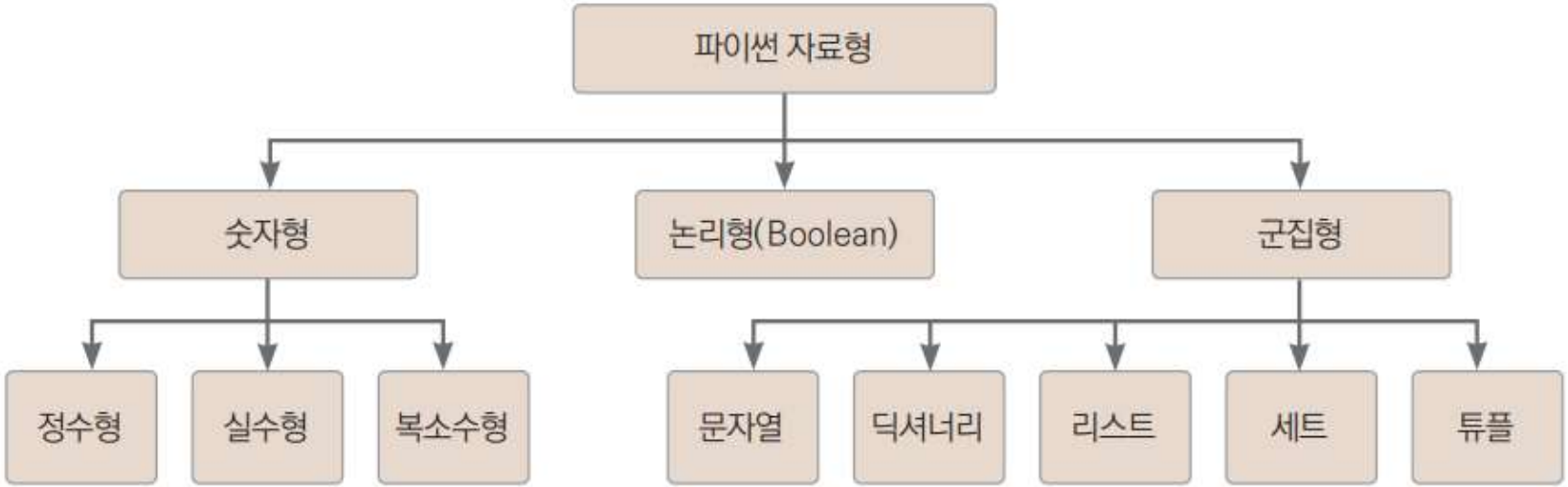
파이썬 자료형 - 숫자

김지성 강사

파이썬 자료형

✓ 파이썬의 자료형에는 크게 숫자/논리/군집으로 나눌 수 있음

▼ 파이썬 자료형



파이썬 자료형 - 숫자형

- ✓ 정수형/실수형/복소수형이 존재한다.
- ✓ 정수
  - ○ 정수형은 **양의 정수, 0, 음의 정수**를 의미
  - ○ 정수형은 파이썬에서 <**int**>로 표현된다.
  - ○ 파이썬에서의 정수는 비트 수에 제한되지 않기 때문에 메모리의 한계까지 확장 가능
- ✓ 실수
  - ○ 실수형은 **소수점이 포함된 숫자**를 의미하고 <**float**>으로 표현된다.
  - ○ 부동 소수점 숫자는 **소수점 이하 15자리**까지 정확하게 표현
  - ○ 1은 정수, 1.0은 실수

파이썬 자료형 - 숫자형

✓ 정수형 숫자 실습

```
# 정수형 : class 'int'
import sys

number1 = 127
number2 = 0
number3 = -128
print(number1, type(number1))
print(number2, type(number2))
print(number3, type(number3))


max = sys.maxsize
min = -max-1
print("최대값 : ",max, type(max))
print("최소값 : ",min, type(min))

x = 10000000000000000000000000000000000000000000000
x = x + 1
print(x)
```

