세상의 속도를
따라잡고 싶다면

점프 투 파이썬

박응용 지음(위키독스 운영자)

이지스퍼블리싱

02

파이썬 프로그래밍의 기초, 자료형

02-1 숫자형

02-2 문자형 자료형

02-1 숫자형

숫자형(number)이란?

- 숫자 형태로 이루어진 자료형

항목	파이썬 사용 예
정수	123, -345, 0
실수	123.45, -1234.5, 3.4e10
8진수	0o34, 0o25
16진수	0x2A, 0xFF

02-1 숫자형

숫자형은 어떻게 만들고 사용할까?

정수형(integer)

- 정수를 뜻하는 자료형

```
>>> a = 123    <- 양의 정수 대입
>>> a = -178   <- 음의 정수 대입
>>> a = 0      <- 숫자 0 대입
```

실수형(floating-point)

- 소수점이 포함된 숫자

```
>>> a = 1.2
>>> a = -3.45
```

```
>>> a = 4.24E10    <- 4.24×10^10
>>> a = 4.24e-10   <- 4.24×10^-10
```

※ 컴퓨터식 지수 표현 방식

8진수(octal)

- 숫자 0 + 알파벳 소문자 o 또는 대문자 O

```
>>> a = 0o177
>>> print(a)
127  <- 1×8^2+7×8+7=127
```

16진수(hexadecimal)

- 숫자 0 + 알파벳 소문자 x

```
>>> a = 0x8ff
>>> b = 0xABC
>>> print(b)
2748  <- 10×16^2+11×16+12=2748
```

02-1 숫자형

숫자형을 활용하기 위한 연산자

사칙 연산

```
>>> a = 3
>>> b = 4
>>> a + b
7
>>> a - b
-1
>>> a * b
12
>>> a / b
0.75
```

x의 y제곱을 나타내는 ** 연산자

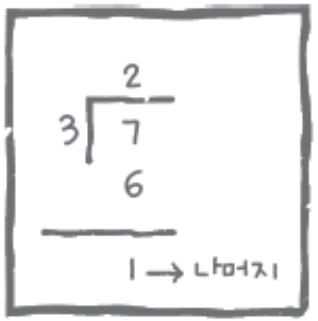
```
>>> a = 3
>>> b = 4
>>> a ** b
81
```

02-1 숫자형

숫자형을 활용하기 위한 연산자

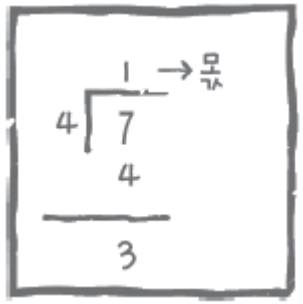
나눗셈 후 나머지를 리턴하는 % 연산자

```
>>> 7 % 3
1
>>> 3 % 7
3
```



나눗셈 후 몫을 리턴하는 // 연산자

```
>>> 7 // 4
1
```



02-2 문자열 자료형

- **문자열(string)이란?**
  - 문자, 단어 등으로 구성된 문자들의 집합

```
"Life is too short, You need Python"
"a"
"123"
```

02-2 문자열 자료형

▪ 문자열은 어떻게 만들고 사용할까?

1. **큰따옴표(“)**

```
"Hello World"
```

2. **작은따옴표(‘)**

```
'Python is fun'
```

3. **큰따옴표 3개(“””)**

```
"""Life is too short, You need python"""
```

4. **작은따옴표 3개(‘’’)**

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

02-2 문자열 자료형

▪ 문자열 안에 작은따옴표나 큰따옴표를 포함시키고 싶을 때

1. 작은따옴표(')

- 큰따옴표(")로 둘러싸기

```
>>> food = "Python's favorite food is perl"
```

2. 큰따옴표(")

- 작은따옴표(')로 둘러싸기

```
>>> say = '"Python is very easy." he says.'
```

3. 역슬래시 사용하기

- 역슬래시(₩) 뒤의 작은따옴표(')나 큰따옴표(")는 문자열을 둘러싸는 기호의 의미가 아니라 문자 ('), (") 그 자체를 의미

```
>>> food = 'Python\'s favorite food is perl'
>>> say = "\"Python is very easy.\" he says."
```

