

2. 객체, 변수, 자료형

1. 객체(object)와 변수(variable)
2. 함수 기초 이해하기
3. 파이썬의 아홉 가지 자료형
4. mutable 자료형 vs. immutable 자료형
5. 자료형 변환 함수
6. 주석 처리
7. 정리

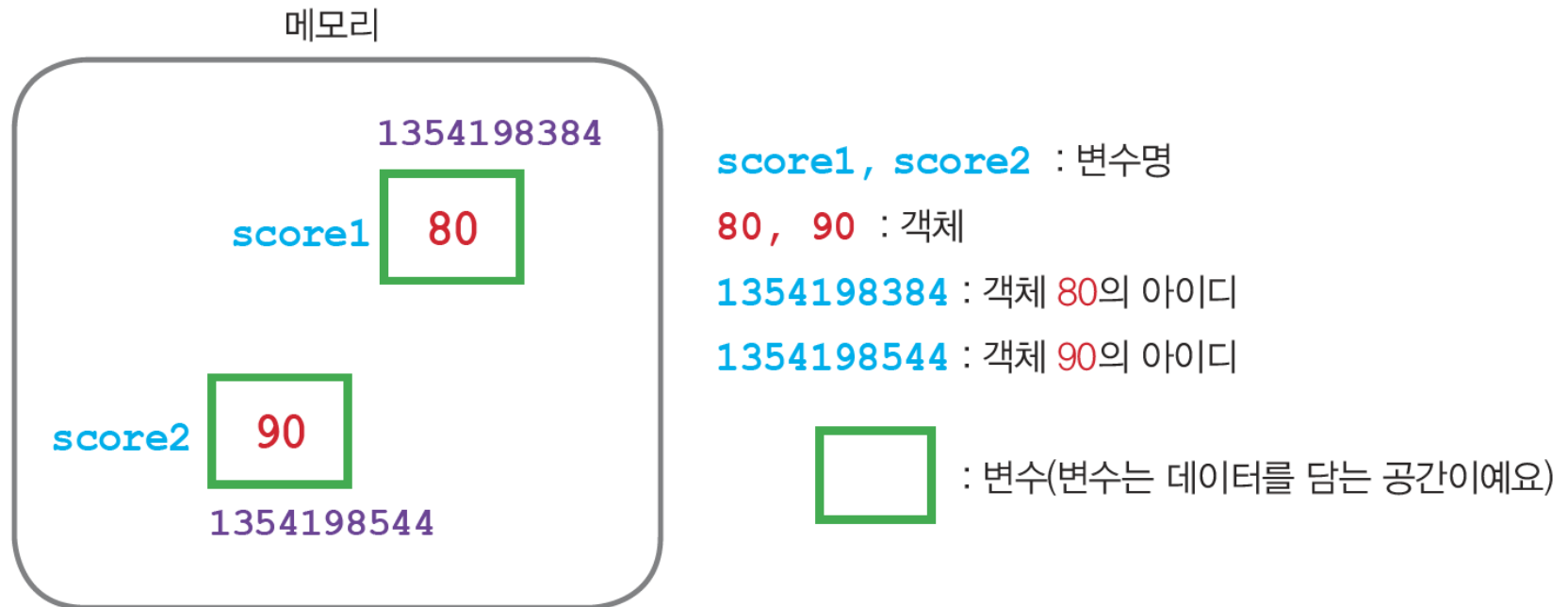
1. 객체(object)와 변수(variable)

◆ 객체, 변수, 변수명

- 컴퓨터는 어떤 일을 할 때, 그 일에 필요한 모든 데이터를 컴퓨터 메모리 위에 저장해 놓고 처리함.
- 예를 들어, 100명의 성적 처리를 하려면 100개의 성적을 메모리에 저장해야 함. 즉, 100개의 성적 데이터가 각각 공간을 가져야 함.
- 객체(object) - 메모리에 저장된 데이터 자체
- 변수(variable) - 객체를 담는 공간
- 변수명(variable name) - 변수의 이름

1. 객체(object)와 변수(variable)

◆ 변수 만들기



변수명 = 값



우변에 있는 값을 변수에 넣으시오!!

