

[Python_Mentoring]Mentoring_Code

파이썬 기본문법_자료형

숫자형

사칙연산

In [1]:

```
a = 10  
b = 5
```

In [2]:

```
a + b
```

Out[2]:

15

In [3]:

```
a - b
```

Out[3]:

5

In [4]:

```
a * b
```

Out[4]:

50

In [5]:

```
a / b
```

Out[5]:

2.0

제곱을 나타내는 연산자

In [6]:

```
a ** b
```

Out[6]:

100000

나눗셈 후 나머지 반환

In [8]:

```
a % b
```

Out[8]:

0

나눗셈 후 몫을 반환

In [9]:

```
a // b
```

Out[9]:

2

숫자형 실습

1. 국어 : 90, 수학: 80, 영어: 70 의 평균을 구하세요.

In [10]:

```
kor = 90  
ma = 80  
eng = 70
```

In [11]:

```
(kor + ma + eng) / 3
```

Out[11]:

80.0

1. 2의 12제곱을 구하세요.

In [12]:

```
2 ** 12
```

Out[12]:

4096

문자열

문자열 만들기

In [13]:

```
"큰따옴표 안에 문자열 넣기"
```

Out[13]:

```
'큰따옴표 안에 문자열 넣기'
```

In [14]:

```
'작은따옴표 안에 문자열 넣기'
```

Out[14]:

```
'작은따옴표 안에 문자열 넣기'
```

In [15]:

```
"""큰따옴표 3개 안에 문자열 넣기"""
```

Out[15]:

```
'큰따옴표 3개 안에 문자열 넣기'
```

In [16]:

```
'''작은따옴표 3개 안에 문자열 넣기'''
```

Out[16]:

```
'작은따옴표 3개 안에 문자열 넣기'
```

문자열 안에 따옴표 넣기

In [17]:

```
"큰따옴표"를 넣으려면 작은따옴표 안에 넣기'
```

Out[17]:

```
'"큰따옴표"를 넣으려면 작은따옴표 안에 넣기'
```

In [18]:

```
'작은따옴표'를 넣으려면 큰따옴표 안에 넣기"
```

Out[18]:

```
""작은따옴표'를 넣으려면 큰따옴표 안에 넣기"
```

In [19]:

```
'따옴표 앞에 \w'백슬래쉬\w'를 넣어도 됩니다.'
```

Out[19]:

```
"따옴표 앞에 '백슬래쉬'를 넣어도 됩니다."
```

문자열 여러줄 넣기

In [20]:

```
multilines = '''
작은따옴표를
이용한
여러줄의
글쓰기
입니다.
'''
```

In [22]:

```
print(multilines)
```

```
작은따옴표를
이용한
여러줄의
글쓰기
입니다.
```

In [23]:

```
multiline = """
큰따옴표를
이용한
여러줄의
글쓰기
입니다.
"""
```

In [24]:

```
print(multiline)
```

```
큰따옴표를
이용한
여러줄의
글쓰기
입니다.
```

In [25]:

```
multi = "이스케이프\n코드를\n이용한\n여러줄의\n글쓰기\n입니다."
```

In [26]:

```
print(multi)
```

이스케이프
코드를
이용한
여러줄의
글쓰기
입니다.

이스케이프 코드

- 프로그래밍할 때 사용할 수 있도록 미리 정의해 둔 "문자조합"
 - `\n`: 줄바꿈
 - `\t`: 수평탭
 - `\`: 문자 '\'
 - `\'`: 문자열 안의 작은따옴표
 - `\"`: 문자열 안의 큰따옴표

문자열 더하기

In [27]:

```
a = '문자열을'
b = '합쳐봅시다.'
```

In [28]:

```
a + b
```

Out[28]:

```
'문자열을합쳐봅시다.'
```

문자열 곱하기

In [29]:

```
a = '문자열을 반복해보아요.'
```

In [31]:

```
a * 3
```

Out[31]:

```
'문자열을 반복해보아요.문자열을 반복해보아요.문자열을 반복해보아요.'
```

문자열 반복 응용

1. 문자열 반복하기를 이용하여 사각형을 만들어보세요.

In [34]:

```
a = '*'
print(a * 3)
print(a, a)
print(a * 3)
```

```
***
* *
***
```

문자열 인덱싱

- 각 문자의 번호를 붙인 것
- 파이썬의 첫번째 숫자는 '0'

In [35]:

```
a = 'Life is too short, you need python'
```

In [36]:

```
a[3]
```

Out[36]:

```
'e'
```

In [37]:

```
a[-0]
```

Out[37]:

```
'L'
```

In [38]:

```
a[-1]
```

Out[38]:

```
'n'
```

문자열 슬라이싱

- 문자열 인덱싱처럼 단순히 한 문자만을 뽑아내는 것이 아니라 단어를 뽑아낼 수 있는 것.

문자열 인덱싱 VS. 문자열 슬라이싱

- 문자열 인덱싱

In [39]:

```
a = 'Life is too short, you need python'
```

In [40]:

```
word = a[0] + a[1] + a[2] + a[3]
```

In [41]:

```
word
```

Out[41]:

```
'Life'
```

- 문자열 슬라이싱

In [42]:

```
a[0:4]
```

Out[42]:

```
'Life'
```

문자열 슬라이싱

- 원하는 부분을 뽑아낼 수 있다.

In [43]:

```
a[19:]
```

Out[43]:

```
'you need python'
```

- 시작, 끝 번호를 생략하면 처음부터 뽑아내거나, 끝까지 뽑아낸다.

In [44]:

```
a[:]
```

Out[44]:

```
'Life is too short, you need python'
```

- ':' 기호를 사용할 수 있다.

