mengubah sekali banyak
9   ganjil[2:5] = [11,13,15]
10  print(ganjil)
```

Add Member List

```
>>> ganjil = [1,3,5,7]

>>> ganjil.append(9)
>>> print(ganjil)
[1,3,5,7,9]

>>> ganjil.extend([11,13,15])
>>> print(ganjil)
[1,3,5,7,9,11,13,15]
```

```
>>> genap = [2, 4, 6]
>>> print(genap + [8, 10, 12])
[2, 4, 6, 8, 10, 12]

>>> print(['p','y'] * 2)
['p','y','p','y]
```

Insert List Members

```
>>> ganjil = [5,7,11,13,15]

>>> # kita akan menyisipkan 9 setelah angka 7
>>> ganjil.insert(2,9)
>>> print(ganjil)
[5,7,9,11,13,15]
```

Deleting List Members

script.py | IPython Shell

```
1   my_list = ['p', 'y', 't', 'h', 'o', 'n', 'i', 'n', 'd', 'o']
2   my_list.remove('p')
3
4   # output ['y', 't', 'h', 'o', 'n', 'i', 'n', 'd', 'o']
5   print(my_list)
6
7   my_list.remove('n')
8   # remove hanya menghapus elemen pertama yang dijumpai
9   # output: ['p', 'y', 't', 'h', 'o', 'i', 'n', 'd', 'o']
10
11  # Output 'y'
12  print(my_list.pop(1))
13
14  del my_list[2]
15  print(my_list)
16
17  my_list.clear()
18  # Output []
19  print(my_list)
```

Sort List Members

```
>>> alfabet = ['a','b','d','f','e','c','h','g','j','i']
>>> alfabet.sort()
>>> print(alfabet)
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

>>> alfabet.sort(reverse=True)
>>> print(alfabet)
['j', 'i', 'h', 'g', 'f', 'e', 'd', 'c', 'b', 'a']
```

Reverse List Order

```
>>> alfabet = ['a','c','d','e','b']
>>> alfabet.reverse()
>>> print(alfabet)
['b','e','d','c','b','a']
```

02. TUPLE

Tuples are another data type similar to lists. The difference with a list is its members Cannot be changed (immutable). Lists are mutable, while tuples are immutable. When a tuple is created, its contents cannot be modified anymore. Tuples are declared using brackets ( ). Their members are separated by commas.

Tuples are useful for data that is meant to be unaltered. For example, tuple the colour composition for white is (255,255,255). Like lists, we can access tuple members by using their index.

EXAMPLE

script.py IPython Shell

```
1    white = (255,255, 255)
2    red = (255,0,0)
3    print(white)
4    print(red[0])
5    print(red[1])
6
7    # akan menghasilkan error
8    # tuple bersifat immutable
9    red[0] = 128
```

Creating Tuple

script.py IPython Shell

```
# membuat tuple kosong
# Output: ( )
my_tuple = ()
print(my_tuple)

# tuple dengan 1 elemen
# Output: (1,)
my_tuple = (1,)
print (my_tuple)

# tuple berisi integer
# output = (1, 2, 3)
my_tuple = (1, 2, 3)
print(my_tuple)
# tuple bersarang
# Output: ("hello", [1, 2, 3], (4, 5, 6))
my_tuple = ("hello", [1, 2, 3], (4, 5, 6))
print(my_tuple)

# Tuple bisa tidak menggunakan tanda ()
# Output (1, 2, 3)
my_tuple = 1, 2, 3

# memasukkan anggota tuple ke variabel yang bersesuaian
# a akan berisi 1, b berisi 2, dan c berisi 3
# output 1 2 3
a, b, c = my_tuple
print(a, b, c)
```

Accessing Tuple Members

script.py IPython Shell

```
my_tuple = ('p','y','t','h','o','n')
# Output: 'p'
print(my_tuple[0])

# Output: 'y'
print(my_tuple[1])

# Output: 'n'
print(my_tuple[-1])

# Output: 'o'
print(my_tuple[-2])

# IndexError
print(my_tuple[6])
```

script.py IPython Shell

```
my_tuple = ('p','r','o','g','r','a','m','m','i','n','g')
# akses dari indeks 0 s/d 2