1. 객체(object)와 변수(variable)

◆ 변수 만들기

```
>>> a = 80
```

80이라는 객체를 만들어서 변수 a에 저장하시오

```
>>> b = 90
```

90이라는 객체를 만들어서 변수 b에 저장하시오

```
>>> 80 = 90
```

```
SyntaxError: can't assign to literal
```

```
>>> 90 = a
```

```
SyntaxError: can't assign to literal
```

기호 '=' 좌변에는 변수를 적고, 우변에는 값을 적어야 합니다.

1. 객체(object)와 변수(variable)


◆ 변수 만들기

- '='의 우변에 계산식이 올 수도 있다.

```
>>> x = 234 + 789 239 + 789의 결과를 변수 x에 저장하시오
>>> y = a + b      변수 a와 b를 더한 결과를 변수 y에 저장하시오
```

- 세미콜론(;)을 이용하여 한 줄에 여러 개의 변수 만들기

```
>>> a = 10
>>> b = 20
>>> c = 30
>>> print(a, b, c)
10 20 30
```



```
>>> a = 10; b = 20; c = 30
>>> print(a, b, c)
10 20 30
```

1. 객체(object)와 변수(variable)

◆ 변수명 만들기

- ① 영문자 대소문자를 사용할 수 있고, 대문자와 소문자는 다른 문자로 취급함.
- ② 숫자를 사용할 수 있음.
- ③ 특수문자는 '_'만 사용할 수 있음.
- ④ 숫자로 시작하면 안 됨.
- ⑤ 한글로도 변수명을 만들 수 있음.
- ⑥ 파이썬 키워드는 변수명으로 사용할 수 없음.

1. 객체(object)와 변수(variable)

◆ 변수명 예제

- 아래에서 score-1과 5data는 잘못된 변수명이다.

number_of_students	score-1	학생수
DaTa	5data	hello
		numberOfStudents

- 여러 단어를 합해서 하나의 변수명으로 사용하는 경우에는 위의 예에서 number_of_students처럼 단어 사이에 '_'을 이용하여 연결할 것을 권장함. (PEP-8)

1. 객체(object)와 변수(variable)

◆ 파이썬 키워드 목록

- 키워드(keyword)는 파이썬 언어에서 의미있는 단어들이.
- 키워드로 변수명을 만들 수 없음.
- 키워드 목록 확인하기

```
>>> import keyword
>>> keyword.kwlist
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del',
'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda',
'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```


1. 객체(object)와 변수(variable)

◆ 객체 출력하기

- 변수 안에 저장된 값을 확인해 보려면 print() 함수를 사용해야 함.
- 코마를 이용하면 여러 개의 변수 값들을 출력할 수 있음.

```
>>>a = 80; b = 90; x = 1023; y = 170
>>> print(a)          # 변수 a의 값, 즉, 변수 a의 객체를 출력합니다.
80
>>> print(b)          # 변수 b의 값을 출력합니다.
90
>>> print(x, y)       # print() 안에 코마를 이용해서 여러 변수를 넣을 수도 있습니다.
1023 170
```

1. 객체(object)와 변수(variable)

◆ IDLE에서 간단히 변수 출력하기

- IDLE에서는 프롬프트에서 변수명만 적어도 그 변수가 갖고 있는 객체의 값을 출력해 줌.

```
>>> m = 500
>>> m      # IDLE에서는 print() 함수 없이 변수명만 입력해도 변수값을 출력해 줍니다.
500
```

```
>>> a = 10; b = 20; c = 30
>>> print(a, b, c)
10 20 30
```

```
>>> a = 10; b = 20; c = 30
>>> a, b, c      # IDLE에서 변수를 콤마로 분리하면,
(10, 20, 30)     # 괄호로 묶어서 값들을 출력합니다.
```

1. 객체(object)와 변수(variable)

- ◆ 코드를 파일에 저장하여 실행할 때에는 반드시 print() 함수를 이용해서 출력해야 함.

코드 1

```
a = 10; b = 20 # 변수 a와 b 생성합니다.  
c = a + b      # c는 30을 저장합니다.  
d = a * b      # d는 200을 저장합니다.  
print(c)       # c 출력합니다.  
print(d)       # d 출력합니다.
```

결과 1

30

반드시 print() 써야 합니다.

코드 2