In [45]:

```
a[19:-7]
```

Out[45]:

```
'you need'
```

슬라이싱으로 문자열 나누기

In [46]:

```
a = '20191020sunny'
```

In [47]:

```
date = a[:8]
```

In [48]:

```
weather = a[8:]
```

In [49]:

```
date
```

Out[49]:

```
'20191020'
```

In [50]:

```
weather
```

Out[50]:

```
'sunny'
```

(슬라이싱을 이용한)문자열 바꾸기

In [51]:

```
a = 'Jupiternotebook'
```

In [52]:

```
a[:3] + "y" + a[4:]
```

Out[52]:

```
'Jupyternotebook'
```

문자열 포매팅

- 문자열 내의 특정한 값을 바꿔야 할 경우가 있을 때 이것을 가능하게 해주는 것이 바로 문자열 포매팅이다.
- 숫자 대입

In [53]:

```
"나는 %d 개의 파이를 먹었다" % 5
```

Out[53]:

```
'나는 5 개의 파이를 먹었다'
```

- 문자열 대입

In [54]:

```
"나는 %s 하는 것을 좋아한다." % '파이썬 공부'
```

Out[54]:

```
'나는 파이썬 공부 하는 것을 좋아한다.'
```

- 변수 대입

In [55]:

```
temp = 5
'오늘의 온도는 %d 도 입니다.' % temp
```

Out[55]:

```
'오늘의 온도는 5 도 입니다.'
```

- 2개 이상의 값 넣기

In [56]:

```
task = 3
sle = 2
"나는 %d 개의 과제를 끝내고, %d 일 동안 잠을 잤다." % (task, sle)
```

Out[56]:

```
'나는 3 개의 과제를 끝내고, 2 일 동안 잠을 잤다.'
```

- % 문자 사용하기

In [57]:

```
hu = 74
'오늘의 습도는 %d% 입니다.' % hu
```

Out[57]:

```
'오늘의 습도는 74% 입니다.'
```

format() 함수 사용하기

- 포맷 함수 사용하여 포매팅하기

In [58]:

```
"나는 {0} 개의 사과를 먹었다.".format(3)
```

Out[58]:

```
'나는 3 개의 사과를 먹었다.'
```

- 2개 이상의 값 넣기

In [59]:

```
"나는 {0}개의 꿀을 먹고, {1}일 동안 잤다.".format(3, 5)
```

Out[59]:

```
'나는 3개의 꿀을 먹고, 5일 동안 잤다.'
```

문자열 관련 함수

- 문자열 길이(len())

In [60]:

```
a = 'Hello World! I love python world!'
```

In [61]:

```
len(a)
```

Out[61]:

```
33
```

- 특정 문자 개수 세기(.count())

In [62]:

```
a = 'shoot'  
a.count('o')
```

Out[62]:

```
2
```

- 특정 문자 위치 찾기(find, index)

In [63]:

```
a = 'Python is powerful'
```

In [64]:

```
a.find('i')
```

Out[64]:

7

In [65]:

```
a.find('z')
```

Out[65]:

-1

In [66]:

```
a.index('i')
```

Out[66]:

7

In [67]:

```
a.index('z')
```

```
-----  
----  
ValueError                                Traceback (most recent call last)  
<ipython-input-67-c8e03ccafa1c> in <module>  
----> 1 a.index('z')
```

ValueError: substring not found

문자열 관련 함수

- 문자열 삽입하기(.join())

In [69]:

```
','.join('abcd')
```

Out[69]:

'a,b,c,d'

- 대문자 만들기(.upper())

In [70]:

```
a = 'abcde'
```

In [71]:

```
a.upper()
```

Out[71]:

```
'ABCDE'
```

- 소문자 만들기(.lower())

In [72]:

```
b = 'ABCDE'  
b.lower()
```

Out[72]:

```
'abcde'
```

- 공백 지우기(.strip())

In [73]:

```
a = '   good   '  
a.lstrip()
```

Out[73]:

```
'good   '
```

In [74]:

```
a = '   good   '  
a.rstrip()
```

Out[74]:

```
'   good'
```

In [75]:

```
a = '   good   '  
a.strip()
```

Out[75]:

```
'good'
```

- 문자열 바꾸기(.replace(),)

In [76]:

```
a = 'Hello world, Python is fun'  
a.replace('Python', 'Game')
```

Out[76]:

```
'Hello world, Game is fun'
```

- 문자열 나누기(.split())

In [77]:

```
a.split()
```

Out[77]:

```
['Hello', 'world,', 'Python', 'is', 'fun']
```

In [78]:

```
a.split(' ',)
```

Out[78]:

```
['Hello world', ' Python is fun']
```

문자열 실습

1. 전화번호에서 하이픈을 제거하고 출력하세요

- 010-1111-2222

In [79]:

```
a = '010-1111-2222'  
a.replace('-', '')
```

Out[79]:

```
'01011112222'
```

1. 전화번호의 마지막 4자리만 출력하세요

- 010-1234-5678

In [83]:

```
a = '010-1234-5678'  
a[-4:]
```

Out[83]:

```
'5678'
```

1. 소문자 a를 대문자 A로 변경하세요

- abcdef1234aa32a

In [85]:

```
a = 'abcdef1234aa32a'  
a.replace('a', 'A')
```

Out[85]:

```
'Abcdef1234AA32A'
```

1. a:b:c:d 를 a#b#c#d로 바꾸세요

- replace 함수 사용
- split과 join 함수 사용

In [86]:

```
a = 'a:b:c:d'
a.replace(':', '#')
```

Out[86]:

```
'a#b#c#d'
```

In [91]:

```
a = 'a:b:c:d'
a = a.split(':')
a = '#'.join(a)
print(a)
```

```
a#b#c#d
```

리스트

리스트의 인덱싱

- 문자열과 인덱싱 방법은 똑같음

In [92]:

```
a = [4,8,7]
```

In [93]:

```
a
```

Out[93]:

```
[4, 8, 7]
```

In [94]:

```
a[0]
```

Out[94]:

```
4
```

In [95]:

```
a[-1]
```

Out[95]:

```
7
```

- 결과값을 뽑아서 사칙연산을 할 수 있음

In [96]:

```
a[1] + a[2]
```

Out[96]:

15

In [97]:

```
a[1] * a[2]
```

Out[97]:

56

이중 리스트의 인덱싱

- 리스트 안에 리스트가 있을 때는 대괄호를 2번 사용하여 결과값을 가져옴

In [98]:

```
a = [1, 2, 3, ['a', 'b', 'c']]
```

In [99]:

```
a
```

Out[99]:

```
[1, 2, 3, ['a', 'b', 'c']]
```

In [100]:

```
a[-1]
```

Out[100]:

```
['a', 'b', 'c']
```

In [101]:

```
a[-1][0]
```

Out[101]:

```
'a'
```

삼중 리스트의 인덱싱

- 이중 리스트 인덱싱을 할 때처럼 대괄호 3번으로 결과값을 뽑아냄

In [102]:

```
a = [1, 2, ['a', 'b', ['Life', 'is']]]
```

In [103]:

```
a
```

Out[103]:

```
[1, 2, ['a', 'b', ['Life', 'is']]]
```

In [104]:

```
a[-1][-1][0]
```

Out[104]:

```
'Life'
```

리스트의 슬라이싱

- 인덱싱처럼 문자열의 슬라이싱과 완전히 같음

In [105]:

```
a = [1,2,3,4,5]
```

In [106]:

```
a[0:2]
```

Out[106]:

```
[1, 2]
```

In [107]:

```
a[:2]
```

Out[107]:

```
[1, 2]
```

In [108]:

```
a[2:]
```

Out[108]:

```
[3, 4, 5]
```

이중 리스트의 슬라이싱

- 인덱싱과 똑같이 이중 리스트의 인덱싱을 해준 후에 슬라이싱을 하면 됨

In [109]:

```
a = [1, 2, 3, ['a', 'b', 'c'], 4, 5]
```

In [110]:

```
a[2:5]
```

Out[110]:

```
[3, ['a', 'b', 'c'], 4]
```

In [111]:

```
a[3][:2]
```

Out[111]:

```
['a', 'b']
```

리스트 연산

In [112]:

```
a = ['a', 'b', 'c']  
b = ['d', 'e', 'f']
```

- 리스트 더하기

In [113]:

```
a + b
```

Out[113]:

```
['a', 'b', 'c', 'd', 'e', 'f']
```

- 리스트 반복하기

In [114]:

```
a * 3
```

Out[114]:

```
['a', 'b', 'c', 'a', 'b', 'c', 'a', 'b', 'c']
```

- 리스트 길이 구하기

In [115]:

```
len(a)
```

Out[115]:

```
3
```

리스트 요소 수정 & 삭제

- 리스트 요소 수정

In [116]:

```
a = [1,2,3]
```

In [117]:

```
a[2] = 4
```

In [118]:

```
a
```

Out[118]:

```
[1, 2, 4]
```

- 리스트 요소 삭제

In [119]:

```
a = [1,2,3]
```

In [120]:

```
del a[1]
```

In [121]:

```
a
```

Out[121]:

```
[1, 3]
```

In [122]:

```
a = [1,2,3,4,5]
```

In [123]:

```
del a[2:]
```

In [124]:

```
a
```

Out[124]:

```
[1, 2]
```

리스트 관련 함수들

- 리스트 요소 추가(.append())

In [125]:

```
a = [1,2,3]
```

In [126]:

```
a.append(4)
```

In [127]:

```
a
```

Out[127]:

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

In [128]:

```
a.append([5,6])
```

In [129]:

```
a
```

Out[129]:

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

- 리스트 정렬(.sort())

In [130]:

```
a = [1,5,3,4,2]
```

In [131]:

```
a.sort()
```

In [132]:

```
a
```

Out[132]:

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

In [133]:

```
a = ['a', 'd', 'b', 'c']
```

In [134]:

```
a.sort()
```

In [135]:

```
a
```

Out[135]:

```
['a', 'b', 'c', 'd']
```

- 리스트 뒤집기(.reverse())

In [136]:

```
a = ['c', 'd', 'b', 'a']
```

In [137]:

```
a.reverse()
```

In [138]:

```
a
```

Out[138]:

```
['a', 'b', 'd', 'c']
```

- 리스트 위치 반환(index(x))-x의 위치를 반환

In [140]:

```
a = [1,2,3]
```

In [141]:

```
a.index(2)
```

Out[141]:

```
1
```

In [142]:

```
a.index(3)
```

Out[142]:

```
2
```

- 리스트 요소 삽입(insert(x,y))-x의 위치에 y를 삽입

In [143]:

```
a = [1,2,3]
```

In [144]:

```
a.insert(0,4)
```

In [145]:

```
a
```

Out[145]:

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

In [146]:

```
a.insert(3,5)
```

In [147]:

```
a
```

Out[147]:

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

- 리스트 요소 제거(.remove(x))-리스트 안에 있는 x 중에 가장 앞에 있는 요소 제거

In [148]:

```
a = [1,2,3,1,2,3]
```

In [149]:

```
a.remove(3)
```

In [150]:

```
a
```

Out[150]:

```
[1, 2, 1, 2, 3]
```

In [151]:

```
a.remove(3)
```

In [152]:

```
a
```

Out[152]:

```
[1, 2, 1, 2]
```

- 리스트 요소 끄집어내기(.pop(x))-x번째 안에 있는 요소를 출력하고 삭제

In [153]:

```
a = [1,2,3]
```

In [155]:

```
a.pop()
```

Out[155]:

3

In [156]:

```
a
```

Out[156]:

[1, 2]

- 리스트에 포함된 요소 x의 개수 세기(.count(x))

In [157]:

```
a = [1, 1, 1, 2, 3]
```

In [158]:

```
a.count(1)
```

Out[158]:

3

- 리스트 확장

In [159]:

```
a = [1, 2, 3]
```

In [160]:

```
a.extend([4, 5])
```

In [161]:

```
a
```

Out[161]:

[1, 2, 3, 4, 5]

리스트 실습

1. '닥터 스트레인지', '스플릿', '럭키' 를 포함한 리스트를 만드세요.

In [163]:

```
movie = ['닥터 스트레인지', '스플릿', '럭키']
```

In [164]:

```
movie
```

Out [164]:

```
['닥터 스트레인지', '스플릿', '럭키']
```

1. 리스트 뒤에 '배트맨' 요소를 추가하세요.