파이썬 자료형 - 숫자형

✓ 실수형 숫자 실습

```
# 실수형 : class 'float'
float_number1 = 123.456
float_number2 = -123.456
print(float_number1, type(float_number1))
print(float_number2, type(float_number2))

float_number3 = 123.e3
float_number4 = -123.456e-3
print(float_number3, type(float_number3))
print(float_number4, type(float_number4))
```

파이썬 자료형 - 논리

김지성 강사

파이썬 자료형 - 논리형

- ✓ 파이썬 bool type의 리터럴은 **True**와 **False**가 있다.
- ✓ 대소문자를 구분하며 true / false 혹은 TRUE /FALSE 처럼 표기할 수 없고 반드시 첫 글자를 대문자로 하고 나머지는 소문자로 하는 True / False 로만 입력 가능하다.

True

참

False

거짓

```
type(True)
```

```
bool
```

```
type(False)
```

```
bool
```

```
type(true)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_15216\1822790664.py in <module>
----> 1 type(true)

NameError: name 'true' is not defined
```

```
type(FALSE)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_15216\3505605817.py in <module>
----> 1 type(FALSE)

NameError: name 'FALSE' is not defined
```

파이썬 자료형 - 논리형

✓ 아래와 같이 변수명으로 true/false를 사용하면 헷갈릴 수 있으므로 사용은 지양한다.

```
true = '참입니다.'
print(true)

false = 0
print(false)
```

파이썬 자료형 - 문자열

김지성 강사

파이썬 자료형 - 문자열

- ✓ 문자열(String)은 문자와 단어 등으로 구성된 문자들의 집합을 의미
- ✓ Python의 문자열은 유니 코드 문자를 나타내는 byte 배열이다.
  - ○ 단일 문자는 길이가 1인 문자열
  - ○ 파이썬에서의 유니코드는 1문자당 1,2,4바이트를 할당

```
import sys
print("\'\' : ",sys.getsizeof(''))

print("a : ",sys.getsizeof('a'))
print("aa : ",sys.getsizeof('aa'))

print("あ : ",sys.getsizeof('あ'))
print("ああ : ",sys.getsizeof('ああ'))

print("가 : ",sys.getsizeof('가'))
print("가가 : ",sys.getsizeof('가가'))

print("ㄱ : ",sys.getsizeof('ㄱ'))
print("ㄱㄱ : ",sys.getsizeof('ㄱㄱ'))
```

파이썬 자료형 - 문자열

✓ 문자열(String)은 문자와 단어 등으로 구성된 문자들의 집합을 의미

✓ Python에서 문자열 변수를 만들기 위해서는 [' ',” ”]을 사용하여 사이에 문자열을 넣으면 된다.

```
# 파이선 문자열
str1 = 'Python String! 한글'
str2 = "Python String! 한글"
str3 = '''Python String! 한글'''
str4 = """Python String! 한글"""
print(str1, str2, str3, str4, sep="\n")

# 문자열 안에 따옴표 자체를 포함하는 방법
str5 = 'Python String! "한글"'
str6 = 'Python String! \'한글\''
str7 = "Python String! '한글'"
str8 = "Python String! \"한글\""
print(str5, str6, str7, str8, sep="\n")
```

파이썬 자료형 - 문자열

✓ 연습 문제1

✓ 다음과 사진과 같이 동일하게 출력되게끔 [' '," "]을 사용하여 문자열을 만들어보세요.

```
Python String! "한글"
Python String! '한글'
```

파이썬 자료형 - 문자열

✓ 연습 문제2

✓ 다음과 사진과 같이 동일하게 출력되게끔 [' '," "]을 사용하여 문자열을 만들어보세요.

```
Python String! 한글
문자열은 파이썬에서 아주 중요합니다.
여러 줄 문자열을 연습해보세요!
```

파이썬 자료형 - 문자열 메소드

✓ 문자열(String)은 다양한 처리 함수들을 가지고 있다.

✓ len(문자열)은 문자열의 길이를 구해주는 함수이다.

