Python Data Structure

```
import warnings
warnings.filterwarnings('ignore')
```

I. String

- 문자열(String) 생성

```
S1 = 'The truth is out there.'
print(S1)
```

```
The truth is out there.
```

1) Concatenation

- '+': 문자열 연결
- '*': 문자열 반복

```
print('=' * 40)
print('Wt', S1)
print('=' * 40)
```

```
========================================
        The truth is out there.
========================================
```

```
S2 = 'The truth is'
S3 = ' out there.'
```

```
S2 + S3
```

```
S2 * 3
```

```
S3 * 3
```

2) Length

```
len(S1)
```

```
23
```

3) Indexing

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
T	h	e		t	r	u	t	h		i	s		o	u	t		t	h	e	r	e	.
-23	-22	-21	-20	-19	-18	-17	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(S1)
```

```
The truth is out there.
```

```
S1[1]
```

```
S1[5]
```

```
S1[0]
```

```
S1[-1]
```

```
S1[-13]
```

```
S1[-18]
```

4) Slicing

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
T	h	e		t	r	u	t	h		i	s		o	u	t		t	h	e	r	e	.
-23	-22	-21	-20	-19	-18	-17	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(S1)
```

```
The truth is out there.
```

- with Indexing

```
S1[4] + S1[5] + S1[6] + S1[7] + S1[8]
```

- with Slicing
  - S1[이상:미만]

```
S1[4:9]
```

```
S1[-19:-14]
```

```
S1[17:22]
```

```
S1[-6:-1]
```

```
S1[10:12]
```

```
S1[:12]
```

```
S1[13:]
```

5) Formatting

- '%s': String

```
'The X-Files is an American science fiction %s.' % 'Drama'
```

- 우측정렬

```
'%7s' % 'Drama'
```

- 좌측정렬

```
'%-7s' % 'Drama'
```

- '%d': Integer

```
'The program spanned nine seasons, with %d episodes.' % 202
```

- '%f': Floating Point

```
'%f' % 3.141593
```

```
'%.2f' % 3.141593
```

```
'%5.2f' % 3.141593
```

- 2개 값 대입

```
String = 'Drama'
Integer = 202

'The X-Files is an American science fiction %s, with %d episodes.' % (String, Integer)
```

II. List

1) [ ] 기호로 생성 - 값 변경 가능

```
L1 = [1, 3, 5, 7, 9]
print(L1)
```

```
[1, 3, 5, 7, 9]
```

```
print(type(L1))
print(type(L1[0]))
```

```
<class 'list'>
<class 'int'>
```

```
L2 = ['HP', 'IBM', 'DELL', 'EMC', 'MS']
print(L2)
```

```
['HP', 'IBM', 'DELL', 'EMC', 'MS']
```

```
print(type(L2))
print(type(L2[0]))
```

```
<class 'list'>
<class 'str'>
```

```
L3 = [1, '삼', 5, '칠', 9]
print(L3)
```

```
[1, '삼', 5, '칠', 9]
```

```
print(type(L3))
print(type(L3[0]))
print(type(L3[1]))
```

```
<class 'list'>
<class 'int'>
<class 'str'>
```

```
L4 = [1, 3, ['HP', 'MS']]
print(L4)
```

```
[1, 3, ['HP', 'MS']]
```

```
print(type(L4))
print(type(L4[1]))
print(type(L4[2]))
print(type(L4[2][0]))
```

```
<class 'list'>
<class 'int'>
<class 'list'>
<class 'str'>
```

```
L5 = [5, 7, ('IBM', 'EMC')]
print(L5)
```

```
[5, 7, ('IBM', 'EMC')]
```

```
print(type(L5))
print(type(L5[1]))
print(type(L5[2]))
print(type(L5[2][0]))
```

```
<class 'list'>
<class 'int'>
<class 'tuple'>
<class 'str'>
```

2) Indexing

- with L1

```
print(L1)
```

```
[1, 3, 5, 7, 9]
```

```
L1[2]
```

```
5
```

```
L1[2] + L1[4]
```

```
14
```

```
L1[-2]
```

```
7
```

- with L4

```
print(L4)
```

```
[1, 3, ['HP', 'MS']]
```

```
L4[1]
```

```
3
```

```
L4[2]
```

```
['HP', 'MS']
```

```
L4[2][1]
```

```
L4[2][0] + L4[2][1]
```

3) Slicing

- with L1

```
print(L1)
```

```
[1, 3, 5, 7, 9]
```

```
L1[1:4]
```

```
[3, 5, 7]
```

```
L1[:3]
```

```
[1, 3, 5]
```

```
L1[2:]
```

```
[5, 7, 9]
```

- with L6

```
L6 = [1, 3, 5, [2, 4, 6]]
print(L6)
```

```
[1, 3, 5, [2, 4, 6]]
```

```
L6[2:]
```

```
[5, [2, 4, 6]]
```

```
L6[3]
```

```
[2, 4, 6]
```

```
L6[3][0:2]
```

```
[2, 4]
```

4) Change Values

- 5 to 6

```
print(L1)
```

```
[1, 3, 5, 7, 9]
```

```
L1[2] = 6
print(L1)
```

```
[1, 3, 6, 7, 9]
```