In [165]:

```
movie.append('배트맨')
```

In [166]:

```
movie
```

Out [166]:

```
['닥터 스트레인지', '스플릿', '럭키', '배트맨']
```

1. '닥터 스트레인지' 와 '스플릿' 사이에 '슈퍼맨'을 추가하세요.

In [167]:

```
movie.insert(1, '슈퍼맨')
```

In [168]:

```
movie
```

Out [168]:

```
['닥터 스트레인지', '슈퍼맨', '스플릿', '럭키', '배트맨']
```

1. '럭키' 요소를 삭제하세요.

In [170]:

```
movie.remove('럭키')
```

In [171]:

```
movie
```

Out [171]:

```
['닥터 스트레인지', '슈퍼맨', '스플릿', '배트맨']
```

1. '스플릿' 요소를 뽑아낸 후 삭제하세요.

In [173]:

```
movie.pop(2)
```

Out[173]:

'스플릿'

In [174]:

```
movie
```

Out[174]:

['닥터 스트레인지', '슈퍼맨', '배트맨']

튜플

튜플의 형태

In [175]:

```
t1 = (1,)
```

In [176]:

```
t2 = 1,2,3
```

In [177]:

```
t3 = ('a', 'b', ('ab', 'cd'))
```

- 튜플은 한 개의 요소만을 가질 때는 요소 뒤에 ,(콤마)를 반드시 붙여야 한다.
- 괄호를 생략해도 무관하다.
- 리스트는 항목 값을 변경(삭제)할 수 있지만, 튜플은 불가능하다.
- 리스트와 매우 유사하다.

튜플의 인덱싱

In [178]:

```
t1 = (1, 2, 'a', 'b')
```

In [179]:

```
t1[0]
```

Out[179]:

1

In [180]:

```
t1[3]
```

Out[180]:

```
'b'
```

튜플의 슬라이싱

In [181]:

```
t1[2:]
```

Out[181]:

```
('a', 'b')
```

튜플의 더하기, 곱하기

- 더하기

In [182]:

```
t1 = (1, 2, 'a', 'b')  
t2 = (3, 4)
```

In [183]:

```
t1 + t2
```

Out[183]:

```
(1, 2, 'a', 'b', 3, 4)
```

- 곱하기

In [184]:

```
t2 = (3, 4)
```

In [185]:

```
t2 * 3
```

Out[185]:

```
(3, 4, 3, 4, 3, 4)
```

튜플의 길이 구하기

In [186]:

```
t1 = (1, 2, 'a', 'b')
len(t1)
```

Out[186]:

4

리스트와 튜플 변경

- 리스트에서 튜플 만들기

In [187]:

```
a = [1,2,3,4,5,6,7]
```

In [188]:

```
tuple(a)
```

Out[188]:

(1, 2, 3, 4, 5, 6, 7)

- 튜플에서 리스트 만들기

In [189]:

```
b = (1,2,3,4,5,6,7)
```

In [190]:

```
list(b)
```

Out[190]:

[1, 2, 3, 4, 5, 6, 7]

튜플 실습

1. (1,3,5)라는 튜플에 7값을 추가하여 (1,3,5,7)처럼 만들어 출력해보세요.

In [192]:

```
t = (1,3,5)
```

In [193]:

```
t1 = (7,)
```

In [194]:

```
t + t1
```

Out[194]:

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

1. [1,2,3,4]라는 리스트를 튜플로 변환해보세요.

In [195]:

```
l = [1,2,3,4]
```

In [196]:

```
tuple(l)
```

Out[196]:

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

1. (1,3,5,7)이라는 튜플을 리스트로 변환해보세요.

In [197]:

```
tt = (1,3,5,7)
```

In [198]:

```
list(tt)
```

Out[198]:

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

딕셔너리

딕셔너리 형태

In [199]:

```
Jeong_dic = {'name': 'Jeong', 'phone': '01094739051', 'birth': '0618'}
```

In [200]:

```
a = {1: 'hi'}  
b = {'a': [1,2,3]}
```

딕셔너리 쌍 추가하기

In [201]:

```
a = {1: 'hi'}
```

In [202]:

```
a[2] = 'b'
```

In [203]:

```
a
```

Out[203]:

```
{1: 'hi ', 2: 'b'}
```

In [204]:

```
a['subject'] = 'python'
```

In [205]:

```
a
```

Out[205]:

```
{1: 'hi ', 2: 'b', 'subject': 'python'}
```

In [206]:

```
a[3] = [1,2,3]
```

In [207]:

```
a
```

Out[207]:

```
{1: 'hi ', 2: 'b', 'subject': 'python', 3: [1, 2, 3]}
```

딕셔너리 요소 삭제하기

- del 함수를 사용하여 대괄호 안에 key를 입력하면 key와 value가 삭제된다

In [209]:

```
a
```

Out[209]:

```
{1: 'hi ', 2: 'b', 'subject': 'python', 3: [1, 2, 3]}
```

In [210]:

```
del a[1]
```

In [211]:

```
a
```

Out[211]:

```
{2: 'b', 'subject': 'python', 3: [1, 2, 3]}
```

In [212]:

```
del a['subject']
```

In [213]:

```
a
```

Out[213]:

```
{2: 'b', 3: [1, 2, 3]}
```

딕셔너리에서 key를 사용해 value 얻기

In [214]:

```
Jeong_dic['name']
```

Out[214]:

```
'Jeong'
```

In [215]:

```
Jeong_dic['birth']
```

Out[215]:

```
'0618'
```

In [216]:

```
a[2]
```

Out[216]:

```
'b'
```

In [217]:

```
a[3]
```

Out[217]:

```
[1, 2, 3]
```

딕셔너리 만들 때 주의 사항

- key사 중복되어선 안된다.

In [218]:

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

In [219]:

```
a
```

Out[219]:

```
{1: 'b'}
```

- key에는 리스트가 사용될 수 없다. 튜플은 가능하다.