○ 한글, 공백, 특수문자도 1글자로 인식

```
str1 = "hello"
str2 = "안녕하세요."
print(len(str1))
print(len(str2))
```

파이썬 자료형 - 문자열

✓ 연습 문제1

✓ 아래 문자열의 길이를 구하세요.

```
str1 = "Hello, Python!"
str2 = "안녕하세요."

# TODO: 각 문자열의 길이를 출력하세요.
```

파이썬 자료형 - 문자열

✓ 연습 문제2

✓ 아래 문자열의 길이를 구하세요.

```
str1 = "Python Programming"
str2 = "    Leading and trailing spaces    "

# TODO: 각 문자열의 길이를 출력하세요.
```

파이썬 자료형 - 문자열 메소드

- ✓ 문자열(String)은 다양한 처리 함수들을 가지고 있다.
- ✓ 문자열을 서로 연결하거나 반복할 수 있다.
  - ○ “hello” + “world” = “hello world”
  - ○ “x” * 5 = “xxxxx”

```
print("hello" + "world")
print("x" * 5)
```

파이썬 자료형 - 문자열

✓ 연습 문제1

✓ 동일한 출력 결과가 나오게 문자열을 결합하세요.

```
str1 = "Hi"
str2 = "Bye"

# TODO: 주어진 조건에 따라 문자열을 출력하세요.
```

```
HiHiHi ByeByeBye
```

파이썬 자료형 - 문자열

✓ 연습 문제2

✓ 동일한 출력 결과가 나오게 문자열을 결합하세요.

```
str1 = "Programming"

# TODO: 문자열 'X'를 str1의 길이만큼 반복하여 출력하세요.
```

XXXXXXXXXXX

파이썬 자료형 - 문자열

✓ 연습 문제3

✓ 동일한 출력 결과가 나오게 문자열을 결합하세요.

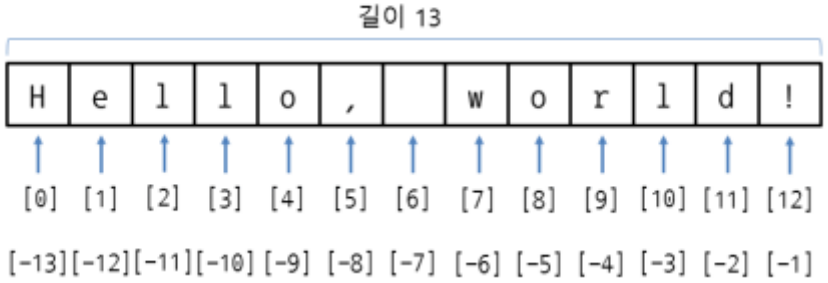
```
str1 = "AB"
str2 = "CD"

# TODO: 주어진 조건에 따라 문자열을 출력하세요.
```

ABABABCDCD

파이썬 자료형 - 문자열 슬라이싱

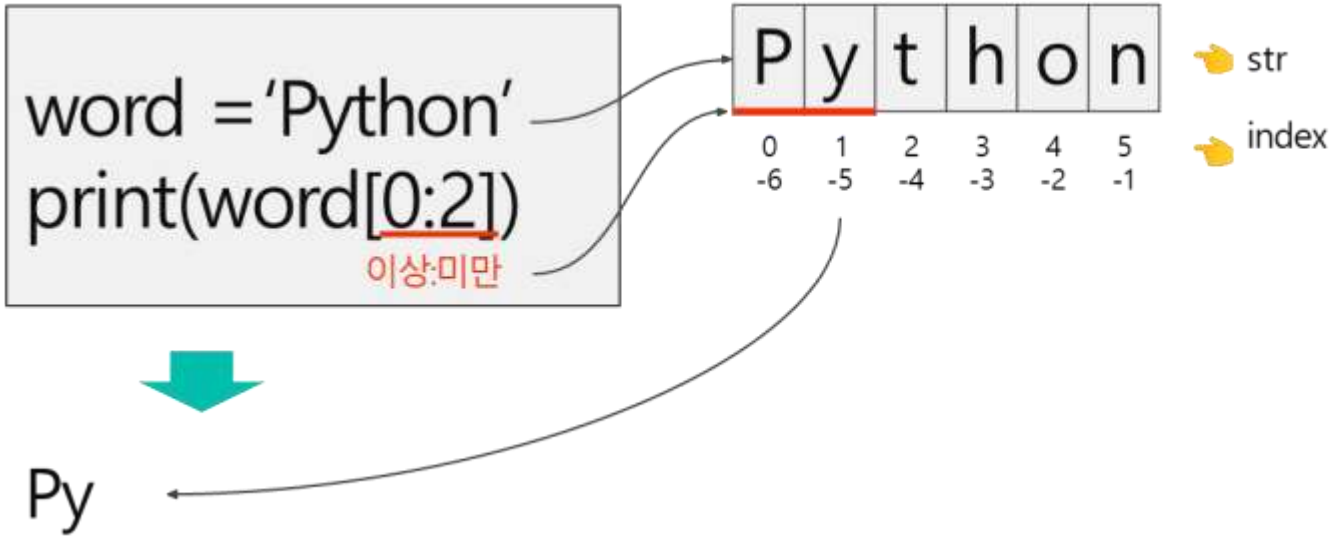
✓ 문자열을 잘라낼 수 있는데 이러한 것을 슬라이싱이라고 표현한다.