# output: ('p','r','o')
print(my_tuple[:3])

# Akses dari indeks 2 s/d 5
# output: ('r','o','g','r')
print(my_tuple[2:6])

# Akses dari indeks 3 sampai akhir
# output: ('r','o','g','r','a','m','m','i','n','g')
print(my_tuple[3:])
```

Changing Tuple Members

script.py IPython Shell

```
1   my_tuple = (2, 3, 4, [5, 6])
2   # kita tidak bisa mengubah anggota tuple
3   # bila kita hilangkan tanda komentar # pada baris ke 6
4   # akan muncul error: # TypeError: 'tuple' object does not support item
    assignment
5
6   # my_tuple[1] = 8
7
8   # tapi list di dalam tuple bisa diubah
9   # output: (2, 3, 4, [7, 6])
10  my_tuple[3][0] = 7
11  print(my_tuple)
12
13  # tuple bisa diganti secara keseluruhan dengan penugasan kembali
14  # output: ('p','y','t','h','o','n')
15  my_tuple = ('p','y','t','h','o','n')
16  print(my_tuple)
17
18  # anggota tuple juga tidak bisa dihapus menggunakan del
19  # perintah berikut akan menghasilkan error TypeError
20  # kalau Anda menghilangkan tanda komentar #
21
22  #del my_tuple[0]
23
24  # kita bisa menghapus tuple keseluruhan
25
26  del my_tuple
```

Counting and Indexing Tuple's Members

script.py IPython Shell

```
1  my_tuple = ('p','y','t','o','n','i','n','d','o')
2  # count
3  # output: 2
4  print(my_tuple.count('n'))
5
6  # index
7  # Output 4
8  print(my_tuple.index('n'))
```

03. SET

Set is an unordered data type in Python. Set has members unique (no duplication). So, if we put two equal members in a set, the set will automatically eliminate one of them. Sets can be used to perform mathematical set operations such as joins, slices, difference, and complement.

A set is created by placing its members inside curly braces { }, separated by commas. We can also create sets from lists by entering the list into the set () function. Sets can contain mixed data, both integers, floats, strings, etc. However, a set cannot contain lists, sets, and dictionaries.

EXAMPLE

script.py | IPython Shell

```python
# set integer
my_set = {1,2,3}
print(my_set)

# set dengan menggunakan fungsi set()
my_set = set([1,2,3])
print(my_set)

# set data campuran
my_set = {1, 2.0, "Python", (3,4,5)}
print(my_set)

# bila kita mengisi duplikasi, set akan menghilangkan salah satu
# output: {1,2,3}
```

Making Set

script.py | IPython Shell

```
1   # set integer
2   my_set = {1,2,3}
3   print(my_set)
4
5   # set dengan menggunakan fungsi set()
6   my_set = set([1,2,3])
7   print(my_set)
8
9   # set data campuran
10  my_set = {1, 2.0, "Python", (3,4,5)}
11  print(my_set)
12
13  # bila kita mengisi duplikasi, set akan menghilangkan salah satu
14  # output: {1,2,3}
15  my_set = {1,2,2,3,3,3}
16  print(my_set)
17
18  # set tidak bisa berisi anggota list
19  # contoh berikut akan muncul error TypeError
20  my_set = {1,2,[3,4,5]}
```

Making Empty Set

```
>>> # membuat variabel a dengan {}
>>> a = {}
>>> print(type(a))
<class 'dict'>

>>> # harus menggunakan fungsi set()
>>> a = set()
>>> print(type(a))
<class 'set'>
```

Changing Set Members

script.py IPython Shell

```
1   # membuat set baru
2   my_set = {1,2,3}
3   print(my_set)
4
5   # bila kita hilangkan tanda # dari baris 9 akan muncul error TypeError

6   #my_set[0]
7
8   # menambah satu anggota
9   # output: {1,2,3,4}
10  my_set.add(4)
11  print(my_set)
12
13  # menambah beberapa anggota
14  # set akan menghilangkan duplikasi
15  # output: {1,2,3,4,5,6}
16  my_set.update([3,4,5,6])
17  print(my_set)
```

Deleting Set Members

script.py IPython Shell