In [220]:

```
a = {[1,2,3]:'a'}
```

```
-----
TypeError                                Traceback (most recent call last)
<ipython-input-220-5ae01e0d3ca7> in <module>
----> 1 a = {[1,2,3]:'a'}
```

TypeError: unhashable type: 'list'

In [221]:

```
a = {(1,2,3):'a'}
```

In [222]:

```
a
```

Out[222]:

```
{(1, 2, 3): 'a'}
```

딕셔너리 관련 함수들

- key 리스트 만들기(.keys())

In [223]:

```
Jeong_dic
```

Out[223]:

```
{'name': 'Jeong', 'phone': '01094739051', 'birth': '0618'}
```

In [224]:

```
Jeong_dic.keys()
```

Out[224]:

```
dict_keys(['name', 'phone', 'birth'])
```

- value 리스트 만들기(.values())

In [225]:

```
Jeong_dic.values()
```

Out [225]:

```
dict_values(['Jeong', '01094739051', '0618'])
```

- key, value 쌍 얻기(.items())

In [226]:

```
Jeong_dic.items()
```

Out [226]:

```
dict_items([('name', 'Jeong'), ('phone', '01094739051'), ('birth', '0618')])
```

- key:value 쌍 모두 지우기(.clear())

In [227]:

```
a
```

Out [227]:

```
{(1, 2, 3): 'a'}
```

In [228]:

```
a.clear()
```

In [229]:

```
a
```

Out [229]:

```
{}
```

- key로 value 얻기(.get())
 - key 값이 없을 경우, 디폴트 값을 지정해 줄 수 있다.

In [230]:

```
Jeong_dic.get('name')
```

Out [230]:

```
'Jeong'
```

In [231]:

```
Jeong_dic.get('re')
```

In [232]:

```
Jeong_dic.get('re', 'none') # key 값이 없을 경우 뒤에 있는 디폴트 값을 출력해라
```

Out[232]:

'none'

- 해당 key가 딕셔너리 안에 있는지 조사하기(" in dic)

In [233]:

```
'name' in Jeong_dic
```

Out[233]:

True

In [234]:

```
're' in Jeong_dic
```

Out[234]:

False

딕셔너리 실습

1. 딕셔너리를 만드세요

- 메로나 : 1000 , 플라포 : 1200 , 빵빠레 : 1800

In [237]:

```
ice = {'메로나':1000, '플라포':1200, '빵빠레':1800}
```

In [238]:

```
ice
```

Out[238]:

```
{'메로나': 1000, '플라포': 1200, '빵빠레': 1800}
```

1. 딕셔너리를 추가하세요

- 죠스바 : 1200 , 월드콘 : 1500

In [239]:

```
ice['죠스바'] = 1200
```

In [240]:

```
ice['월드콘'] = 1500
```

In [241]:

```
ice
```

Out[241]:

```
{'메로나': 1000, '플라포': 1200, '빵빠레': 1800, '조스바': 1200, '월드콘': 1500}
```

1. 메로나의 가격을 출력하세요

In [242]:

```
ice.get('메로나')
```

Out[242]:

```
1000
```

1. 플라포를 딕셔너리에서 삭제하세요

In [243]:

```
del ice['플라포']
```

In [244]:

```
ice
```

Out[244]:

```
{'메로나': 1000, '빵빠레': 1800, '조스바': 1200, '월드콘': 1500}
```

1. 딕셔너리에서 누가바를 출력했을때, '없습니다.'라고 출력하게 만드세요.

In [245]:

```
ice.get('누가바', '없습니다.')
```

Out[245]:

```
'없습니다.'
```

복습문제

1. list1 = ['a', 'c', 'd', 'b', 'e']라는 리스트를 list1 = ['a', 'b', 'c', 'd', 'e']로 만들어 보세요.

In [246]:

```
list1 = ['a', 'c', 'd', 'b', 'e']
```

In [247]:

```
list1.sort()
```

In [248]:

```
list1
```

Out[248]:

```
['a', 'b', 'c', 'd', 'e']
```

1. list2 = ['This', 'is', 'a', 'book.']라는 리스트를 change라는 변수를 사용해서 change = This is a book 라는 문자열을 만들어 보세요. (문자열에서 사용 했던 join함수를 활용해 보세요.)

In [249]:

```
list2 = ['This', 'is', 'a', 'book.']
```

In [250]:

```
change = ' '.join(list2)
```

In [251]:

```
change
```

Out[251]:

```
'This is a book.'
```

1. tuple1 = (1, 2, 3, 4)라는 튜플에 5라는 값을 추가해서 (1, 2, 3, 4, 5)처럼 만들어 보세요.

In [252]:

```
tuple1 = (1,2,3,4)
```

In [253]:

```
tuple2 = (5,)
```

In [254]:

```
tuple1 + tuple2
```

Out[254]:

```
(1, 2, 3, 4, 5)
```

1. dic1 = {'A':90, 'B':80, 'C':70, 'D':60, 'F':50}에서 'D'에 해당 되는 값을 추출한 값은(dic1)에 삭제되는 값은(dic2)에 넣어 만들어 보세요.

In [260]:

```
dic1 = {'A':90, 'B':80, 'C':70, 'D':60, 'F':50}
```

In [261]:

```
dic2 = dic1.pop('D')
```

In [262]:

dic2

Out [262]:

60

In [263]:

dic1

Out [263]:

{'A': 90, 'B': 80, 'C': 70, 'F': 50}

파이썬 제어문

IF문

- 주어진 조건을 판단한 후 그 상황에 맞게 처리하기 위해 사용하는 것이 if문이다.
- 참과 거짓을 판단하는 문장
- 주의할 점
 - 들여쓰기(인덴팅)을 잘해야한다.
 - 조건문 다음에 콜론(:)을 무조건 붙여야한다.
 - 콜론을 붙이면 자동으로 인덴팅이 된다. 하지만 인덴팅을 해야할 경우가 있으면, 스페이스 4번 혹은 탭을 누르면 된다.

if문의 기본 구조

In [264]:

```
money = True
if money :
    print('택시를 타고 가라')
else:
    print('걸어가라')
```

택시를 타고 가라

비교연산자

- $x < y$: x가 y보다 작다.
- $x > y$: x가 y보다 크다.
- $x == y$: x와 y가 같다.
- $x != y$: x와 y가 같지 않다.
- $x >= y$: x가 y보다 크거나 같다.
- $x <= y$: x가 y보다 작거나 같다.