- [n] : n번째 글자 1글자 잘라내기
- [n:m] : n부터 m-1까지 문자열을 리턴
- [:m] : 앞의 숫자를 생략하면 0번째~m-1까지 글자를 잘라냄
- [n:] : 뒤의 숫자를 생략하면 n~끝까지 잘라냄
- [:] : 모두 생략하면 전체 글자 전부를 의미
- [-n] : 음수 값을 지정하면 뒤에서부터 접근하여 잘라냄



파이썬 자료형 - 문자열 슬라이싱

✓ 대괄호 안에 범위에서 [‘시작 인덱스’:’끝 인덱스’]로 슬라이싱을 진행




```
print(word[0:2])
print(word[:2])
print(word[:-4])
print(word[4:6])
print(word[4:])
print(word[-2:])
print(word[1:3])
print(word[-5:-3])
```

```
Py
Py
Py
on
on
on
yt
yt
```

파이썬 자료형 - 문자열 슬라이싱

✓ 문자열을 잘라낼 수 있는데 이러한 것을 슬라이싱이라고 표현한다.

```
# 파이선 문자열 슬라이싱
str1 = "Hello Python!"

# 문자열 잘라내기
print(str1[0], str1[6])  # 1글자 잘라내기
print(str1[3:7])  # [n:m] : n부터 m-1까지 문자열을 리턴합니다.
print(str1[:5])   # 앞의 숫자를 생략하면 0부터 입니다.
print(str1[6:])   # 뒤의 숫자를 생략하면 끝 글자까지 입니다.
print(str1[:])    # 모두 생략하면 전체입니다.
print(str1[:-2], end='\n')  # 음수 값을 지정하면 뒤에서부터 카운팅합니다.
print()
# 문자열 건너뛰기
print(str1[::2]) # 2칸씩 건너뛰기
print(str1[1:8:3]) # 3칸씩 건너뛰기
```

파이썬 자료형 - 문자열

✓ 연습 문제1

✓ 동일한 출력 결과를 보이게끔 만들어보세요.

```
text = "Hello, world!"

# TODO: 'Hello'라는 부분을 출력하세요.
```

```
Hello
```

파이썬 자료형 - 문자열

✓ 연습 문제2

✓ 동일한 출력 결과를 보이게끔 만들어보세요.

```
text = "Hello, world!"

# TODO: 마지막 두 글자를 출력하세요.
```

```
d!
```

파이썬 자료형 - 문자열

✓ 연습 문제3

✓ 동일한 출력 결과를 보이게끔 만들어보세요.

```
text = "Hello, world!"

# TODO: 다음과 같이 출력하세요.
```

```
Hlo ol!
```

파이썬 자료형 - 문자열

✓ 연습 문제4

✓ 동일한 출력 결과를 보이게끔 만들어보세요.

```
text = "Hello, world!"

# TODO: 문자열을 역순으로 출력하세요.
```

```
!dlrow ,olleH
```

파이썬 자료형 - 문자열

✓ 연습 문제5

✓ 동일한 출력 결과를 보이게끔 만들어보세요.

```
text = "Programming"

# TODO: 주어진 문자열의 첫 5글자를 뒤로 보내서 새로운 문자열을 만드세요.
```

```
ammingProgr
```

파이썬 자료형 - 문자열 메소드

✓ 문자열(String)은 다양한 처리 함수들을 가지고 있다.