```
1   # membuat set baru
2   my_set = {1, 2, 3, 4, 5}
3   print(my_set)
4
5   # menghapus 4 dengan discard
6   # output: {1, 2, 3, 5}
7   my_set.discard(4)
8   print(my_set)
9
10  # menghapus 5 dengan remove
11  # output : {1, 2, 3}
12  my_set.remove(5)
13  print(my_set)
14
15  # anggota yang mau dihapus tidak ada dalam set
16  # discard tidak akan memunculkan error
17  # output: {1, 2, 3}
18  my_set.discard(6)
```

Deleting All Set Members

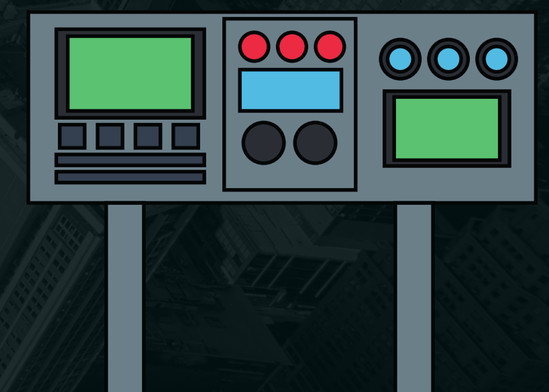
script.py IPython Shell

```python
# membuat set baru
# output: set berisi anggota yang unik
my_set = set("HelloPython")
print(my_set)

# pop anggota
# output: anggota acak
print(my_set.pop())

# pop anggota lainnya
# output: anggota acak
print(my_set.pop())

# mengosongkan set
# output: set()
my_set.clear()
print(my_set)
```

04. DICTIONARY

Dictionary is a data type in which each member consists of key -value pairs (key-value). Similar to a dictionary where there are words that have meaning. Dictionaries are generally used for large data and to randomly access data members.

Dictionary members do not have indexes. Dictionaries are declared using curly braces { }, where the members have the form key: value or key: value and each member is separated by a comma. Key and value can have any type. To access the value of a dictionary member, we use the key.



EXAMPLE

script.py IPython Shell

```
1  d = {1:'satu', 2:'dua', 'tiga':3}
2  print(tipe(d))
3  print("d[1] = ", d[1])
4  print("d['tiga'] = ", d['tiga'])
5  # Error
6  print("d[3] = ", d[3])
```

Create a Dictionary

```
# Membuat dictionary kosong
my_dict = {}

# dictionary dengan kunci integer
my_dict = {1: 'sepatu', 2: 'tas'}

# dictionary dengan kunci campuran
my_dict = {'warna': 'merah', 1: [2,3,5]}

# membuat dictionary menggunakan fungsi dict()
my_dict = dict([(1:'sepatu'), (2:'bola')])
```

Accessing Member Dictionary

script.py IPython Shell

```
1   my_dict = {'nama':'Budi', 'usia':27}
2
3   # output: Budi
4   print(my_dict['nama'])
5
6   # output 27
7   print(my_dict.get('usia'))
8   # Mengakses kunci yang tidak tersedia menyebabkan KeyError
9   my_dict['alamat']
```

Changing Dictionary Member

script.py IPython Shell

```python
my_dict = {'nama':'Gani', 'usia':35}

# mengupdate nilai
my_dict['usia'] = 36
# Output: {'nama': 'Gani', 'usia':36}
print(my_dict)

# menambah anggota
my_dict['alamat'] = 'Medan'
# output: {'alamat': 'Medan', 'nama':'Gani', 'usia':36}
print(my_dict)
```

Deleting Dictionary Members

script.py IPython Shell

```python
# membuat dictionary baru
pangkat = {1:1, 2:4, 3:9, 4:16, 5:25}

# menghapus anggota tertentu
# output: 9
print(pangkat.pop(3))

# menghapus anggota secara acak
# output: (1,1)
print(pangkat.popitem())

# yang tersisa adalah {2:4, 4:16, 5:25}
print(pangkat)

# delete 5
del pangkat[5]

# output: {2:4, 4:16}
print(pangkat)

# menghapus semua anggota
pangkat.clear()

# menghapus dictionary pangkat
del pangkat

# Error karena pangkat sudah dihapus
print(pangkat)
```

∞

Thanks for Attention