In [265]:

```
money = 2000
if money >= 3000 :
    print('택시를 타고 가라')
else:
    print('걸어가라')
```

걸어가라

And, Or, Not

- `x or y` : `x`와 `y` 둘중에 하나만 참이면 참이다.
- `x and y` : `x`와 `y` 모두 참이어야 참이다.
- `not x` : `x`가 거짓이면 참이다.

In [266]:

```
money = 2000
card = True
if money >= 3000 or card:
    print('택시를 타고 가라')
else:
    print('걸어가라')
```

택시를 타고 가라

in, not in

- `x in s`(리스트, 튜플, 문자열) : `s` 안에 `x`가 있다.
- `x not in s`(리스트, 튜플, 문자열) : `s` 안에 `x`가 없다.

In [267]:

```
1 in [1,2,3]
```

Out[267]:

True

In [268]:

```
1 not in [1,2,3]
```

Out[268]:

False

In [269]:

```
'a' in ('a', 'b', 'c')
```

Out[269]:

True

In [270]:

```
'a' in 'python'
```

Out[270]:

False

In [271]:

```
pocket = ['paper', 'cellphone', 'money']
if 'money' in pocket:
    print('택시를 타고 가라')
else:
    print('걸어가라')
```

택시를 타고 가라

조건문에서 아무 일도 하지 않게 설정하고 싶다면?

- pass

In [273]:

```
pocket = ['paper', 'money', 'cellphone']
if 'money' in pocket:
    pass
else:
    print('카드를 꺼내라')
```

다양한 조건을 판단하는 elif

In [274]:

```
pocket = ['paper', 'cellphone']
card = True
if 'money' in pocket:
    print('택시를 타고 가라')
else:
    if card:
        print('택시를 타고 가라')
    else:
        print('걸어가라')
```

택시를 타고 가라

In [275]:

```
pocket = ['paper', 'cellphone']
card = True
if 'money' in pocket:
    print('택시를 타고가라')
elif card:
    print('택시를 타고가라')
else:
    print('걸어가라')
```

택시를 타고가라

if문 한 줄로 작성하기

In [276]:

```
if 'money' in pocket:
    pass
else:
    print('카드를 꺼내라')
```

카드를 꺼내라

In [277]:

```
if 'money' in pocket: pass
else: print('카드를 꺼내라')
```

카드를 꺼내라

조건부 표현식

In [278]:

```
score = 60
if score >= 60:
    message = 'success'
else:
    message = 'failure'
```

In [279]:

```
message = 'success' if score >= 60 else 'failure'
```

연습문제

1. 사용자로부터 값을 입력받은 후 해당 값에 +20을 더한 값을 출력하세요.(단, 20을 더한 값이 255이상이라면 255를 출력하게 하세요.)
 - input 함수 사용

In [282]:

```
a = int(input('값을 입력해주세요: '))
if a >= 235:
    print(255)
else:
    print(a+20)
```

값을 입력해주세요: 230

250

1. 사용자로부터 값을 입력받은 후 해당 값에 20을 뺀 값을 출력하세요.(단, 20을 뺀 값이 0보다 작으면 0을 출력하세요.)
 - input 함수 사용

In [284]:

```
a = int(input('값을 입력해주세요: '))
if a <= 20:
    print(0)
else:
    print(a-20)
```

값을 입력해주세요: 21

1

1. 사용자로부터 입력 받은 시간이 정각인지 판별하는 함수를 만드세요.
 - input 함수 사용

In [286]:

```
a = input('시간을 입력해주세요: ')
if a[-2:] == '00':
    print('정각입니다.')
else:
    print('정각이 아닙니다.')
```

시간을 입력해주세요: 02:00

정각입니다.

1. 사용자가 입력한 값이 fruit 리스트에 포함되어 있는지 확인하고, 포함되었으면 '정답입니다.', 포함되어 있지 않다면 '오답입니다.'를 출력하세요.
 - fruit = ["사과", "포도", "홍시"]
 - input 함수 사용

In [288]:

```
fruit = ['사과', '포도', '홍시']
a = input('과일 이름을 입력해주세요: ')
if a in fruit:
    print('정답입니다.')
else:
    print('오답입니다.')
```

과일 이름을 입력해주세요: 딸기
오답입니다.

1. 사용자가 입력한 값이 fruit 딕셔너리 키(key) 값에 포함되었다면 “정답입니다.”, 아니면 “오답입니다.”를 출력하세요.

- fruit = {"봄": "딸기", "여름": "토마토", "가을": "사과"}
- input 함수 사용

In [290]:

```
fruit = {'봄' : '딸기', '여름' : '토마토', '가을' : '사과'}
a = input('계절을 입력하세요: ')
if a in fruit.keys():
    print('정답입니다.')
else:
    print('오답입니다.')
```

계절을 입력하세요: 가을
정답입니다.

1. 사용자가 입력한 값이 fruit 딕셔너리 값(Value)에 포함되었다면 “정답입니다.”, 아니면 “오답입니다.”를 출력하세요.

- fruit = {"봄": "딸기", "여름": "토마토", "가을": "사과"}
- input 함수 사용

In [292]:

```
fruit = {'봄' : '딸기', '여름' : '토마토', '가을' : '사과'}
a = input('과일을 입력하세요: ')
if a in fruit.values():
    print('정답입니다.')
else:
    print('오답입니다.')
```

과일을 입력하세요: 수박
오답입니다.

1. 사용자가 입력한 점수에 해당하는 학점을 출력하세요.