```
a = 10; b = 20  
c = a + b  
d = a * b  
c  
d
```

파일에 코드를 저장해서 수행시킬 때
변수명만 넣으면 아무것도 출력되지 않습니다.

결과 2

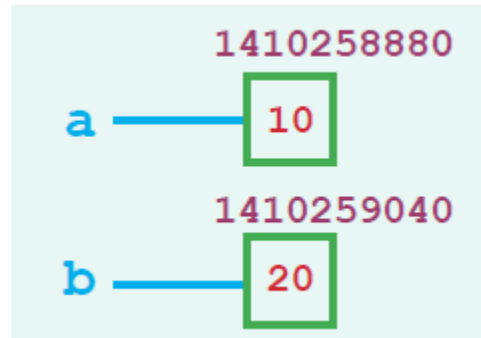
아무 것도 나오지 않아요.

1. 객체(object)와 변수(variable)

◆ id() 함수

- 객체를 만들면 id가 할당됨.
- 객체에 할당된 id를 보려면 id() 함수를 이용함.
- 두 객체가 동일한 객체인지를 판단할 필요가 있을 때 id() 함수를 이용함.

```
>>> a = 10
>>> id(a)
1410258880
>>> b = 20
>>> id(b)
1410259040
```



1. 객체(object)와 변수(variable)

◆ 변수 삭제하기 - del 키워드

- del 키워드를 이용하여 변수를 삭제함.

```
>>> x = 100
>>> print(x)
100
>>> del x
```

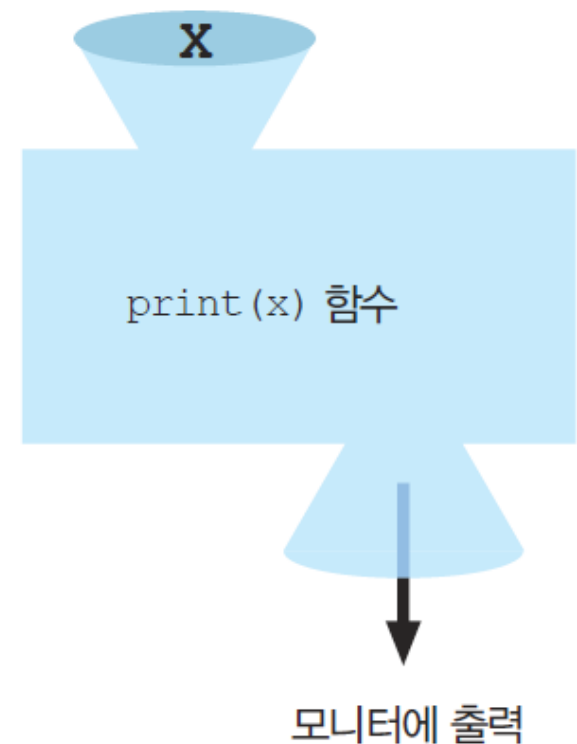
- 삭제한 변수를 사용하면, NameError 발생함.

```
>>> print(x)    # 없는 변수를 사용하려고 해서 NameError가 발생함.
.....
NameError: name 'x' is not defined
```

2. 함수 기초 이해하기

◆ 함수는 블랙박스 개념이다.

- `print()` 함수는 괄호 안에 내용을 모니터에 출력하라는 함수임.
- `print()`가 내부적으로 어떤 처리를 거쳐서 모니터 출력을 하는지 모르지만, 우리는 `print()` 함수를 이용하면 괄호 안에 내용이 모니터에 출력된다는 것을 알고 `print()` 함수를 사용함.



2. 함수 기초 이해하기

◆ 함수의 인수 (arguments)

- 모든 함수는 함수 이름 옆에 바로 괄호가 있어서, 그 괄호 안에 어떤 값을 넣어야 함. (넣을 내용이 없으면 빈 괄호로 둠)
- 함수의 괄호 안에 넣는 값을 ‘인수’라고 함.

```
>>> print('Hello world') # 'Hello world'가 인수임.  
Hello world  
>>> a = 10  
>>> print(a)          # a의 값인 10이 인수임.  
10
```

2. 함수 기초 이해하기

◆ 함수 abs()의 인수 (arguments)

- abs() 함수는 괄호에 넣는 수의 절대값을 결과로 내주는 함수임.

```
>>> y = abs(-5)
>>> print(y)
5
```

y = abs(-5) -5가 인수(argument)입니다.

① abs(-5)를 수행하면 5가 결과로 나옵니다.

② abs(-5) 자리에 반환값 5가 대체됩니다.

결과로 나온 값을
'반환값(return value)'라고
부릅니다.

y = ~~abs(-5)~~

5

③ 반환값 5가 y에 저장됩니다.

2. 함수 기초 이해하기

◆ print() 함수의 반환값

- abs() 함수는 절대값을 구한 후에 결과를 반환함.
- 모든 함수가 반환값을 갖는 것은 아님.
- print() 함수는 반환값이 없음. 반환값이 없을 때 결과를 변수를 받아서 출력하면 None을 출력함.

```
>>> y = print('hello world') # print('hello world') 함수의 결과를 변수 y에 넣습니다.  
hello world  
>>> print(y) # print() 함수의 반환값이 없어요. 반환값이 없을 때는 None이 나와요.  
None
```

3. 파이썬의 아홉 가지 자료형

◆ 자료형 (data type)

- 변수에 객체를 저장할 때, 컴퓨터는 객체가 어떤 유형의 객체인지를 판단함.
- 이를 '자료형 (data type)'이라고 함.

```
>>> a = 10      # 10은 정수임. 정수는 int로 표시됨.
```

```
>>> type(a)     # type() 함수는 괄호 안에 객체의 자료형을 알려줌.
```

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

```
>>> b = 3.5     # 3.5는 실수임. 실수는 float로 표시됨.
```

```
>>> type(b)
```

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

3. 파이썬의 아홉 가지 자료형

◆ 파이썬이 제공하는 자료형 (data type)

`int`(정수), `float`(실수), `complex`(복소수), `bool`(부울), `bytearray`, `bytes`
`str`(문자열), `list`(리스트), `tuple`(튜플), `set`(집합), `dict`(사전), `frozenset`

분류	자료형		예
수치 자료형 (3장)	정수 (<code>int</code>)		..., -3, -2, -1, 0, 1, 2, 3, ...
	실수 (<code>float</code>)		3.14, 5.5, 8.0, 0.54, -3.89, ...
	복소수 (<code>complex</code>)		3+4j, 5.7+2j, 2+9j, 5+1j, ...
부울 자료형 (6장)	부울 (<code>bool</code>)		True, False
군집 형태 자료형 (컨테이너 자료형)	시퀀스(Sequence) 자료형	문자열 (<code>str</code>) - 4장	'hello', "python", 'data', ...
		리스트 (<code>list</code>) - 9장	[1,2,3,4], ['red', 'blue'], [3.5, 2.4], ...
		튜플 (<code>tuple</code>) - 10장	(1,2,3,4), ('red', 'blue'), (3.4, 5.5, 1.2), ...
	집합 (<code>set</code>) - 11장		{1,2,3}, {'red', 'blue'}, {3.5, 1.2}, ...
	사전 (<code>dict</code>) - 12장		{1:'one', 2:'two'}, {'red':5, 'blue':2}, ...

3. 파이썬의 아홉 가지 자료형

◆ 파이썬이 제공하는 자료형 (data type)

1. 정수 (int) 자료형

```
>>> n = 50
>>> type(n)
<class 'int'>
```

type() 함수는 인수로 넣은 데이터의 자료형을 알려줍니다.

```
>>> type(50)    # 인수로 데이터를 바로 넣어도 됩니다.
<class 'int'>
```

2. 실수 (float) 자료형

```
>>> f = 3.7
>>> type(f)
<class 'float'>
```

3. 파이썬의 아홉 가지 자료형

◆ 파이썬이 제공하는 자료형 (data type)

3. 복소수 (complex) 자료형

```
>>> c = 2 + 5j  
>>> type(c)  
<class 'complex'>
```

4. 부울 (bool) 자료형

```
>>> b = True  
>>> type(b)  
<class 'bool'>
```

```
>>> d = False  
>>> type(d)  
<class 'bool'>
```

3. 파이썬의 아홉 가지 자료형

◆ 파이썬이 제공하는 자료형 (data type)

5. 문자열 (str) 자료형 - 따옴표로 문자들을 묶음.

```
>>> s1 = 'python'                # 작은따옴표
>>> s2 = "hello world"          # 큰따옴표
>>> s3 = '''study programming''' # 작은따옴표 세 개
>>> s4 = """python is good"""   # 큰따옴표 세 개
>>> type(s1), type(s2), type(s3), type(s4) # 콤마로 분리하면 괄호로 묶어서 출력합니다.
(<class 'str'>, <class 'str'>, <class 'str'>, <class 'str'>)
```

6. 리스트 (list) 자료형 - []로 데이터들을 묶음.

```
>>> score = [80, 90, 77, 95, 89]
>>> type(score)
<class 'list'>
```

3. 파이썬의 아홉 가지 자료형

◆ 파이썬이 제공하는 자료형 (data type)

7. 튜플 (tuple) 자료형 - ()로 데이터들을 묶음.

```
>>> T = (1,3,5,7)
>>> type(T)
<class 'tuple'>
```

8. 집합 (set) 자료형 - 집합 기호({})로 데이터들을 묶음.

```
>>> data = {1,3,4,6,7}
>>> type(data)
<class 'set'>
```

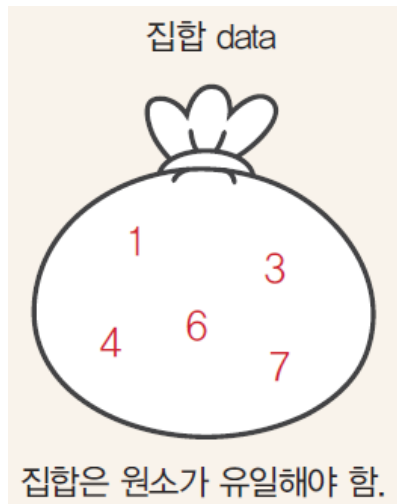
3. 파이썬의 아홉 가지 자료형

◆ 파이썬이 제공하는 자료형 (data type)

9. 사전 (dict) 자료형 - 집합 기호({})를 이용하는데, 반드시 원소는 **키:값**의 쌍으로 저장되어야 함.

```
>>> school = {1:200, 2:220, 3:170}
>>> type(school)
<class 'dict'>
```

〈 집합과 사전 〉



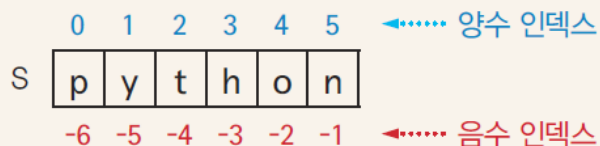
3. 파이썬의 아홉 가지 자료형

◆ 시퀀스 자료형 (sequence data types)

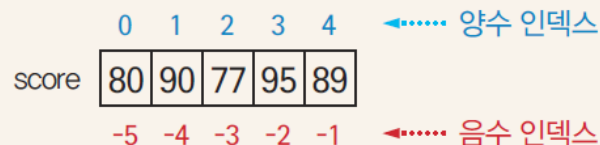
참고

시퀀스(sequence) 자료형은 순서가 있는 자료형으로 문자열, 리스트, 튜플이 여기에 속해요. 순서가 있기 때문에 순서대로 저장되고, 저장되는 공간에 인덱스가 붙게 됩니다. 아래 그림을 보세요. 인덱스 개념은 나중에 자세히 나오는데, 지금은 시퀀스 자료형에는 인덱스가 있다는 것을 알아 두세요.

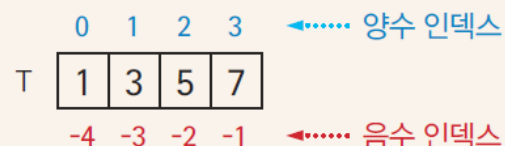
```
>>> s = 'python'
>>> print(s[0])
p
```



```
score = [80, 90, 77, 95, 89]
>>> print(score[3])
95
```

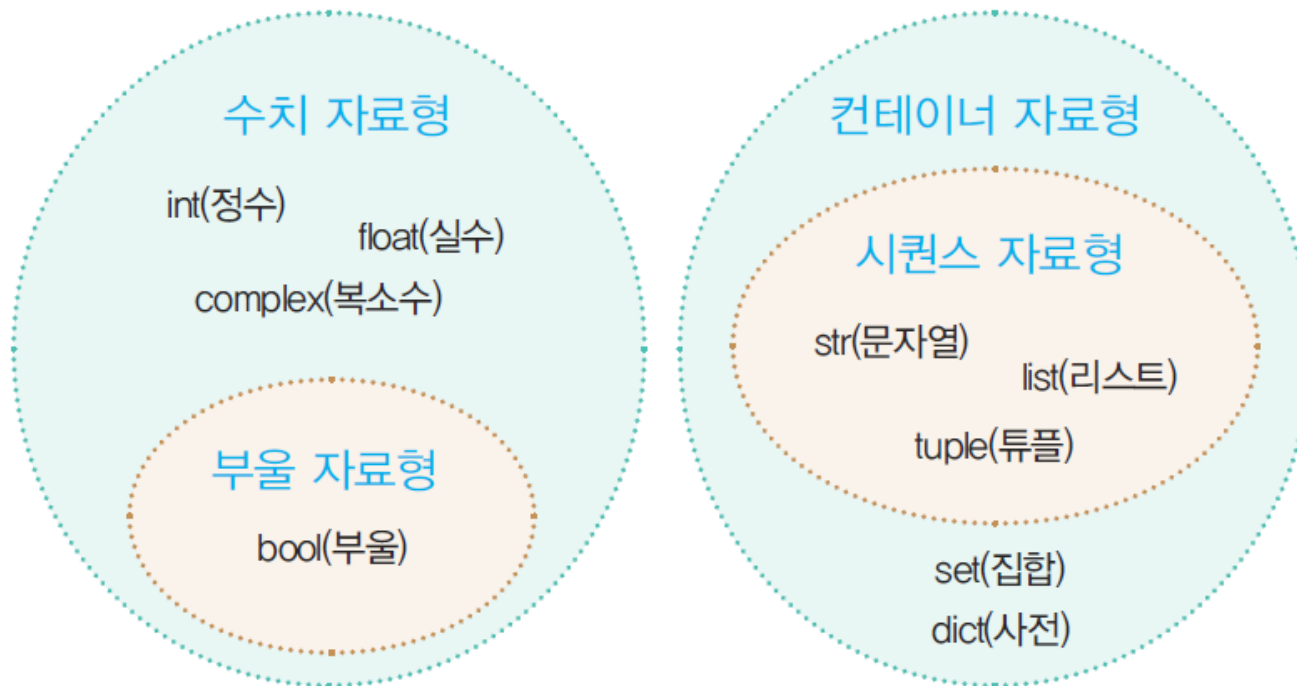


```
T = (1,3,5,7)
>>> print(T[-2])
5
```



3. 파이썬의 아홉 가지 자료형

◆ 수치 자료형과 컨테이너(container) 자료형



4. mutable 자료형 vs immutable 자료형

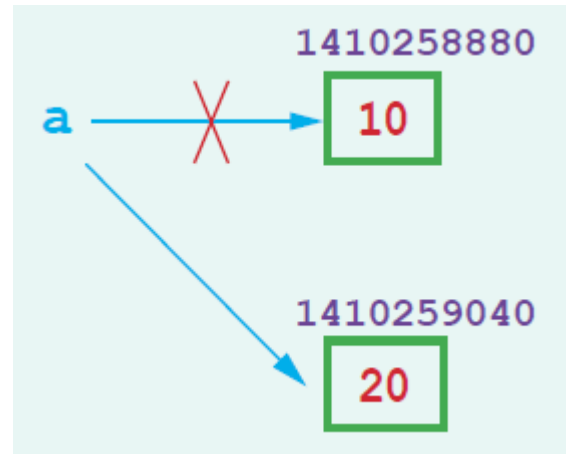
- ◆ mutable 자료형 - 변경 가능한 자료형
- ◆ immutable 자료형 - 변경 불가능한 자료형

mutable 자료형	리스트, 집합, 사전
immutable 자료형	정수, 실수, 복소수, 부울, 문자열, 튜플

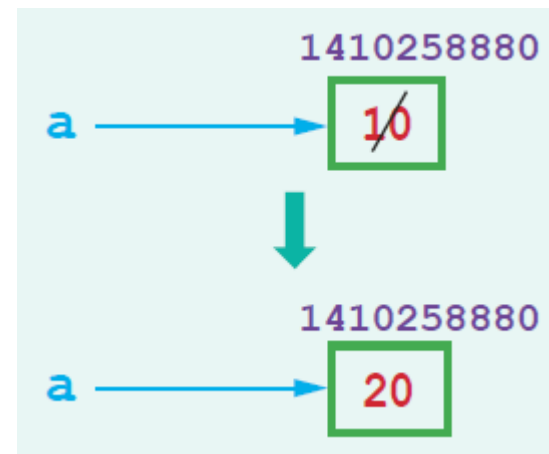
4. mutable 자료형 vs immutable 자료형

◆ immutable 한 정수 자료형

```
>>> a = 10  
>>> id(a)  
1410258880  
>>> a = 20  
>>> id(a)  
1410259040
```



- 만약에 정수 자료형이 mutable하다면 오른쪽과 같이 `a`가 가리키는 공간의 값이 바뀌게 됨.



4. mutable 자료형 vs immutable 자료형

◆ 리스트 (mutable)와 튜플 (immutable) 비교

리스트	튜플
<pre>>>> math_score = [80, 90, 85, 70] >>> math_score[2] = 88 >>> print(math_score) [80, 90, 88, 70]</pre>	<pre>>>> english_score = (95, 82, 90, 80) >>> english_score[1] = 85 TypeError: 'tuple' object does not support item assignment</pre>

5. 자료형 변환 함수

함수명	의미
<code>int(x)</code>	인수 x의 자료형을 <code>int</code> 로 변환한 객체를 반환합니다.
<code>float(x)</code>	인수 x의 자료형을 <code>float</code> 로 변환한 객체를 반환합니다.
<code>complex(x)</code>	인수 x의 자료형을 <code>complex</code> 로 변환한 객체를 반환합니다.
<code>bool(x)</code>	인수 x의 자료형을 <code>bool</code> 로 변환한 객체를 반환합니다.
<code>str(x)</code>	인수 x의 자료형을 <code>str</code> 로 변환한 객체를 반환합니다.
<code>list(x)</code>	인수 x의 자료형을 <code>list</code> 로 변환한 객체를 반환합니다.
<code>tuple(x)</code>	인수 x의 자료형을 <code>tuple</code> 로 변환한 객체를 반환합니다.
<code>set(x)</code>	인수 x의 자료형을 <code>set</code> 으로 변환한 객체를 반환합니다.
<code>dict(x)</code>	인수 x의 자료형을 <code>dict</code> 로 변환한 객체를 반환합니다.

5. 자료형 변환 함수

실수를 정수로 변환하기 (int() 함수)

```
>>> x = 3.2
>>> y = int(x)
>>> print(y)
3
>>> type(y)
<class 'int'>
```

정수를 실수로 변환하기 (float() 함수)

```
>>> x = 5
>>> y = float(x)
>>> print(y)
5.0
>>> type(y)
<class 'float'>
```

자료형과 똑같은 이름으로 내장 함수가 있습니다.
int 자료형은 int() 함수가 있습니다.
float 자료형은 float 함수가 있습니다.



○ = int (↑)

int가 아닌 자료형을 넣고 int 형으로 바꾸어 줍니다.

6. 주식 처리

◆ 파이썬 주식

- 한 줄 주식은 # 기호를 사용함. (# 이후가 주식 내용임)
- 여러 줄 주식은 작은 따옴표 세 개(““ ... ””) 또는 큰 따옴표 세 개(“““ .. ”””) 사용함.

```
'''
```

```
    작성자 : 홍길동
```

```
    작성일 : 2017년 12월 11일
```

```
'''
```

←..... 여러 줄 주식

```
x = -100
```

```
y = abs(x)
```

```
print(x, y)
```

정수 x의 절대값을 계산하여 변수 y에 넣습니다.

주식

7. 정리

◆ 수치 자료형 vs. 컨테이너 자료형

수치 자료형	숫자 표현	int, float, complex
	참/거짓 표현	bool
컨테이너 자료형	시퀀스 자료형	str, list, tuple
	순서 없는 자료형	set, dict

◆ immutable 자료형 vs. mutable 자료형

immutable 자료형	mutable 자료형
수치 자료형 – int, float, complex 부울 자료형 – bool 컨테이너 자료형 – str, tuple	컨테이너 자료형 – list, set, dict