02-2 문자열 자료형

▪ 여러 줄인 문자열을 변수에 대입하고 싶을 때

1. 이스케이프 코드 '₩n' 삽입

```
>>> multiline = "Life is too short\nYou need python"
```

2. 작은따옴표 3개(''')

```
>>> multiline = '''
... Life is too short
... You need python
... '''
```

작은따옴표 3개를 사용한 경우

3. 큰따옴표 3개(""")

```
>>> multiline = """
... Life is too short
... You need python
... """
```

큰따옴표 3개를 사용한 경우

02-2 문자열 자료형

▪ 문자열 연산하기

1. 문자열 더해서 연결하기

```
>>> head = "Python"
>>> tail = " is fun!"
>>> head + tail
'Python is fun!'
```

2. 문자열 곱하기

```
>>> a = "python"
>>> a * 2
'pythonpython'
```

3. 문자열 길이 구하기

- 파이썬 기본 내장 함수 len() 사용

```
>>> a = "Life is too short"
>>> len(a)
17
```

02-2 문자열 자료형

▪ 문자열 인덱싱과 슬라이싱

▪ 문자열 인덱싱(indexing)

- '가리킨다'는 의미
- a[번호]
  - 문자열 안의 특정 값을 뽑아 냄
  - 마이너스(-)
    - 문자열 뒤부터 셈

L	i	f	e		i	s		t	o	o		s	h	o	r	t	,		Y	o	u		n	e	e	d		P	y	t	h	o	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33

```
>>> a = "Life is too short, You need Python"
>>> a[0]
'L'
>>> a[12]
's'
>>> a[-1]
'n'
```

```
a[0]: 'L', a[1]: 'i', a[2]: 'f', a[3]: 'e', a[4]: ' ', …
```

"파이썬은 0부터 숫자를 센다."

02-2 문자열 자료형

▪ 문자열 인덱싱과 슬라이싱

▪ 문자열 슬라이싱(slicing)

- ‘잘라낸다’는 의미
- a[시작 번호:끝 번호]
  - 시작 번호부터 끝 번호까지의 문자를 뽑아 냄
  - 끝 번호에 해당하는 것은 포함하지 않음

L	i	f	e		i	s		t	o	o		s	h	o	r	t	,		Y	o	u		n	e	e	d		P	y	t	h	o	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33

```
>>> a = "Life is too short, You need Python"
>>> a[0:4]
'Life'
```

```
>>> a = "20230331Rainy"
>>> date = a[:8]
>>> weather = a[8:]
>>> date
'20230331'
>>> weather
'Rainy'
```

02-2 문자열 자료형

▪ 문자열 포매팅이란?

▪ 문자열 포매팅(formatting)

- 문자열 안에 어떤 값을 삽입하는 방법

1. 숫자 바로 대입
   - 문자열 포맷 코드 **%d**

```
>>> "I eat %d apples." % 3
'I eat 3 apples.'
```

2. 문자열 바로 대입
   - 문자열 포맷 코드 **%s**

```
>>> "I eat %s apples." % "five"
'I eat five apples.'
```

3. 숫자 값을 나타내는 변수로 대입

```
>>> number = 3
>>> "I eat %d apples." % number
'I eat 3 apples.'
```

4. 2개 이상의 값 넣기

```
>>> number = 10
>>> day = "three"
>>> "I ate %d apples. so I was sick for %s days." % (number, day)
'I ate 10 apples. so I was sick for three days.'
```

02-2 문자열 자료형

문자열 포맷 코드

문자열 포맷 코드의 종류

코드	설명
%s	문자열(string)
%c	문자 1개(character)
%d	정수(integer)
%f	부동소수(floating-point)
%o	8진수
%x	16진수
%%	Literal %(문자 % 자체)

```
>>> "I have %s apples" % 3
'I have 3 apples'
>>> "rate is %s" % 3.234
'rate is 3.234'
```

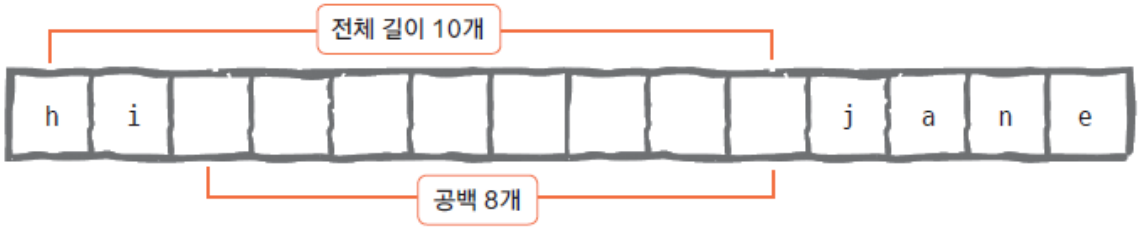
02-2 문자열 자료형

▪ 포맷 코드와 숫자 함께 사용하기

1. 정렬과 공백
   - %s를 숫자와 함께 사용하면, 공백과 정렬 표현 가능

```
>>> "%10s" % "hi"
'        hi'      ← hi가 오른쪽 정렬됨.
```


```
>>> "%-10sjane." % 'hi'
'hi        jane.' ← hi가 왼쪽 정렬됨.
```



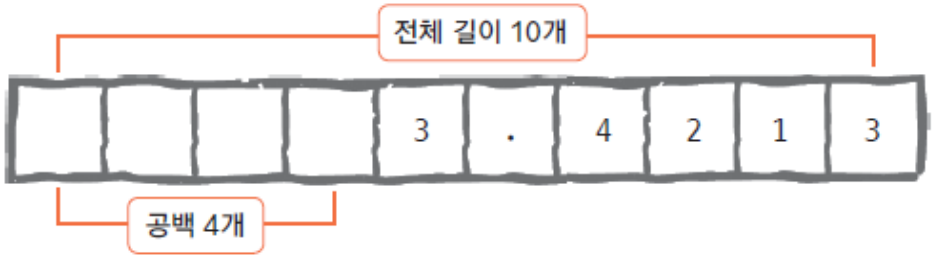
02-2 문자열 자료형

▪ 포맷 코드와 숫자 함께 사용하기

2. 소수점 표현하기
   - %f를 숫자와 함께 사용하면, 소수점 뒤에 나올 숫자의 개수 조절 및 정렬 가능

```
>>> "%0.4f" % 3.42134234
'3.4213'
```

```
>>> "%10.4f" % 3.42134234
'    3.4213'
```



02-2 문자열 자료형

format 함수를 사용한 포매팅

숫자 바로 대입하기

```
>>> "I eat {0} apples".format(3)
'I eat 3 apples'
```

문자열 바로 대입하기

```
>>> "I eat {0} apples".format("five")
'I eat five apples'
```

숫자 값을 가진 변수로 대입하기

```
>>> number = 3
>>> "I eat {0} apples".format(number)
'I eat 3 apples'
```

02-2 문자열 자료형

format 함수를 사용한 포매팅

2개 이상의 값 넣기

```
>>> number = 10
>>> day = "three"
>>> "I ate {0} apples. so I was sick for {1} days.".format(number, day)
'I ate 10 apples. so I was sick for three days.'
```

이름으로 넣기

```
>>> "I ate {number} apples. so I was sick for {day} days.".format(number=10, day=3)
'I ate 10 apples. so I was sick for 3 days.'
```

02-2 문자열 자료형

format 함수를 사용한 포매팅

인덱스와 이름을 혼용해서 넣기

```
>>> "I ate {0} apples. so I was sick for {day} days.".format(10, day=3)
'I ate 10 apples. so I was sick for 3 days.'
```

왼쪽 정렬

```
>>> "{0:<10}".format("hi")
'hi        '
```

오른쪽 정렬

```
>>> "{0:>10}".format("hi")
'        hi'
```

가운데 정렬

```
>>> "{0:^10}".format("hi")
'    hi    '
```

02-2 문자열 자료형

format 함수를 사용한 포매팅

공백 채우기

```
>>> "{0:=^10}".format("hi")
'====hi===='
>>> "{0:!<10}".format("hi")
'hi!!!!!!!!'
```

소수점 표현하기

```
>>> y = 3.42134234
>>> "{0:0.4f}".format(y)
'3.4213'
```

```
>>> "{0:10.4f}".format(y)
'    3.4213'
```

{ 또는 } 문자 표현하기

```
>>> "{{ and }}".format()
'{ and }'
```

02-2 문자열 자료형

f 문자열 포매팅

- 파이썬 3.6 버전부터 f 문자열 포매팅 기능 제공
- 문자열 앞에 f 접두사를 붙이면 f 문자열 포매팅 기능 사용 가능

```
>>> name = '홍길동'
>>> age = 30
>>> f'나의 이름은 {name}입니다. 나이는 {age}입니다.'
'나의 이름은 홍길동입니다. 나이는 30입니다.'
```

```
>>> age = 30
>>> f'나는 내년이면 {age + 1}살이 된다.'
'나는 내년이면 31살이 된다.'
```

02-2 문자열 자료형

f 문자열 포매팅

딕셔너리에서 사용

```
>>> d = {'name':'홍길동', 'age':30}
>>> f'나의 이름은 {d["name"]}입니다. 나이는 {d["age"]}입니다.'
'나의 이름은 홍길동입니다. 나이는 30입니다.'
```

정렬

```
>>> f'{"hi":<10}' ← 왼쪽 정렬
'hi        '
>>> f'{"hi":>10}' ← 오른쪽 정렬
'        hi'
>>> f'{"hi":^10}' ← 가운데 정렬
'    hi    '
```

{} 문자 그대로 표시

```
>>> f'{{ and }}'
'{ and }'
```

02-2 문자열 자료형

f 문자열 포매팅

공백 채우기

```
>>> f'{"hi":=^10}'  ← 가운데 정렬하고 '='로 공백 채우기
'====hi===='
>>> f'{"hi":!<10}'  ← 왼쪽 정렬하고 '!'로 공백 채우기
'hi!!!!!!!!'
```

소수점 표현

```
>>> y = 3.42134234
>>> f'{y:0.4f}'     ← 소수점 4자리까지만 표현
'3.4213'
>>> f'{y:10.4f}'    ← 소수점 4자리까지 표현하고 총 자릿수를 '10'으로 맞춤.
'    3.4213'
```

02-2 문자열 자료형

문자열 관련 함수들

- 문자열 자료형이 가진 내장 함수
- 문자 개수 세기 - **count**

```
>>> a = "hobby"
>>> a.count('b')
2
```

- 문자열 삽입 - **join**

```
>>> ",".join('abcd')
'a,b,c,d'
```

02-2 문자열 자료형

▪ 문자열 관련 함수들

- 위치 알려 주기 1 – **find**
  - 찾는 문자열이 처음 나온 위치 반환
  - 없으면 -1 반환

```
>>> a = "Python is the best choice"
>>> a.find('b')
14    <-- 문자열에서 b가 처음 나온 위치
>>> a.find('k')
-1
```

- 위치 알려 주기 2 – **index**
  - find와 마찬가지로,
    찾는 문자열이 처음 나온 위치 반환
  - 단, 찾는 문자열이 없으면 오류 발생

```
>>> a = "Life is too short"
>>> a.index('t')
8
>>> a.index('k')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: substring not found
```

k가 없으므로 오류 발생

02-2 문자열 자료형

▪ 문자열 관련 함수들

- 소문자를 대문자로 바꾸기 – **upper**

```
>>> a = "hi"
>>> a.upper()
'HI'
```

- 대문자를 소문자로 바꾸기 – **lower**

```
>>> a = "HI"
>>> a.lower()
'hi'
```

- 왼쪽 공백 지우기 - **lstrip**

```
>>> a = " hi "
>>> a.lstrip()
'hi '
```

- 오른쪽 공백 지우기 - **rstrip**

```
>>> a= " hi "
>>> a.rstrip()
' hi'
```

02-2 문자열 자료형

▪ 문자열 관련 함수들

- 양쪽 공백 지우기 – **strip**

```
>>> a = " hi "
>>> a.strip()
'hi'
```

- 문자열 바꾸기 – **replace**
  - replace(바뀔_문자열, 바꿀_문자열)

```
>>> a = "Life is too short"
>>> a.replace("Life", "Your leg")
'Your leg is too short'
```

- 문자열 나누기 – **split**
  - 공백 또는 특정 문자열을 구분자로 해서 문자열 분리
  - 분리된 문자열은 리스트로 반환됨

```
>>> a = "Life is too short"
>>> a.split()     <- 공백을 기준으로 문자열 나눔.
['Life', 'is', 'too', 'short']
>>> b = "a:b:c:d"
>>> b.split(':') <- :를 기준으로 문자열 나눔.
['a', 'b', 'c', 'd']
```

감사합니다.