5) Delete Values

```
print(L1)
```

```
[1, 3, 6, 7, 9]
```

```
L1[2:4] = []
print(L1)
```

```
[1, 3, 9]
```

```
del L1[2]
print(L1)
```

```
[1, 3]
```

- Error

```
del L1
print(L1)
```

6) Function( )

```
L7 = [8, 3, 9, 2, 1]
print(L7)
```

```
[8, 3, 9, 2, 1]
```

- 오름차순 정렬

```
L7.sort()
print(L7)
```

```
[1, 2, 3, 8, 9]
```

- 역순 정렬

```
L7.reverse()
print(L7)
```

```
[9, 8, 3, 2, 1]
```

- 마지막에 값('0') 추가

```
L7.append(0)
print(L7)
```

```
[9, 8, 3, 2, 1, 0]
```

- 2번 인덱스에 값('5') 추가

```
L7.insert(2, 5)
print(L7)
```

```
[9, 8, 5, 3, 2, 1, 0]
```

7) Operators

```
L8 = [85, 93, 75, 97, 69]
L9 = [91, 90, 85, 97, 89]
```

- Concatenation
  - '+': 리스트 연결

```
L8 + L9
```

```
[85, 93, 75, 97, 69, 91, 90, 85, 97, 89]
```

- Concatenation
  - '*': 리스트 반복

```
L8 * 2
```

```
[85, 93, 75, 97, 69, 85, 93, 75, 97, 69]
```

```
L9 * 3
```

```
[91, 90, 85, 97, 89, 91, 90, 85, 97, 89, 91, 90, 85, 97, 89]
```

III. Tuple

1) ( ) 기호로 생성 - 값 변경 불가능

```
T1 = (1, 2)
print(T1)
```

```
(1, 2)
```

- Error-1

```
del T1[0]
```

- Error-2

```
T1[0] = 'a'
```

2) Tuple in Tuple

```
T3 = (1, 2, (3, 4))
print(T3)
```

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

- Error-3

```
T3[2][0] = 6
```

3) List in Tuple

```
T4 = (1, 2, [3, 4])
print(T4)
```

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

- Change Values

```
T4[2][1] = 6
print(T4)
```

```
(1, 2, [3, 6])
```

IV. Dictionary

- 순서가 없음(Unordered)
- Key와 Value 한쌍으로 구성

1) {Key:Value} 구조 선언

```
D1 = {'Name':'LEE', 'Age':24}
print(D1)
```

```
{'Name': 'LEE', 'Age': 24}
```

```
print(type(D1))
print(type(D1['Name']))
print(type(D1['Age']))
```

```
<class 'dict'>
<class 'str'>
<class 'int'>
```

```
D1['Name']
```

2) Key:Value 추가

```
D1['Height'] = 183

print(D1)
```

```
{'Name': 'LEE', 'Age': 24, 'Height': 183}
```

3) Key:Value 삭제

```
del D1['Age']
print(D1)
```

```
{'Name': 'LEE', 'Height': 183}
```

4) Function( )

- Key 확인

```
D1.keys()
```

```
dict_keys(['Name', 'Height'])
```

- Value 확인

```
D1.values()
```

```
dict_values(['LEE', 183])
```

- Key:Value 삭제

```
print(D1)
D1.clear()
print(D1)
```

```
{'Name': 'LEE', 'Height': 183}
{}
```

5) Dictionary with List

```
L1 = ['Red', 'Green', 'Blue']
L2 = [255, 127, 63]
```

```
D2 = {x : y for x, y in zip(L1, L2)}
print(D2)
```

```
{'Red': 255, 'Green': 127, 'Blue': 63}
```

V. Casting

1) Data Type

- int to float

```
print(type(9))
print(float(9))
```

```
<class 'int'>
9.0
```

- str to float

```
print(type('9.4'))
print(float('9.4'))
```

```
<class 'str'>
9.4
```

- float to int

```
print(type(9.0))
print(9.0)
print(type(int(9.0)))
print(int(9.0))
```

```
<class 'float'>
9.0
<class 'int'>
9
```

- str to int

```
print(type('9'))
print('9')
print(type(int('9')))
print(int('9'))
```

```
<class 'str'>
9
<class 'int'>
9
```

- float to int
- Warning!!!

```
print(type(3.14))
print(3.14)
print(type(int(3.14)))
print(int(3.14))
```

```
<class 'float'>
3.14
<class 'int'>
3
```

- int to str

```
print(type(9))
print(type(str(9)))
```

```
<class 'int'>
<class 'str'>
```

- float to str

```
print(type(3.14))
print(type(str(3.14)))
```

```
<class 'float'>
<class 'str'>
```

2) Data Structure

- List to Tuple

```
tuple([1, 3, 5, 7, 9])
```

```
(1, 3, 5, 7, 9)
```

- Tuple to List

```
list((1, 3, 5, 7, 9))
```

```
[1, 3, 5, 7, 9]
```

- List to Dictionary

```
dict([['A', 123], ['B', 234], ['C', 567]])
```

```
    {'A': 123, 'B': 234, 'C': 567}
```

#

#

#

The End

#

#

#