✓ 특정 문자를 찾는 count 메소드도 존재한다.

○ count('찾을 문자열')

```
text = "Hello, world! Python is awesome. It is very good"

count_is = text.count("is")
print(count_is)
```

파이썬 자료형 - 문자열 메소드

✓ 문자열(String)은 다양한 처리 함수들을 가지고 있다.

✓ 문자열의 공백을 제거하는 메소드도 존재한다.

- ○ lstrip() : 왼쪽 공백 제거
- ○ rstrip() : 오른쪽 공백 제거
- ○ strip() : 앞, 뒤 공백을 모두 제거

```
# 앞뒤 공백 없애기
str1 = "  앞 뒤에 공백이 있습니다.  "
print('[', str1, ']',sep='')
print('[', str1.lstrip(), ']',sep='')
print('[', str1.rstrip(), ']',sep='')
print('[', str1.strip(), ']',sep='')
```

파이썬 자료형 - 문자열 메소드

- ✓ 문자열(String)은 다양한 처리 함수들을 가지고 있다.
- ✓ 문자열을 대, 소문자로 변경하는 간단한 메소드
  - ○ upper() : 문자열을 대문자로 변경
  - ○ lower() : 문자열을 소문자로 변경

```
# 파이썬 문자열
str1 = "Hello Python!"

# 대문자로 변환
print(str1.upper())
# 소문자로 변환
print(str1.lower())
```

파이썬 자료형 - 문자열 메소드

✓ 문자열(String)은 다양한 처리 함수들을 가지고 있다.

✓ 찾고자하는 문자열이 존재하면 find 혹은 index 메소드를 사용한다.

- ○ find(‘찾을 문자열’, ‘시작 위치’, ‘종료 위치’) : 찾고자하는 문자열이 없으면 -1을 리턴
- ○ index(‘찾을 문자열’, ‘시작 위치’, ‘종료 위치’) : 찾고자하는 문자열이 없으면 에러를 발생

```
# 파이선 문자열
str1 = "Hello Python!"

# find 메소드
print("find")
print(str1.find('h'))
print(str1.find('llo'))
print(str1.find('abc'))

# index 메소드
print("index")
print(str1.index('h'))
print(str1.index('llo'))
print(str1.index('abc'))
```

파이썬 자료형 - 문자열 메소드

✓ 문자열(String)은 다양한 처리 함수들을 가지고 있다.

✓ **python에서의 문자열은 불변 객체**이다. 따라서 **수정이 되는 것이 아닌 새로운 객체를 만들어 반환**한다.

○ replace('원본문자열', '바뀔문자열', 개수=None)

```
str1 = "Hello Python!"
str2 = str1.replace("Hello", "Hi")
print(str1,str2, sep='\n')
```

파이썬 자료형 - 문자열 메소드

✓ 다음과 같은 코드는 에러를 발생시킨다.

✓ **문자열은 기본적으로 불변(immutable)객체**이므로 할당된 **값을 수정할 수 없기 때문**

```
str1 = "Hello Python!"
str1[0] = 'A' # 에러 발생
```

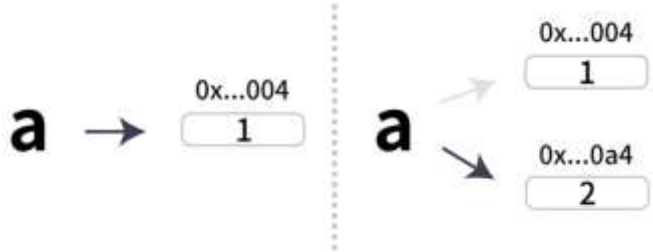
	데이터 타입
가변(mutable)	list, set, dict
불변(immutable)	int, float, bool, tuple, string, unicode

파이썬 자료형 - 문자열 메소드

✓ 가변 객체는 **메모리 안에 담겨 있는 값이 변할 수 있다**.

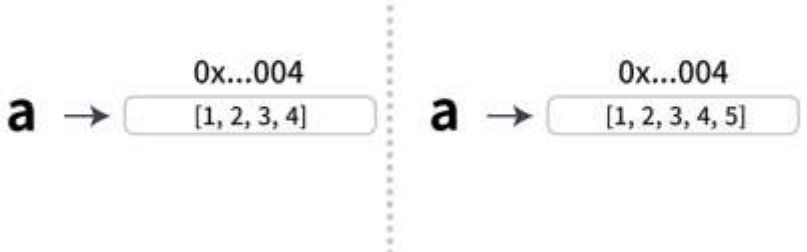
✓ 불변 객체는 **메모리 안에 담겨 있는 값이 변할 수 없다**.

불변



불변 객체이기 때문에 값이 변했을 때 메모리 주소가 변한다.

가변



가변 객체이기 때문에 값이 변했을 때 메모리 주소가 변하지 않는다.

파이썬 자료형 - 문자열 메소드

✓ 2개의 x의 **메모리 주소가 변경됨**을 알 수 있는데, 이는 기존에 “test1”의 값을 가지고 있는 메모리에다가 추가적으로 “test1 test2” 문자열을 만든 것이 아니라 **새로운 메모리에다가 “test1 test2”를 만든 것**을 볼 수 있다.

```
x = 'test1'
print(id(x))

x += 'test2'
print(id(x))
```

파이썬 자료형 - 문자열 메소드

- ✓ 문자열(String)은 다양한 처리 함수들을 가지고 있다.
- ✓ 문자열을 나누거나 합칠수도 있는데, 이 때 split, join 메소드를 사용한다.
  - ○ 문자열 나누기 : split(sep=구분자, maxsplit=분할횟수)
  - ○ 문자열 합치기 : join(iterable)

```
str1 = "a b c d e f, abcdef"
# 문자열 나누기
print(str1.split())
print(str1.split(','))

str1 = "a b c d e f abcdef"
# 문자열 합치기
print(','.join(str1))
```

파이썬 자료형 - 문자열 메소드

- ✓ 문자열(String)은 다양한 처리 함수들을 가지고 있다.
- ✓ 이외 변환 함수들
  - ○ ord : 문자의 아스키 코드값을 리턴하는 함수
  - ○ hex : hex(x)는 정수값을 입력받아 16진수(hexadecimal)로 변환하여 리턴하는 함수
  - ○ oct : oct(x)는 정수 형태의 숫자를 8진수 문자열로 바꾸어 리턴하는 함수
  - ○ chr : 아스키(ASCII) 코드값을 입력으로 받아 그 코드에 해당하는 문자를 리턴하는 함수
  - ○ capitalize : 단어의 첫글자만 대문자로 변환하는 함수

파이썬 자료형 - 문자열

✓ 연습 문제1

✓ 동일한 출력 결과를 보이게끔 만들어보세요.

```
text = "I love Python programming"

# TODO: 'Python' 단어를 제거한 문자열을 출력하세요.
```

```
I love programming
```

파이썬 자료형 - 문자열

✓ 연습 문제2

✓ 동일한 출력 결과를 보이게끔 만들어보세요.

```
text = "Replace all spaces with hyphens"

# TODO: 공백을 '-'로 대체한 문자열을 출력하세요.
```

```
Replace-all-spaces-with-hyphens
```

파이썬 자료형 - 문자열

✓ 연습 문제3

✓ 동일한 출력 결과를 보이게끔 만들어보세요.

```
text = "The more you practice, the better you become."

# TODO: 'the' 단어가 등장하는 횟수를 출력하세요.
```

```
the의 등장 횟수 :  2
```

파이썬 자료형 - 문자열

✓ 연습 문제4

✓ 동일한 출력 결과를 보이게끔 만들어보세요.

```
text = "Tomorrow is a new opportunity."

# TODO: 'o'의 첫 번째와 두 번째 등장 위치를 출력하세요.
```

```
첫 번째 o :  1
두 번째 o :  3
```