- input 함수 사용

In [298]:

```
a = int(input('점수를 입력해주세요: '))
if 100 >= a >= 81:
    print('A')
elif 80 >= a >= 61:
    print('B')
elif 60 >= a >= 41:
    print('C')
elif 40 >= a >= 21:
    print('D')
elif 20 >= a >= 0:
    print('E')
else:
    print('이 세상 점수가 아닙니다.')
```

점수를 입력해주세요: 120
이 세상 점수가 아닙니다.

while문

while문의 기본형

In [1]:

```
treehit = 0
while treehit < 10:
    treehit = treehit + 1
    print("나무를 %d번 찍었습니다." % treehit)
    if treehit == 10:
        print('나무 넘어갑니다.')
```

나무를 1번 찍었습니다.
나무를 2번 찍었습니다.
나무를 3번 찍었습니다.
나무를 4번 찍었습니다.
나무를 5번 찍었습니다.
나무를 6번 찍었습니다.
나무를 7번 찍었습니다.
나무를 8번 찍었습니다.
나무를 9번 찍었습니다.
나무를 10번 찍었습니다.
나무 넘어갑니다.

while문 강제로 빠져나가기

In [2]:

```
coffee = 10
money = 300
while money:
    print('돈을 받았으니 커피를 줍니다.')
    coffee = coffee - 1
    print('남은 커피의 양은 %d개 입니다.' % coffee)
    if coffee == 0:
        print('커피가 다 떨어졌습니다. 판매를 중지합니다.')
        break
```

돈을 받았으니 커피를 줍니다.
남은 커피의 양은 9개 입니다.
돈을 받았으니 커피를 줍니다.
남은 커피의 양은 8개 입니다.
돈을 받았으니 커피를 줍니다.
남은 커피의 양은 7개 입니다.
돈을 받았으니 커피를 줍니다.
남은 커피의 양은 6개 입니다.
돈을 받았으니 커피를 줍니다.
남은 커피의 양은 5개 입니다.
돈을 받았으니 커피를 줍니다.
남은 커피의 양은 4개 입니다.
돈을 받았으니 커피를 줍니다.
남은 커피의 양은 3개 입니다.
돈을 받았으니 커피를 줍니다.
남은 커피의 양은 2개 입니다.
돈을 받았으니 커피를 줍니다.
남은 커피의 양은 1개 입니다.
돈을 받았으니 커피를 줍니다.
남은 커피의 양은 0개 입니다.
커피가 다 떨어졌습니다. 판매를 중지합니다.

while문 강제로 빠져나가기

In [4]:

```

coffee = 10
while True:
    money = int(input('돈을 넣어주세요: '))
    if money == 300:
        print('커피를 줍니다.')
        coffee = coffee - 1
    elif money > 300:
        print('거스름돈 %d를 주고 커피를 줍니다.' % (money - 300))
        coffee = coffee - 1
    else:
        print('돈을 다시 돌려주고 커피를 주지 않습니다.')
        print('남은 커피의 양은 %d개 입니다.' % coffee)
    if coffee == 0:
        print('커피가 다 떨어졌습니다. 판매를 중지합니다.')
        break

```

```

돈을 넣어주세요: 300
커피를 줍니다.
커피가 다 떨어졌습니다. 판매를 중지합니다.

```

while문의 맨 처음으로 돌아가기

In [5]:

```

a = 0
while a < 10:
    a = a + 1
    if a % 2 == 0:
        continue
    print(a)

```

```

1
3
5
7
9

```

무한루프

In [6]:

```
while True:  
    print('계속 진행됩니다.')
```



```

-----
KeyboardInterrupt                                Traceback (most recent call last)
<ipython-input-6-27d0d798353e> in <module>
      1 while True:
----> 2     print('계속 진행됩니다.')

~\Anaconda3\lib\site-packages\ipykernel\iostream.py in write(self, string)
    400         is_child = (not self._is_master_process())
    401         # only touch the buffer in the IO thread to avoid race
S
--> 402         self.pub_thread.schedule(lambda : self._buffer.write(string))
    403         if is_child:
    404             # newlines imply flush in subprocesses

~\Anaconda3\lib\site-packages\ipykernel\iostream.py in schedule(self, f)
    203         self._events.append(f)
    204         # wake event thread (message content is ignored)
--> 205         self._event_pipe.send(b'')
    206     else:
    207         f()

~\Anaconda3\lib\site-packages\zmq\sugar\socket.py in send(self, data, flags, copy, track, routing_id, group)
    398         copy_threshold=self.copy_threshold)
    399         data.group = group
--> 400         return super(Socket, self).send(data, flags=flags, copy=copy,
track=track)
    401
    402     def send_multipart(self, msg_parts, flags=0, copy=True, track=False
, **kwargs):

zmq/backend/cython/socket.pyx in zmq.backend.cython.socket.Socket.send()

zmq/backend/cython/socket.pyx in zmq.backend.cython.socket.Socket.send()

zmq/backend/cython/socket.pyx in zmq.backend.cython.socket._send_copy()

~\Anaconda3\lib\site-packages\zmq\backend\cython\checkrc.pxd in zmq.backend.cython.checkrc._check_rc()

KeyboardInterrupt:

```

while문 연습문제

1. 1부터 1000까지의 자연수 중 3의 배수의 합을 구하세요.

In [7]:

```
a = 0
b = 0
while a <= 1000:
    a = a + 1
    if a % 3 == 0:
        b = b + a
print(b)
```

166833

1. 50이상의 수들의 총합을 구하세요.

- A = [20, 55, 67, 82, 45, 33, 90, 87, 100, 25]

In [10]:

```
A = [20, 55, 67, 82, 45, 33, 90, 87, 100, 25]
B = 0
while A:
    C = A.pop()
    if C >= 50:
        B = B + C
print(B)
```

481

1. 별을 차례대로 1~10개를 출력하세요.

In [12]:

```
a = 0
while a <= 10:
    print(a*' *')
    a = a + 1
```

```
*
**
***
****
*****
*****
*****
*****
*****
*****
*****
```

for문

- 전형적인 for문

In [13]:

```
test_list = ['one', 'two', 'three']
for i in test_list:
    print(i)
```

```
one
two
three
```

- for문의 활용

In [14]:

```
a = [(1,2), (3,4), (5,6)]
for (first, last) in a:
    print(first + last)
```

```
3
7
11
```

for문의 응용

- 총 5명의 학생이 시험을 보았는데 시험 점수가 60점이 넘으면 합격이고 그렇지 않으면 불합격이다. 불합격인지 결과를 보여주세요.

In [15]:

```
marks = [90, 25, 67, 45, 80]
number = 0

for mark in marks:
    number = number + 1
    if mark >= 60:
        print('%d번 학생은 합격입니다.' % number)
    else:
        print('%d번 학생은 불합격입니다.' % number)
```

```
1번 학생은 합격입니다.
2번 학생은 불합격입니다.
3번 학생은 합격입니다.
4번 학생은 불합격입니다.
5번 학생은 합격입니다.
```

for문과 continue

In [17]:

```
marks = [90, 25, 67, 45, 80]
number = 0

for mark in marks:
    number = number + 1
    if mark >= 60:
        print('%d번 학생 축하합니다. 합격입니다.' % number)
    else:
        continue
```

1번 학생 축하합니다. 합격입니다.
3번 학생 축하합니다. 합격입니다.
5번 학생 축하합니다. 합격입니다.

for문과 range 함수

- range 함수는 범위를 나타내줍니다.
- range(시작번호, 끝번호) : 시작번호부터 끝번호까지
- range(끝번호) : 0부터 끝번호까지

In [19]:

```
for i in range(3,10):
    print(i)
```

3
4
5
6
7
8
9

In [21]:

```
for i in range(10):
    print(i)
```

0
1
2
3
4
5
6
7
8
9

range 함수의 예시

In [22]:

```
sum = 0
for i in range(1,11):
    sum = sum + i
print(sum)
```

55

In [23]:

```
marks = [90, 25, 67, 45, 80]
for number in range(len(marks)):
    if marks[number] < 60:
        continue
    print('%d번 학생 축하합니다. 합격입니다.' % (number + 1))
```

1번 학생 축하합니다. 합격입니다.

3번 학생 축하합니다. 합격입니다.

5번 학생 축하합니다. 합격입니다.

for문으로 리스트 만들기

In [24]:

```
a = [1,2,3,4]
result = []
for num in a:
    result.append(num*3)
print(result)
```

[3, 6, 9, 12]

In [25]:

```
result = []
for i in range(5):
    result.append(i)
print(result)
```

[0, 1, 2, 3, 4]

for문 연습 문제

1. for문을 사용하여 5~12의 자연수가 들어있는 리스트를 만드세요.

In [26]:

```
num = []
for i in range(5,13):
    num.append(i)
print(num)
```

[5, 6, 7, 8, 9, 10, 11, 12]

1. for문을 이용하여 학급의 평균점수를 구하세요.
 - A = [70, 60, 55, 75, 95, 90, 80, 80 ,85, 100]

In [27]:

```
A = [70, 60, 55, 75, 95, 90, 80, 80 ,85, 100]
sum_num = 0
for i in A:
    sum_num += i
print(sum_num/len(A))
```

79.0

1. 동물의 이름과 글자수를 출력하세요.
 - pets = ['dog', 'cat', 'parrot', 'squirrel', 'goldfish']

In [28]:

```
pets = ['dog', 'cat', 'parrot', 'squirrel', 'goldfish']
for i in pets:
    print(i, len(i))
```

```
dog 3
cat 3
parrot 6
squirrel 8
goldfish 8
```

1. for문을 이용하여 리스트를 거꾸로 출력하세요.
 - B = ['가', '나', '다', '라']

In [31]:

```
B = ['가', '나', '다', '라']
C = []
for i in B:
    C = list(i) + C
print(C)
```

['라', '다', '나', '가']

함수

함수 예시

In [32]:

```
def gugudan(n):
    print('-----',n,'단', '-----')
    for i in range(1,10):
        print(n,'*',i,'=',n*i)
```

In [33]:

```
gugudan(3)
```

```
----- 3 단 -----  
3 * 1 = 3  
3 * 2 = 6  
3 * 3 = 9  
3 * 4 = 12  
3 * 5 = 15  
3 * 6 = 18  
3 * 7 = 21  
3 * 8 = 24  
3 * 9 = 27
```

In [1]:

```
def hak():  
    sume = float(0.0)  
    q = int(input('몇개의 강의를 들으셨나요? '))  
    for i in range(q):  
        w = input('학점을 입력해주세요: ')  
        if w.upper() == 'A+':  
            w = float(4.5)  
            sume += w  
        elif w.upper() == 'A':  
            w = float(4.0)  
            sume += w  
        elif w.upper() == 'B+':  
            w = float(3.5)  
            sume += w  
        elif w.upper() == 'B':  
            w = float(3.0)  
            sume += w  
        elif w.upper() == 'C+':  
            w = float(2.5)  
            sume += w  
        elif w.upper() == 'C':  
            w = float(2.0)  
            sume += w  
        elif w.upper() == 'D+':  
            w = float(1.5)  
            sume += w  
        elif w.upper() == 'D':  
            w = float(1.0)  
            sume += w  
        elif w.upper() == 'F':  
            w = float(0.0)  
            sume += w  
    print(float(sume/q))
```

In [2]:

```
hak()
```

몇개의 강의를 들으셨나요? 6

학점을 입력해주세요: A

4.0

학점을 입력해주세요: A+

8.5

학점을 입력해주세요: A=

13.0

학점을 입력해주세요: A=

16.5

학점을 입력해주세요: A+

21.0

학점을 입력해주세요: A+

25.5

25.5

6

4.25

In [48]:

```
26.5/6
```

Out [48]:

```
4.416666666666667
```

In [51]:

```
4.5 + 4.5 + 4.5 + 4.5 + 4.5 + 4
```

Out [51]:

```
26.5
```

파일 읽고 쓰기

파일 읽기

```
f = open('파일이름.확장자', 'r', encoding='utf-8-sig')
```

```
data = f.read() or f.readlines()
```

```
f.close()
```

```
with open('파일이름.확장자', 'r', encoding='utf-8-sig') as f:
```

```
    f.read() or f.readlines()
```

파일 쓰기

```
f = open('파일이름.확장자', 'w', encoding='utf-8-sig')  
f.write()  
f.close()
```

```
with open('파일이름.확장자', 'w', encoding='utf-8-sig') as f:
```

```
    f.write()
```

In []: