

Python Overview

Programming and Python

최성철 교수

Director of TEAMLAB

01100
00110



Python?

Python 이란?



- 1991년 귀도 반 로섬 (Guido Van Rossum) 발표
- 플랫폼 독립적
- 인터프리터 언어
- 객체 지향
- 동적 타이핑 언어
- 처음 C언어로 구현되었음

Guido Van Rossum

구도 반 로섬!!



- 1989년 크리스마스에
할 일이 없어 파이썬 개발
- “Monty Python’s Flying Circus”
→ 파이썬 이름의 유래
- 구글, DropBox 근무

Guido Van Rossum

Source: https://en.wikipedia.org/wiki/Guido_van_Rossum

구도 반 로섬!!



Hi Guido,

I came across your resume in a Google web search. **You seem to have an awesome expertise on Python.** I would be glad if you can reply my email and let me know your interest and availability.

.....
Our client immediately needs a PYTHON Developers at its location in *, NJ. Below are the job details. If interested and available, kindly fwd me your updated resume along with the expected rate and the availability.

Guido Van Rossum

Source: <https://goo.gl/GysBnU>

구도 반 로섬!!



Hi Guido,

I came across your resume in a Google web search. **You seem to have an awesome**

너 파이썬좀 짱인듯!

availability.

I'm not interested and not available.

expected rate and the availability.

Guido Van Rossum

Source: <https://goo.gl/GysBnU>

Python?



파이썬의 로고
두 마리의 뱀이 겹쳐 있음

Python의 특징

플랫폼(?)

독립적인(?)

인터프리터(?) 언어

Python의 특징

플랫폼 = OS

: 윈도우, 리눅스, 안드로이드, 맥OS, iOS 등 프로그램이
실행되는 운영 체제를 플랫폼이라고 함

독립적인 = 관계없는, 상관없는

: OS에 상관없이 한번 프로그램을 작성하면 사용 가능

인터프리터 = 통역기를 사용하는 언어...

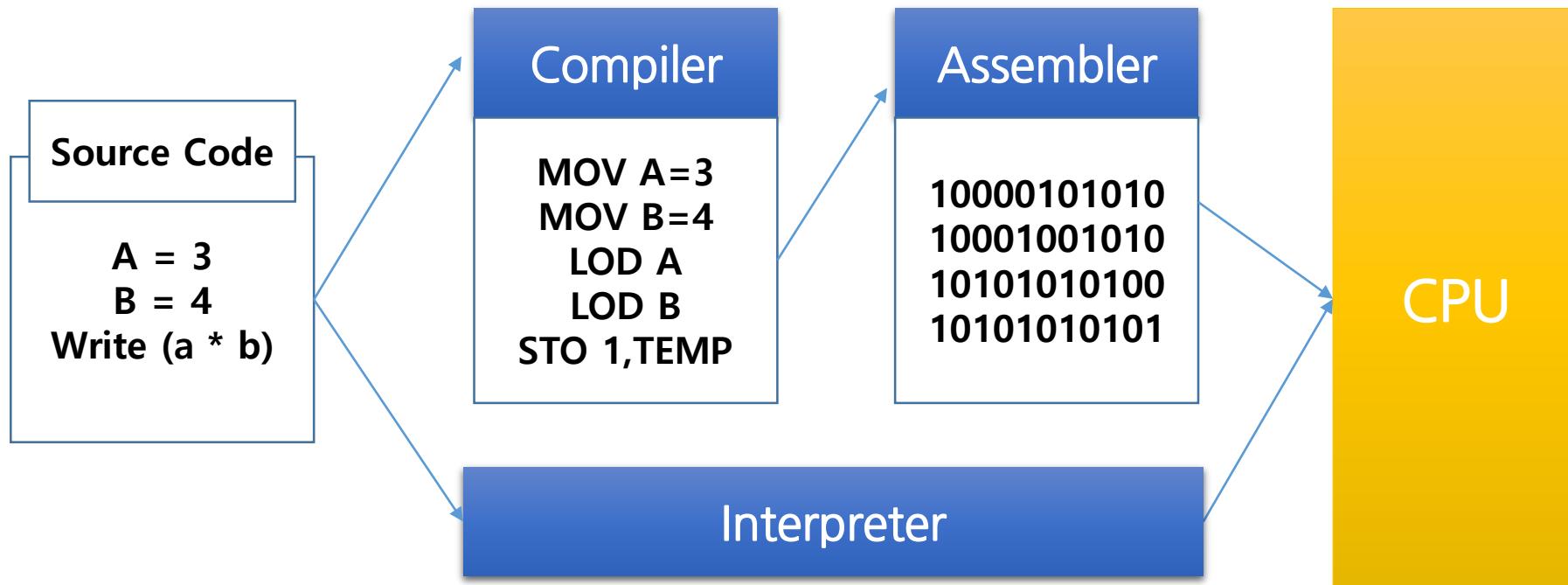
: 소스코드를 바로 실행할 수 있게 지원하는 프로그램 실행 방법

[컴파일러] vs [인터프리터]

컴파일러	인터프리터
<p>소스코드를 기계어로 먼저 번역 해당 플랫폼에 최적화되어 프로그램을 실행</p>	<p>별도의 번역과정 없이 소스코드를 실행시점에 해석하여 컴퓨터가 처리할 수 있도록 함</p>
<p>실행속도가 빠름 한번의 많은 기억장소 필요</p>	<p>간단히 작성, 메모리가 적게 필요 실행속도가 느림</p>
<p>C, 자바, C++, C#</p>	<p>파이썬, 스칼라</p>

[프로그램의 동작과정]

사람이 알 수 있는 고급언어를 기계만 알 수 있는 저급언어로 변환



※ 파이썬은 처음에 컴파일러 언어인 C로 작성되었다.
당연히 실행 시 Assembler와 같은 기계어 변환 과정을 거친다

Python의 특징

객체 지향적(?)

동적(?) 타이핑 언어

Python의 특징

객체지향적 언어

: 실행순서가 아닌 단위 모듈(객체) 중심으로 프로그램을 작성
하나의 객체는 어떤 목적을 달성하기 위한 행동(method)과
속성(attribute)을 가지고 있음

동적 타이핑 언어

: 프로그램이 실행하는 시점에 프로그램이 사용해야 할
데이터에 대한 타입을 결정함

Why Python?

하늘의 별과 같이 많은 언어들...



그들의 특징



C



Java

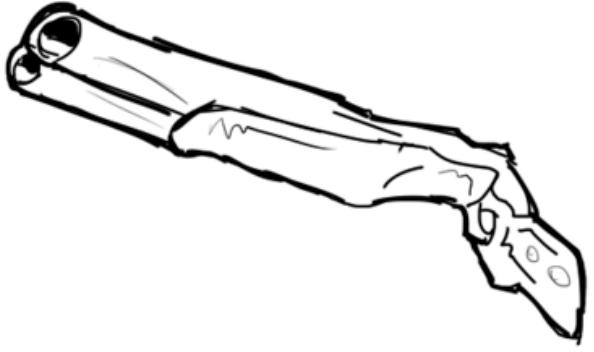


C++

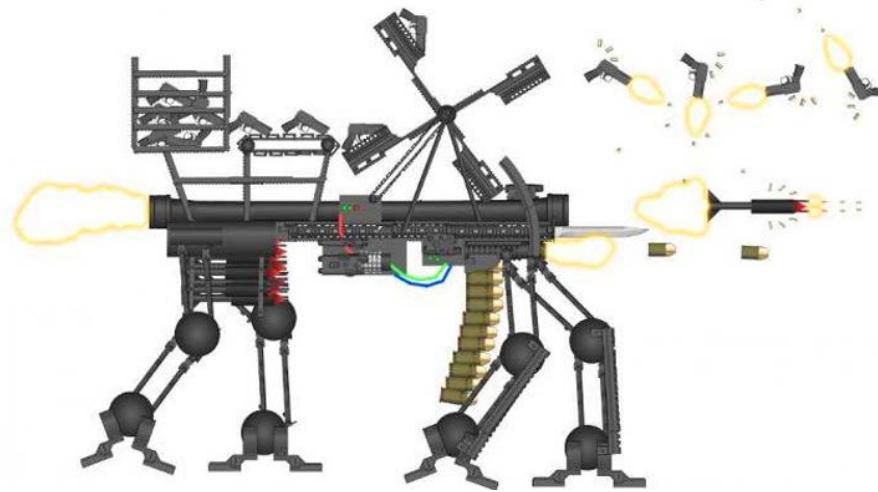


Python

Why Python?



쉽고 간단하며



다양한 기능 제공

Why Python?

인간 지향적인

간단한 문법

Why Python?

화면에 Hello, World 찍기

자바

```
class HelloWorldApp {  
    public static void main(String[] args) {  
        System.out.println("Hello World!");  
    }  
}
```

파이썬

```
Print ("Hello World")
```

Why Python?

1부터 10까지 출력

자바

```
for (i = 1; i < 11; i++){\n    System.out.println (i)\n}
```

파이썬

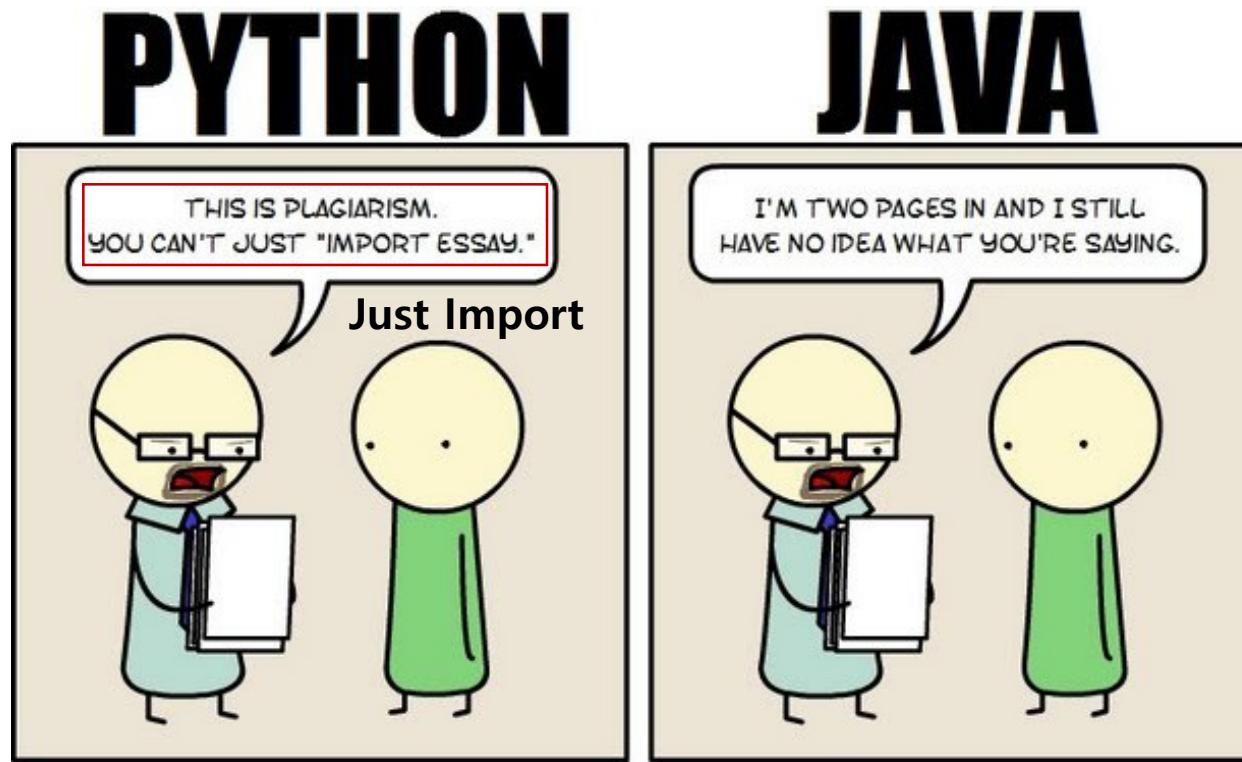
```
for i in range(1, 11):\n    print (i)
```

Why Python?

다양한 라이브러리

넓은 활용범위

Why Python?



파이썬은 대부분의 라이브러리가 이미
다른 사용자에 의해서 구현되어 있음 (특히 통계, 데이터 분석)

어디 어디 쓰였나?



Source: <https://goo.gl/le2OOq>

어디 어디 쓰였나?

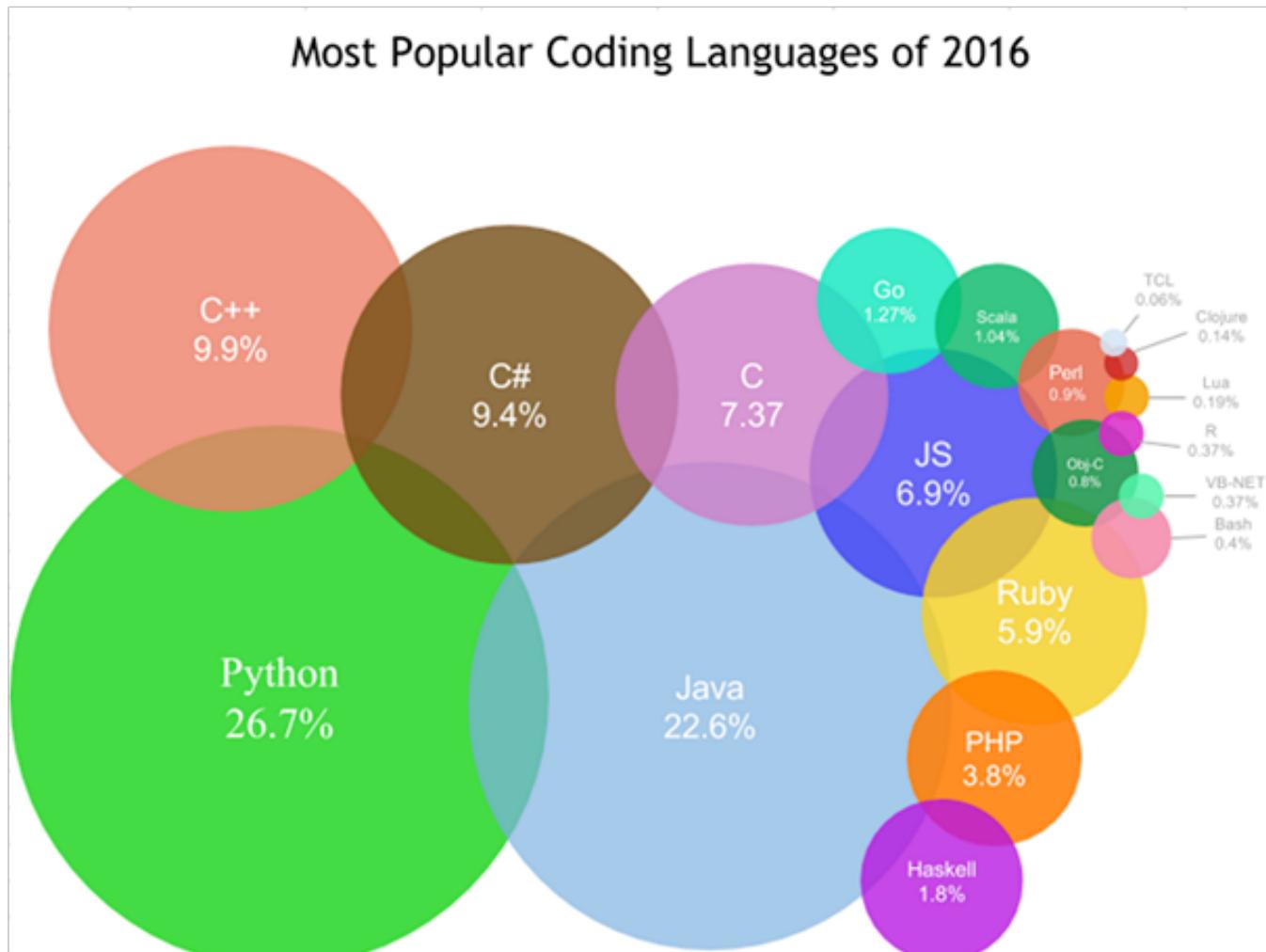


Source: <http://goo.gl/KVJM5I>

Why Python?

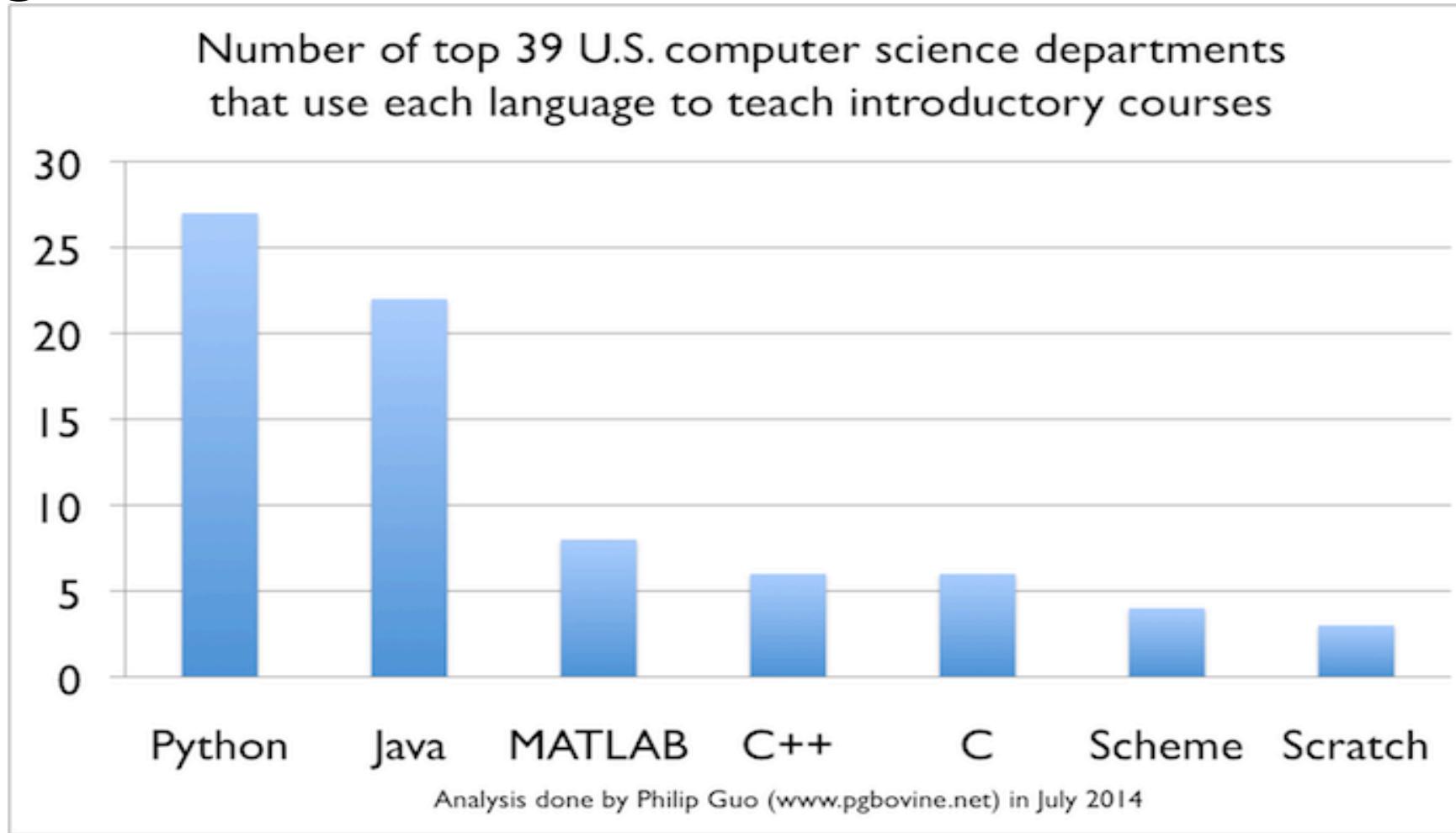
가장 대중화된 언어

Why Python?



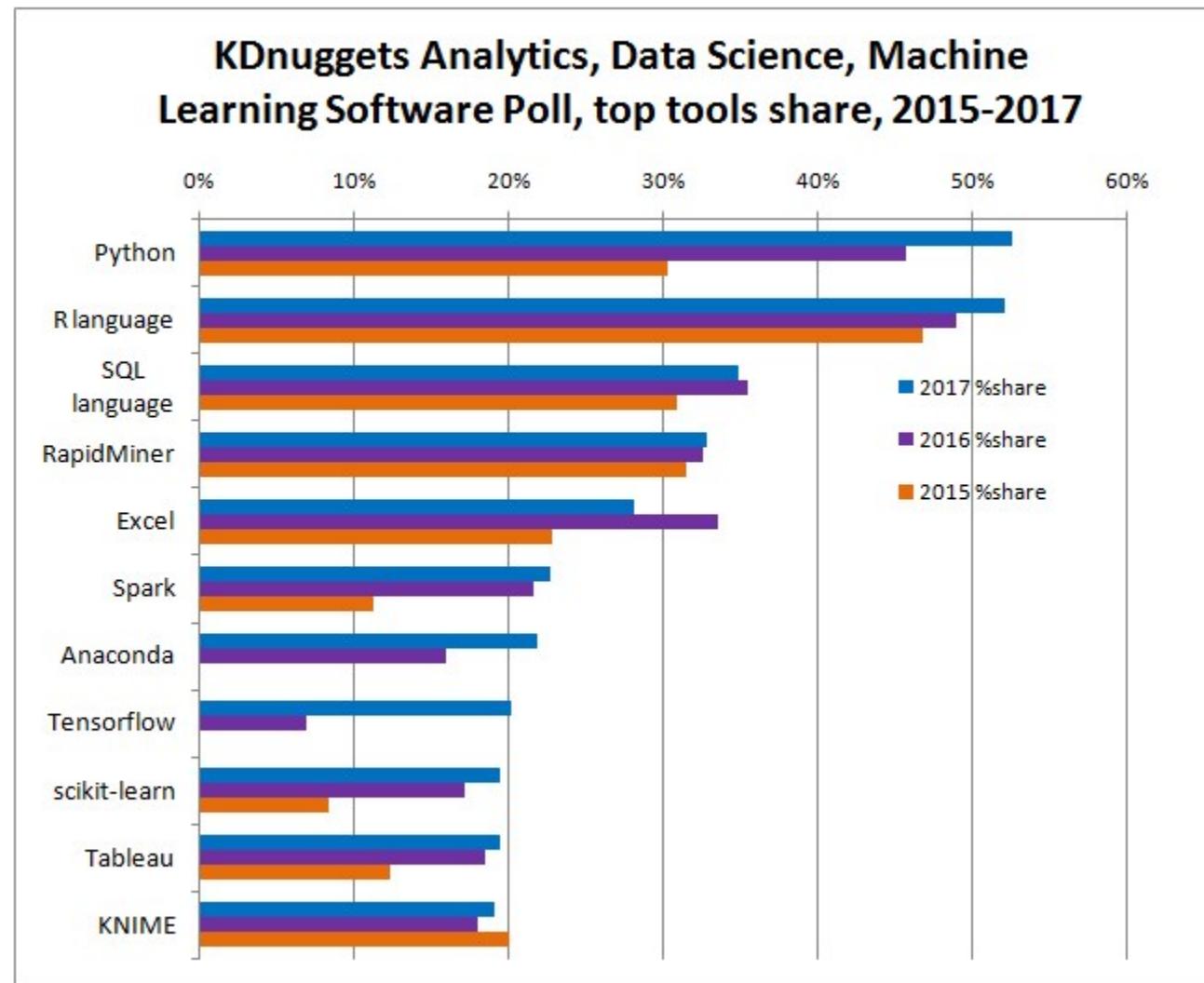
Source: <http://goo.gl/9CBJxE>

Why Python?



Source: <http://goo.gl/oD7L9v>

Why Python?

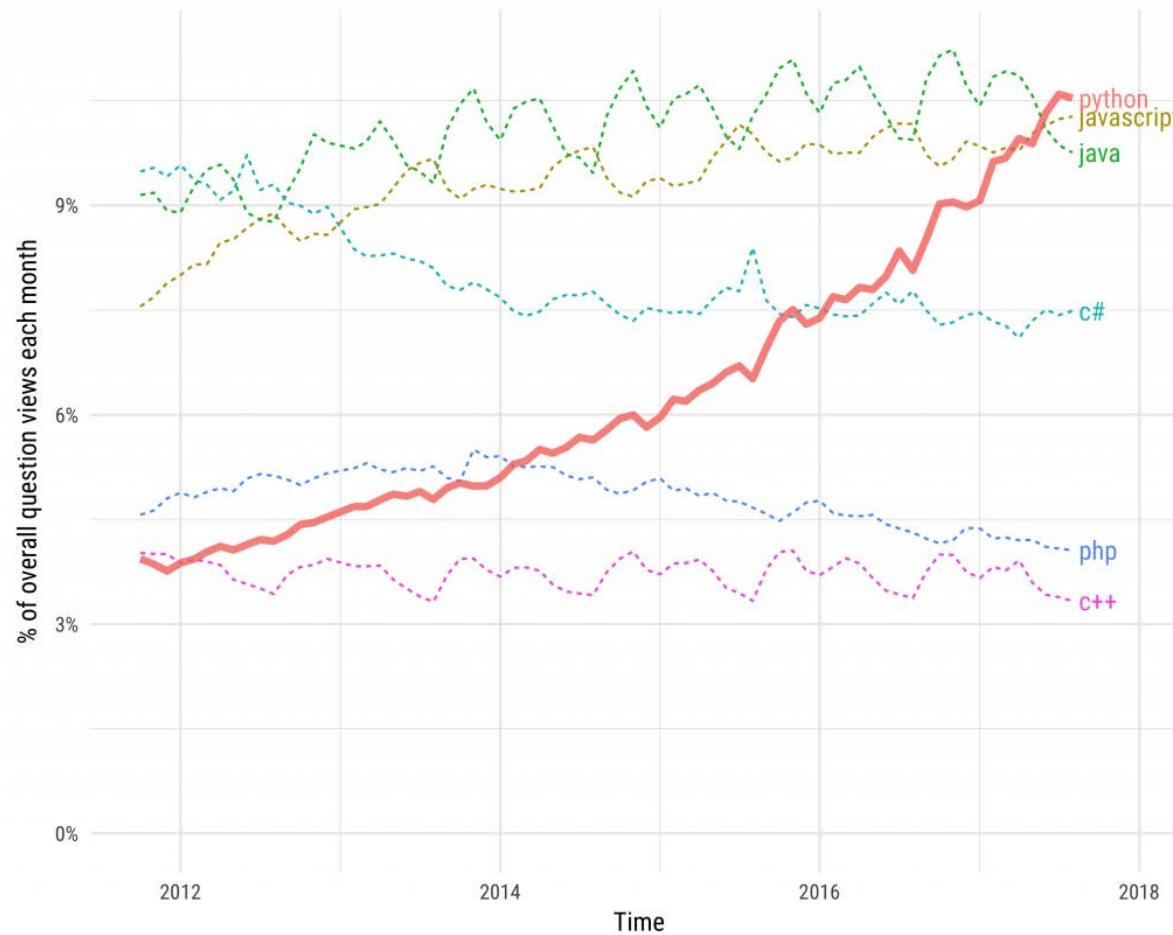


<https://goo.gl/cyDssa>

Why Python?

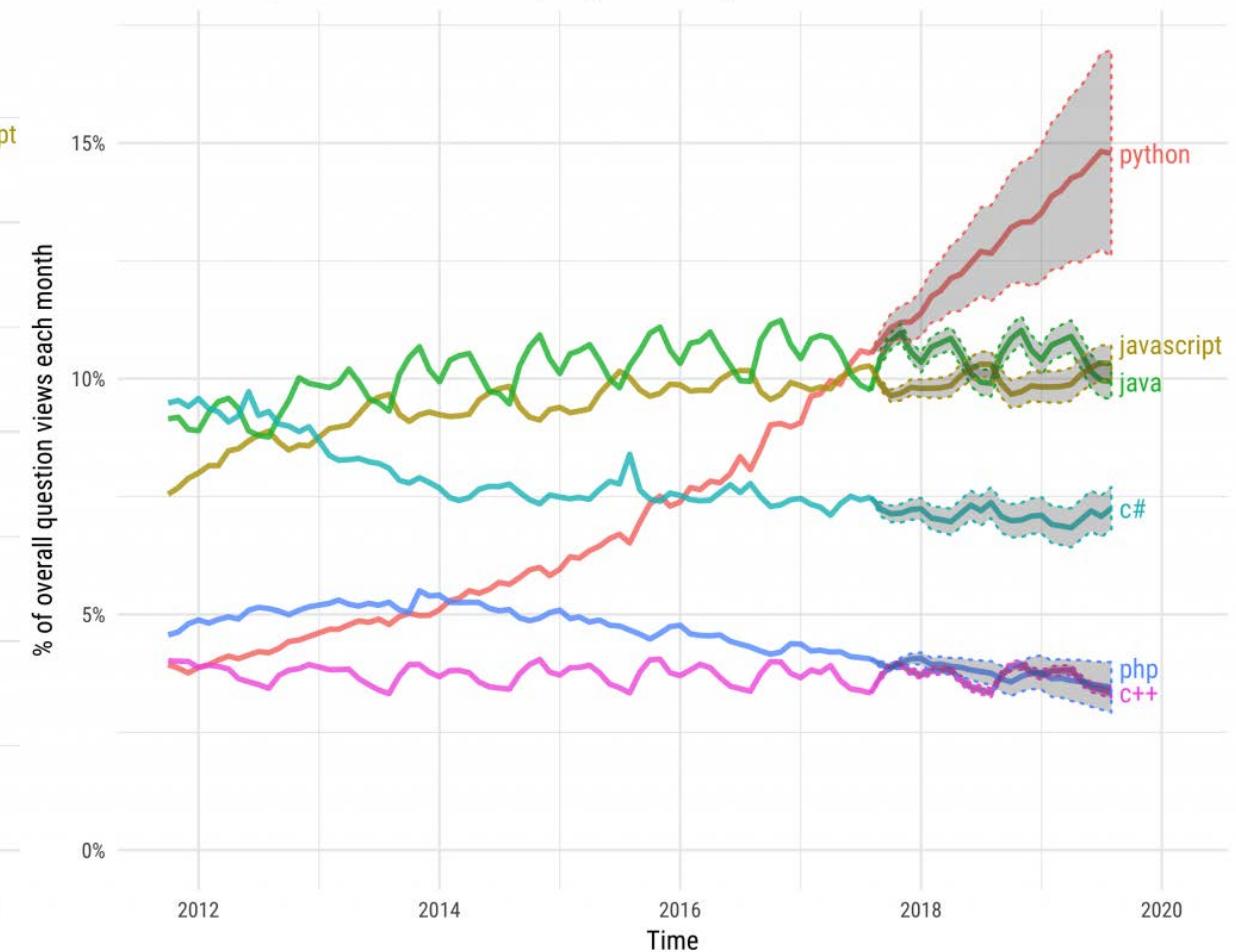
Growth of major programming languages

Based on Stack Overflow question views in World Bank high-income countries



Projections of future traffic for major programming languages

Future traffic is predicted with an STL model, along with an 80% prediction interval.



<https://stackoverflow.blog/2017/09/06/incredible-growth-python/>

3개 국어를 하자

영어, 중국어, 그리고



Life is short.

You need Python.



Human knowledge belongs to the world.

The background of the slide features a white humanoid robot. The robot's head is white with two large, circular, multi-layered eyes. Its right arm is extended forward, with its hand open. Inside the hand, the binary code "0110000110" is displayed in blue 3D digits.

Coding Environment

Programming and Python

최성철 교수

Director of TEAMLAB

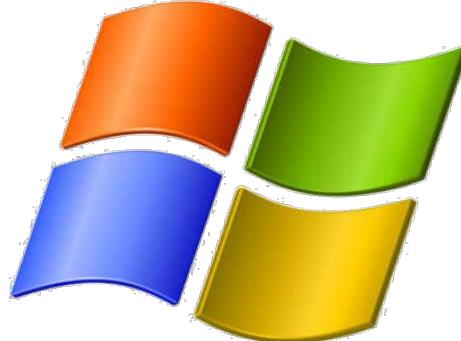
개발 환경 설정

개발환경이란?

- 프로그램을 작성하고, 실행 시키는 환경
- 일반적으로 <코딩 환경>이라고 부름
- 개발환경을 결정
 - 1) 운영 체제 (Operating System, OS)
 - 2) Python Interpreter
 - 3) 코드 편집기 (Editor)

운영체제 - OS

선호하는 운영체제를 선정



Windows



Linux



Mac OS

장

친숙함,
초기엔 쉬움

단

모듈 설치 어려움
참고 문서 부족

모듈 설치 쉬움
공짜, 참고문서 많음

OS 자체 사용이 어려움

모듈 설치 쉬움
참고문서도 많음

비쌈

Python Interpreter

- 2.7과 3.X버전이 현재 주로 사용되고 있음
- 기존 라이브러리 사용 여부에 따라 버전을 선택
- 최근 3.X를 사용하기 시작하는 추세

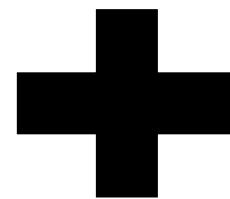
종류	설명	Download
Python	- 일반적인 파이썬, 기본적인 모듈을 포함	https://www.python.org/
Ananconda		http://www.continuum.io/
Canopy	- 다양한 과학 계산용 모듈들을 묶어서 패키지	https://www.enthought.com/products/canopy/

코드 편집기

- 파이썬 코드도 일종의 문서
→ 한글, 워드 처럼 코드를 입력할 문서 편집기가 필요함
- text 타입의 문서를 저장하는 모든 편집기 사용가능

종류	설명	Download
메모장	윈도우의 기본 문서 편집도구	시작 => 메모장 (Windonws)
VI editor	리눅스의 기본 문서 편집도구	http://www.vim.org/
Sublime Text Atom	프로그래밍에 특화된 문서 편집도구	http://www.sublimetext.com/ https://atom.io/
PyCharm	다양한 기능을 갖춘 파이썬 전용 개발 도구	https://www.jetbrains.com/pycharm/

우리의 선택



간편한 파이썬
패키지 관리도구

새롭게 떠오르는
오픈소스 편집기



Human knowledge belongs to the world.

A white humanoid robot arm is shown from the shoulder down, holding a blue digital display. The display shows the binary code "0110000110". The robot's hand is visible at the bottom right.

Memory & Variable

Python Basic

최성철 교수

Director of TEAMLAB

[복습]

기대되는 결과 값?

```
>>> Professor = "Sungchul Choi"  
>>> print (Professor)
```

Python Shell

```
>>> a = 7  
>>> b = 5  
>>> print (a+b)
```

Python Shell

```
>>> a = 7  
>>> b = 5  
>>> print ("a+b")
```

Python Shell

정답

```
>>> Professor = "Sungchul Choi"  
>>> print(Professor)  
Sungchul Choi
```

Python Shell

```
>>> a = 7  
>>> b = 5  
>>> print(a+b)  
12
```

Python Shell

```
>>> a = 7  
>>> b = 5  
>>> print("a+b")  
a+b
```

Python Shell

[참고] 표기법

Python Shell

cmd 또는 터미널 창에서 실행시킨 python shell

Editor

Atom, Sublime Text 등 코드 에디터

정답

```
>>> Professor = "Sungchul Choi"  
>>> print(Professor)  
Sungchul Choi
```

Python Shell

```
>>> a = 7  
>>> b = 5  
>>> print(a+b)  
12
```

Python Shell

```
>>> a = 7  
>>> b = 5  
>>> print("a+b")  
a+b
```

Python Shell

이 순간 컴퓨터에서
무슨 일이 일어난 일은?

[문제]

Professor = “Sungchul Choi” 의 의미는”

- ① Professor의 이름은 Sungchul Choi 이다.
- ② Professor는 Sungchul Choi 이다.
- ③ Professor와 Sungchul Choi는 같다.
- ④ Professor에 Sungchul Choi를 넣어라

[정답]

Professor = “Sungchul Choi” 의 의미는”

④ Professor에 Sungchul Choi를 넣어라

정확히는 Professor라는 변수(?)에

“Sungchul Choi” 라는 값을 넣으라는 의미

[문제]

print (a+b) 와 print ("a+b") 의 차이는?

```
>>> a = 7  
>>> b = 5  
>>> print (a+b)  
12
```

Python Shell

```
>>> a = 7  
>>> b = 5  
>>> print ("a+b")  
a+b
```

Python Shell

[정답]

print (a+b)

- a라는 변수에 있는 값과 b라는 변수에 있는 값을 더해서 화면에 출력하라는 의미

print ("a+b")

- "a+b" 값을 화면에 출력하는 의미

다시 질문

Professor = 'Sungchul Choi'

a=3 , b=7

이 순간 컴퓨터에서
무슨 일이 일어난 걸까요?

[정답]

Professor 라는 이름을 가진 변수에
“Sungchul Choi” 라는 값 을 할당

a 라는 이름을 가진 변수에
3 이라는 값을 할당

그럼 변수는 어디에 저장될까?

변수(Variable)란?

수학식 $2x + 7y = 14$ 에서 변수는
x와 y를 의미함

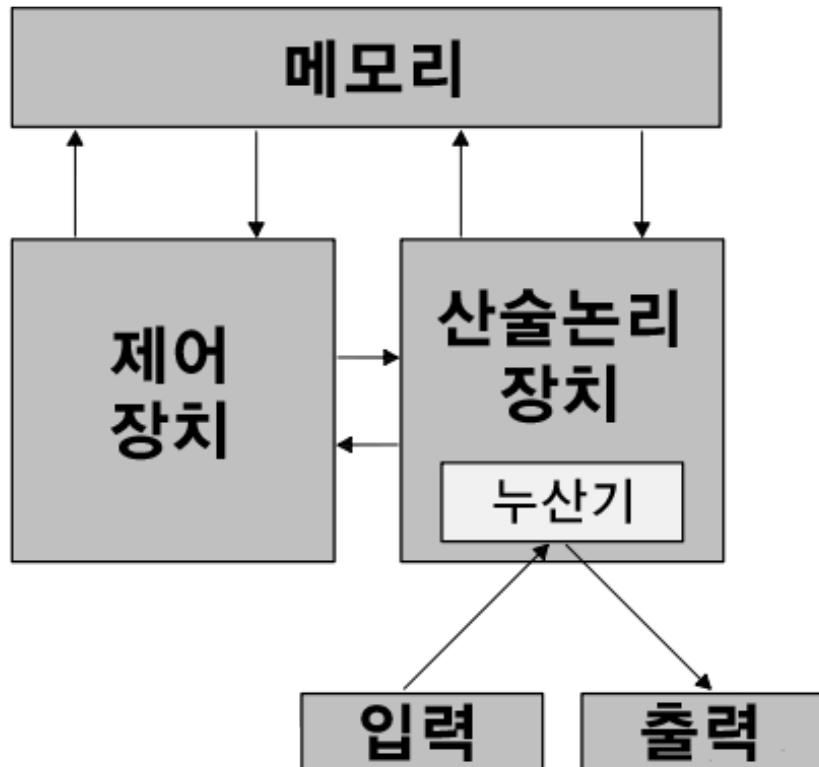
프로그래밍에서 변수는
수학과 약간 다른 개념

변수(Variable)란?

프로그래밍에서는 변수는
값을 저장하는 장소

변수는 메모리 주소를 가지고 있고
변수에 들어가는 값은
메모리 주소에 할당됨

컴퓨터의 구조 – 폰 노이만 아키텍처



폰 노이만 아키텍처에서는 사용자가 컴퓨터에 값을 입력하거나 프로그램을 실행할 경우 그 정보를 먼저 **메모리**에 저장시키고 CPU가 순차적으로 그 정보를 **해석하고 계산하여** 사용자에게 결과값 전달

[알아두면 상식] 폰 노이만은?



양자 역학, 함수 해석학, 집합론, 위상수학, 컴퓨터 과학, 수치해석, 경제학, 통계학 등 다양한 학문 분야에 걸쳐 놀라운 업적을 남긴 **역사상 가장 위대한 천재 중 한 명**

- ✓ 경제학 주요 학문 분야 [게임이론]의 창시 (영화 뷰티풀 마인드 참고)
- ✓ 현대 컴퓨터의 기본 구조인 [폰 노이만 아키텍처] 제시
- ✓ [자기 복제]의 관한 연구로 DNA 발견을 예측
- ✓ 6살 때 8자리 (천만) 단위의 암산을 자유자재로 할 수 있었음
- ✓ 초기 컴퓨터보다 빠른 계산 속도로 수학 문제를 풀었음
(오른쪽에서 4번째 자리수가 7인 가장작은 2의 지수는?)

변수와 메모리 주소

Professor = 'Sungchul Choi', a=3 , b=7 입력 시



메모리 위에선

Memory	Address	Variable
	0x0007	
	0x0006	
	0x0005	
	0x0004	
7	0x0003	b
3	0x0002	a
Sungchul Choi	0x0001	Professor

메모리와 변수

변수 (Variable)

- 프로그램에서 사용하기 위한 특정한 값을 저장하는 공간
- 선언 되는 순간 메모리 특정영역에 공간이 할당됨
- 변수에는 값이 할당되고 해당 값은 메모리에 저장됨
- $A = 8$ 의 의미는 “A는 8이다”가 아닌
A라는 이름을 가진 메모리 주소에 8을 저장하라 임

변수 이름 작명법

- 알파벳, 숫자, 언더스코어(_)로 선언 가능

ex) data = 0, _a12 = 2, _gg = 'afdf'

- 변수명은 의미 있는 단어로 표기하는 것이 좋다

ex) professor_name = 'Sungchul Choi'

- 변수명은 대소문자가 구분된다.

ex) ABC와 Abc는 같지 않다

- 특별한 의미가 있는 예약어는 쓰지 않는다.

ex) for, if, else 등



Human knowledge belongs to the world.



Basic Operation

Python Basic

최성철 교수

Director of TEAMLAB

Basic Operation (간단한 연산)

- 복잡한 프로그램을 작성하기 앞서 간단한 **사칙연산**과 **문자열 처리** 등의 기본을 배워야 함
- 이를 위해 본 장에서는 아래 내용을 습득
 - 1) 기본 자료형 (Data Types)
 - 2) 연산자와 피연산자
 - 3) 데이터 형변환
- 이를 통해 간단한 프로그램 작성의 기초를 익힘

정답

```
>>> Professor = "Sungchul Choi"  
>>> print (Professor)  
Sungchul Choi
```

Python Shell

```
>>> a = 7  
>>> b = 5  
>>> print (a+b)  
12
```

Python Shell

```
>>> a = 7  
>>> b = 5  
>>> print ("a+b")  
a+b
```

Python Shell

기본 자료형 (Fundamental Data Types)

Data Type : 파이썬이 처리하는 기본 데이터 유형

유형		설명		예시	선언 형태
수치자료형	정수형	Integer	양/음의 정수	1,2,3,100, -9	data = 1
	실수형	Float	소수점이 포함된 실수	10.2, -9.3, 9.0	data = 9.0
문자형(문자형)		String	따옴표 (' / ")에 들어가 있는 문자형	abc, a20abc	data = 'abc'
논리/불린 자료형		Boolean	참 또는 거짓	True, False	data = True

자료형 사용 예시

```
>>> a = 1 # Integer
>>> b = 1 # Integer
>>> print (a, b)
1 1
>>> a = 1.5 # Float
>>> b = 3.5 # Float
>>> print (a, b)
1.5 3.5
>>> a = "ABC" # String
>>> b = "101010" # String
>>> print (a, b)
ABC 101010
>>> a = True # Boolean 대소문자 구분
>>> b = False # Boolean 대소문자 구분
>>> print (a, b)
True False
```

Python Shell

연산자(Operator)와 피연산자(operand)

- $+, -, *, /$ 같은 기호들을 연산자라고 칭함
- 연산자에 의해 계산이 되는 숫자들은 피연산자라 칭함
- “ $3 + 2$ ”에서 3과 2는 피연산자, $+$ 는 연산자임
- 수식에서 연산자의 역할은 수학에서 연산자와 동일
- 연산의 순서는 수학에서 연산 순서와 같음
- 문자간에도 $+$ 연산이 가능함

제곱승과 나머지 구하기

“**”는 제곱승 계산 연산자

```
>>> print(3 * 3 * 3 * 3 * 3) # 3을 다섯 번 곱함  
243  
>>> print(3 ** 5)           # 3의 5승  
243
```

Python Shell

“%”는 나머지를 구하는 연산자

```
>>> print(7 / 2)      # 7 나누기 2 (정수형 계산)  
3.5  
>>> print(7 % 2)      # 7 나누기 2의 나머지는  
1
```

Python Shell

증가 또는 감소 연산

$a += 1$ 는 $a = a + 1$ 과 같은 의미로 증가연산 ($+=$)

```
>>> a = 1          # 변수 a에 1을 할당  
>>> a = a + 1    # a에 1를 더한 후 그 값을 a에 할당  
>>> print(a)     # a 출력  
2  
>>> a += 1        # a 증가 연산  
>>> print(a)     # a 출력  
3  
>>> a = a - 1    # a에 1을 뺀 후 그 값을 a에 할당  
>>> a -= 1        # a 감소 연산  
>>> print(a)     # a 출력  
1
```

Python Shell

증가 또는 감소 연산

a = a + 1 의 의미는?

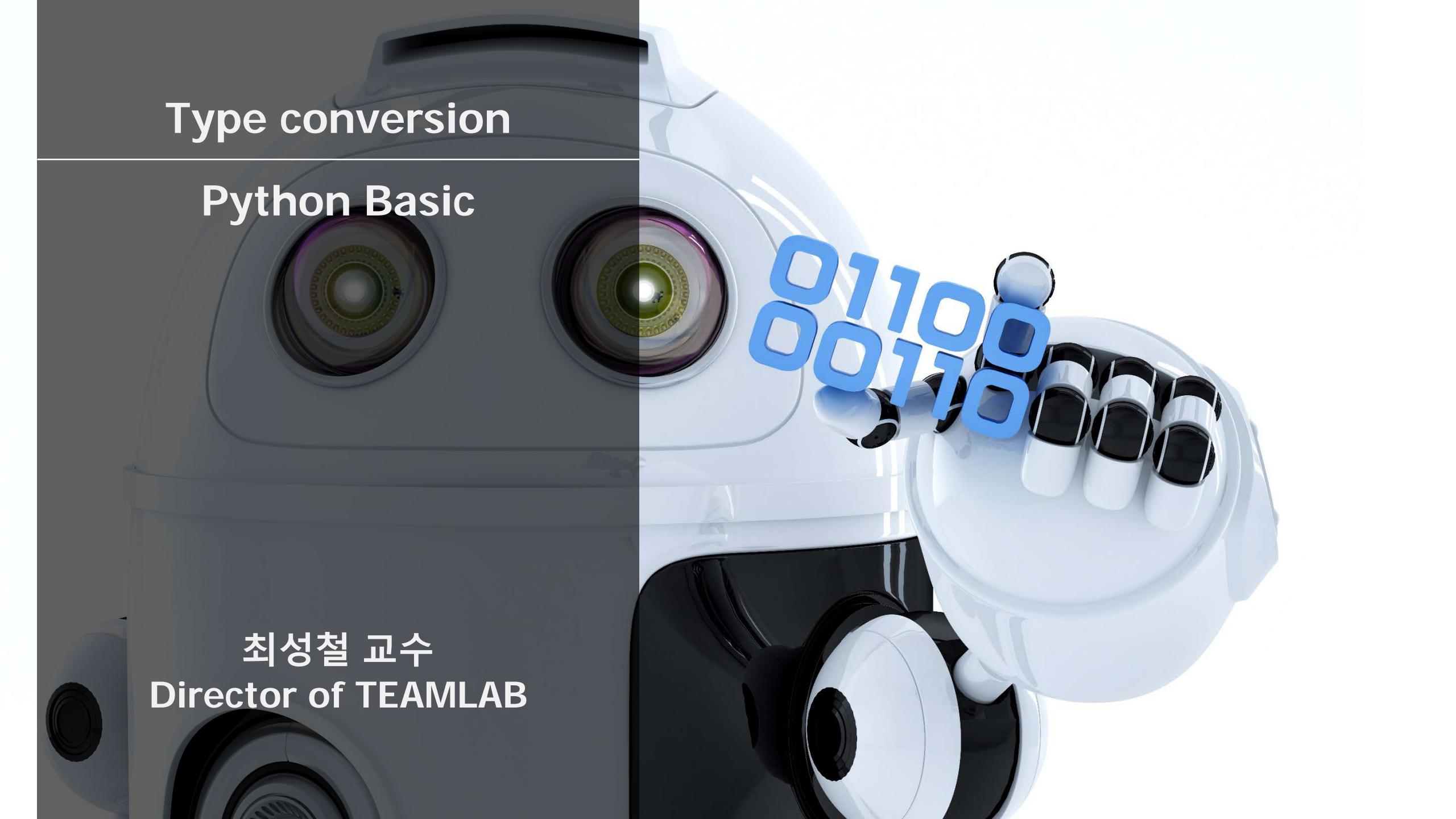
만약 $a = 4$ 일 때 $a = 4 + 1$ 로 a 에 다시 5가 할당(assign)됨

즉 좌변에 a 는 할당 받는 변수 (**variable**)

우변에 a 는 기존 a 의 값(**value**)



Human knowledge belongs to the world.

The background of the slide features a white humanoid robot with large, expressive eyes and a visible mechanical arm. The robot's hand is raised, holding a set of blue 3D-rendered binary digits (01100110).

Type conversion

Python Basic

최성철 교수

Director of TEAMLAB

데이터 형 변환: 정수형 ↔ 실수형

float()와 int() 함수를 사용하여 데이터의 형 변환 가능

```
>>> a = 10          # a 변수에 정수 데이터 10을 할당  
>>> print (a)      # a가 정수형으로 출력  
10
```

Python Shell

```
>>> a = float(10)    # a를 실수형으로 변환 / 정수형은 int()  
>>> print (a)        # a를 출력  
10.0                 # a가 실수형으로 출력됨
```

```
>>> b = 3            # b 에 정수 데이터 3 할당  
>>> print a / b       # 실수형으로 a 나누기 b를 출력  
3.333333333333333  # 실수형 결과값 출력
```

10.3과 10.7 정수형으로 형 변환 후 덧셈하면 결과값은?

데이터 형 변환: 정수형 ↔ 실수형

10.3과 10.7 정수형으로 형 변환 후 덧셈하면 결과값은?

```
>>> a = 10.7  
>>> b = 10.3
```

Python Shell

```
>>> a= int(a)          # a를 정수형으로 형변환후 a에 할당  
>>> b = int (b)        # b를 정수형으로 형변환후 b에 할당  
>>> print (a+b)       # 정수형 a와 b의 합을 출력
```

20

```
>>> print (a)          # 정수형 a값 출력  
10  
>>> print (b)          # 정수형 b값 출력  
10
```

실수형에서 정수형으로 형 변환 시 소수점 이하 내림

데이터 형 변환: 숫자 ↔ 문자열

문자열로 선언된 값도 `int()`, `float()` 함수로 형 변환 가능

```
>>> a = '76.3'  
>>> b = float(a)
```

a에 문자열 76.3을 할당, "은 문자열을 의미
a를 실수형으로 형 변환 후 b에 할당

Python Shell

```
>>> print (a)  
76.3
```

a값 출력

```
>>> print (b)  
76.3
```

b 값 출력

```
>>> print (a + b)  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
TypeError: cannot concatenate 'str' and 'float' objects
```

a와 b를 더함 그러나 문자열과 숫자열의
덧셈이 불가능하여 에러발생

a와 b를 실수형으로 덧셈하고, 문자열로 연결하려면?

데이터 형 변환: 숫자 ↔ 문자열

a와 b를 실수형으로 덧셈하고, 문자열로 연결하려면?

Python Shell

```
>>> a = float(a)          # a를 실수형으로 형 변환 후 a에 할당  
>>> b = a                # 실수형 a 값을 b에 할당  
>>> print (a + b)        # 두 실수형 더한 후 출력  
152.6  
  
>>> a = str(a)           # 실수형 a 값을 문자열로 변환 후 a 할당  
>>> b = str(b)           # 실수형 b 값을 문자열로 변환 후 b 할당  
>>> print (a + b)        # 두 값을 더한 후 출력  
# 문자열간 덧셈은 문자열간 단순 연결  
76.376.3
```

str() 함수는 숫자 값을 문자 값으로 변환함 데이터간의 형 변환을 **casting**이라고 함

데이터 형 확인하기

`type()` 함수는 변수의 데이터 형을 확인하는 함수

```
>>> a=int(10.3)          # a는 정수형으로 10.3을 할당  
>>> b=float(10.3)        # b는 실수형으로 10.3을 할당  
>>> c=str(10.3)          # c는 문자열으로 10.3을 할당  
  
>>> type(a)              # a의 타입을 출력  
<class 'int'>  
>>> type(b)              # b의 타입을 출력  
<class 'float'>  
>>> type(c)              # c의 타입을 출력  
<class 'str'>
```

Python Shell

컴퓨터의 반올림 오차

아래와 같이 나오는 이유는 무엇일까?

Python Shell 2.7 Only

```
>>> c = 38.8          # c에 실수형 38.8 할당  
>>> print(c)        # c 출력  
38.8  
>>> c              # c에 있는 값은?  
38.79999999999997  # 응?
```

컴퓨터의 모든 값은 이진수로 변환되어 메모리에 저장
Python 2.7에서만 나오는 숫자 3x에선 정상으로 나옴

컴퓨터의 반올림 오차

0.1를 이진수 변환하여라

$$0.1 \times 2 = 0.2 \rightarrow 0$$

$$0.2 \times 2 = 0.4 \rightarrow 0$$

$$0.4 \times 2 = 0.8 \rightarrow 0$$

$$0.8 \times 2 = 1.6 \rightarrow 1$$

$$0.6 \times 2 = 1.2 \rightarrow 1$$

$$0.2 \times 2 = 0.4 \rightarrow 0 \dots\dots$$

0.00011001100110011.....₍₂₎

단순한 실수도 이진수로 변환하면 무한소수가 됨

반올림오차는 충분히 작아 반올림을 하여 일반적으로 문제가 되지 않음

[알아두면 상식] 컴퓨터는 왜 이진수를 쓰나?

컴퓨터는 실리콘이라는 재료로 만든 반도체로 구성됨

반도체는 특정 자극을 줬을 때 전기를 통할 수 있게 하는 물질



Source : <http://samsungsemiconstory.com/1>

도체와 부도체에 반해 반도체는 전류의 흐름의 제어가 가능

전류가 흐를 때 1, 흐르지 않을 때 0으로만 숫자를 표현할 수 있음

이진수 한자리를 bit라 칭하고 8개의 bit는 1byte



Human knowledge belongs to the world.



Console In & Out

Python Basic II

최성철 교수
Director of TEAMLAB

01100
00110

**어떻게 프로그램과
데이터를
주고 받을 것인가?**

프로그램과 소통하는 방법



Graphical User Interface



Command Line Interface

Command Line Interface

Graphic User Interface (GUI)와 달리

Text를 사용하여 컴퓨터에 명령을 입력하는

인터페이스 체계

Windows – CMD window

Mac, Linux – Terminal

Console = Terminal = CMD창

어원: 디스플레이와 키보드가 조합된 장치

현재: CLI로 입력하는 화면



IBM 3270 terminal



IBM 7094, a typical Mainframe

Source: <http://www.gliderwiki.org/wiki/116>

콘솔창 입출력 I

`input()` 함수는 콘솔창에서 문자열을 입력 받는 함수

`console_test.py`

```
print ("Enter your name:")
somebody = input() # 콘솔창에서 입력한 값을 somebody에 저장
print ("Hi", somebody, "How are you today?")
```

Editor

실행

```
python console_test.py
```

Terminal

```
Enter your name:
```

```
cs50
```

```
Hi cs50 How are you today?
```

```
LoginID@cs50:~$
```

콘솔창 입출력 II

콤마() 사용할 경우 print 문이 연결됨

```
>>> print ("Hello World!", "Hello Again!!!")
```

```
Hello World! Hello Again!!!
```

, 사용

Python Shell

실행 시 두 문장이 연결 돼서 출력됨

숫자 입력 받기

```
temperature = float(input("온도를 입력하세요 :")) # 입력 시 바로 형 변환 하기
```

```
print(temperature)
```

Editor

temperature.py

```
python temperature.py  
온도를 입력하세요 : 103  
103.0
```

Terminal

실행



Human knowledge belongs to the world.

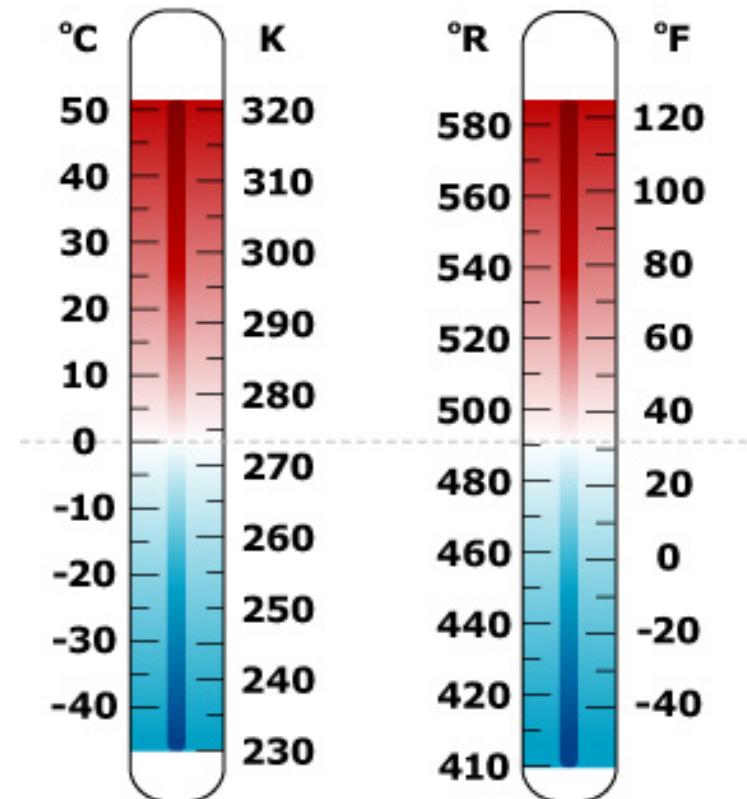


Lab: Fahrenheit Converter

Python Basic II

가천대학교 | 산업경영공학과
최성철 교수

섭씨와 화씨



$$^{\circ}\text{F} = (^{\circ}\text{C} \times 1.8) + 32$$

<http://goo.gl/dGQPES>

Lab: 화씨 변환기

아래와 같이 출력되는 프로그램을 만드시오

```
python fahrenheit.py
```

Terminal

본 프로그램은 섭씨를 화씨로 변환해주는 프로그램입니다

변환하고 싶은 섭씨 온도를 입력해 주세요:

32.2

섭씨온도 : 32.2

화씨온도 : 89.96

fahrenheit.py

섭씨 온도 변환 공식은: $((9/5) * \text{섭씨온도}) + 32$



Human knowledge belongs to the world.



print formatting

Python Basic II

가천대학교 | 산업경영공학과
최성철 교수

**형식(format)에 맞춰서
출력을 할 때가 있음**

A	B	C	D	
1	Date	Daily Income	Daily Expenses	Percent Gain/Loss
2	1-May	\$322.00	\$146.00	221%
3	2-May	\$371.00	\$135.00	275%
4	3-May	\$345.00	\$467.00	74%
5	4-May	\$345.00	\$216.00	160%
6	5-May	\$150.00	\$269.00	56%
7	6-May	\$116.00	\$481.00	24%
8	7-May	\$440.00	\$203.00	217%

**print 문을 활용해서
결과 formatting 하기**

print formatting

프린트 문은 기본적인 출력 외에 출력의 양식을 형식을 지정 가능

Editor

```
print(1,2,3)
print("a" + " " + "b" + " " + "c")
print("%d %d %d" % (1,2,3))
print("{} {} {}".format("a","b","c"))
```

Two types

일반적으로 %-format 과 str.format() 함수를 사용함

```
print( '%s %s' % ('one', 'two') )
```

```
print( '{} {}'.format('one', 'two') )
```

```
print( '%d %d' % (1, 2) )
```

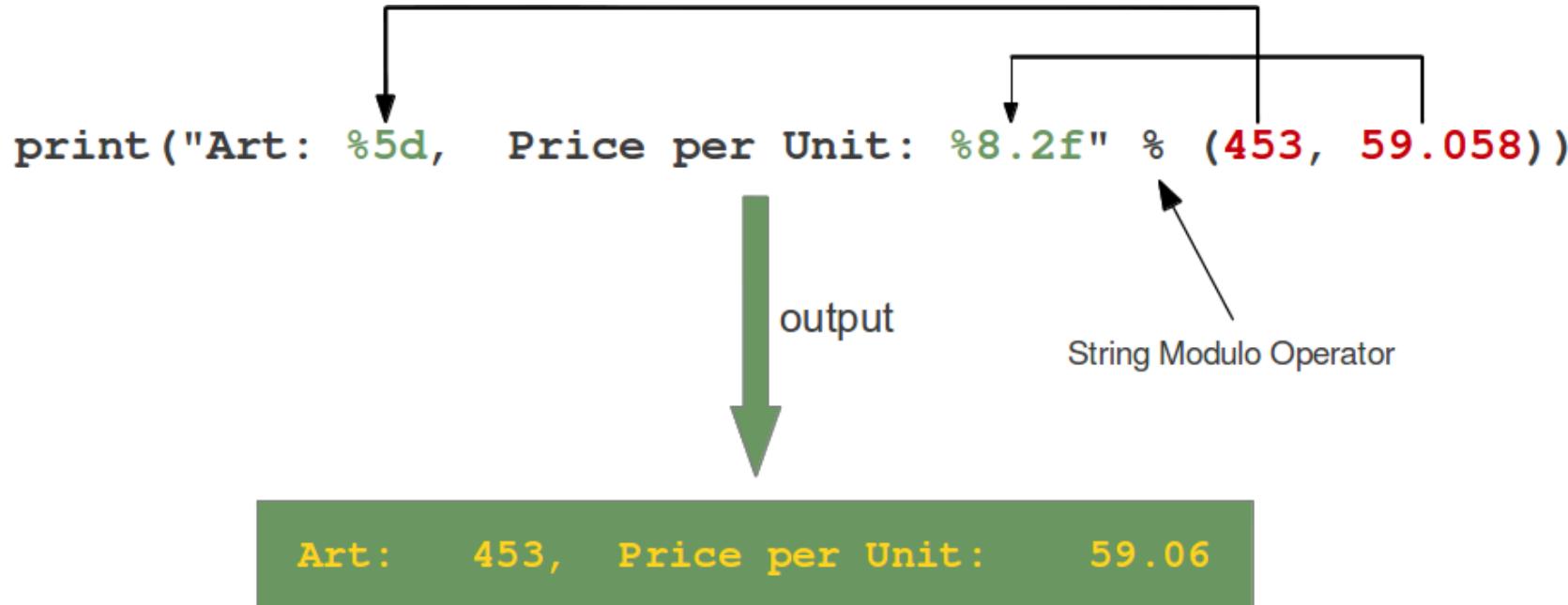
```
print( '{} {}'.format(1, 2) )
```

%-format

“%datatype” % (variable) 형태로 출력 양식을 표현

```
print("I eat %d apples." % 3)
print("I eat %s apples." % "five")
number = 3; day="three"
print("I ate %d apples. I was sick for %s days."
      % (number, day))
print("Product: %s, Price per unit: %f." % ("Apple", 5.243))
```

%-format



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

https://www.python-course.eu/python3_formatted_output.php

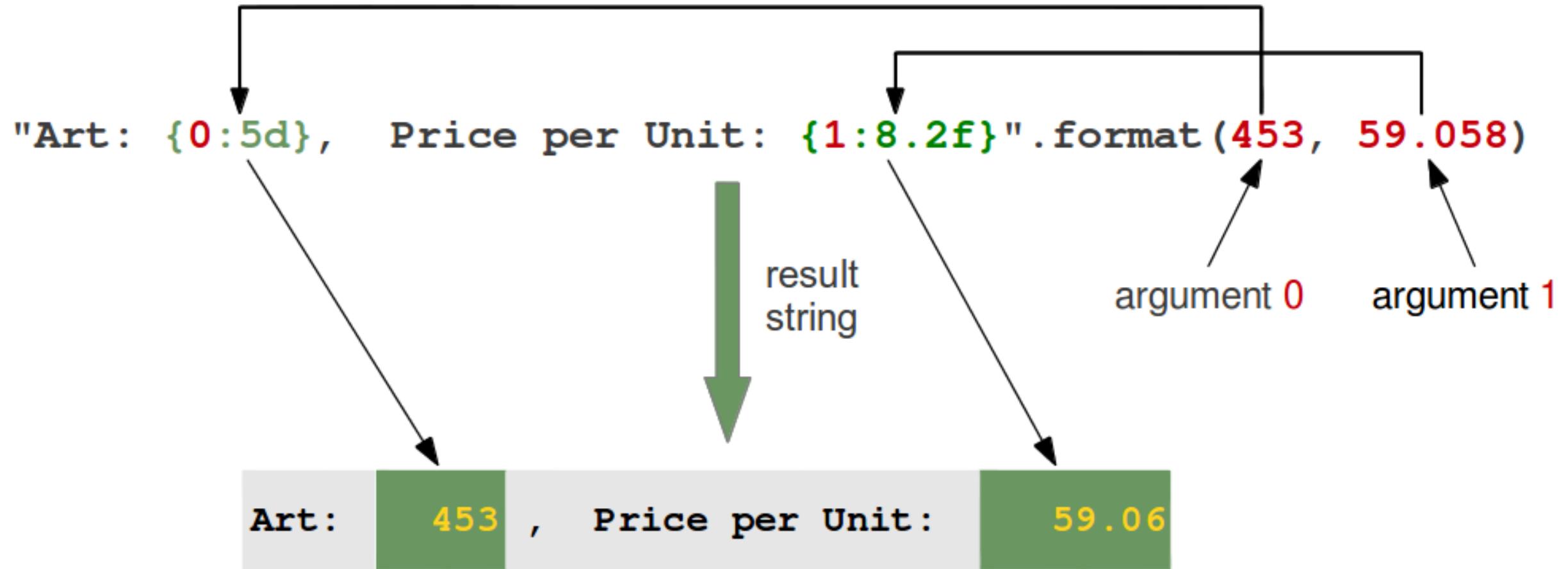
<https://wikidocs.net/13>

str.format()

“~~~~{datatype}~~~~”.format(argument)

```
age = 36; name='Sungchul Choi'  
print("I'm {0} years old.".format(age))  
print("My name is {0} and {1} years old.".format(name,age))  
print(  
    "Product: {0}, Price per unit: {1:.3f}.".format(  
        "Apple", 5.243))
```

str.format()



padding

여유 공간을 지정하여 글자배열 + 소수점 자릿수를 맞추기

```
print("Product: %5s, Price per unit: %.5f." % ("Apple",  
5.243))  
print("Product: {0:5s}, Price per unit:  
{1:.5f}.".format("Apple", 5.243))  
print("Product: %10s, Price per unit: %10.3f." % ("Apple",  
5.243))  
print("Product: {0:>10s}, Price per unit:  
{1:10.3f}.".format("Apple", 5.243))
```

naming

해당 표시할 내용을 변수로 표시하여 입력

```
print("Product: %(name)10s, Price per unit: %(price)10.5f." %  
      { "name" : "Apple", "price" : 5.243 })
```

```
print("Product: {name:>10s}, Price per unit:  
{price:10.5f}.".format(name="Apple", price=5.243))
```

See

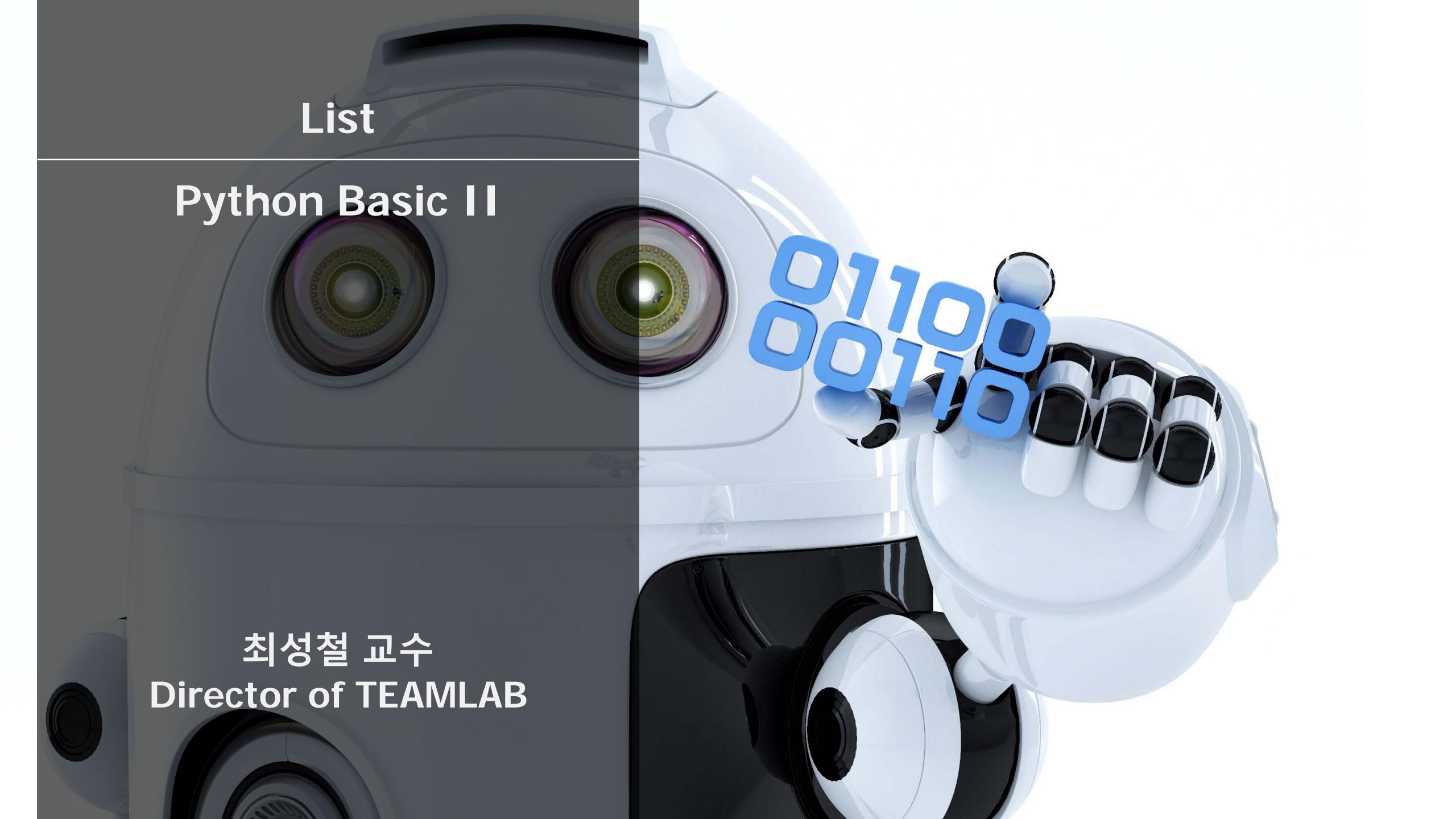
<https://docs.python.org/3/tutorial/inputoutput.html>

https://www.python-course.eu/python3_formatted_output.php

<https://wikidocs.net/13>



Human knowledge belongs to the world.



List

Python Basic II

최성철 교수

Director of TEAMLAB

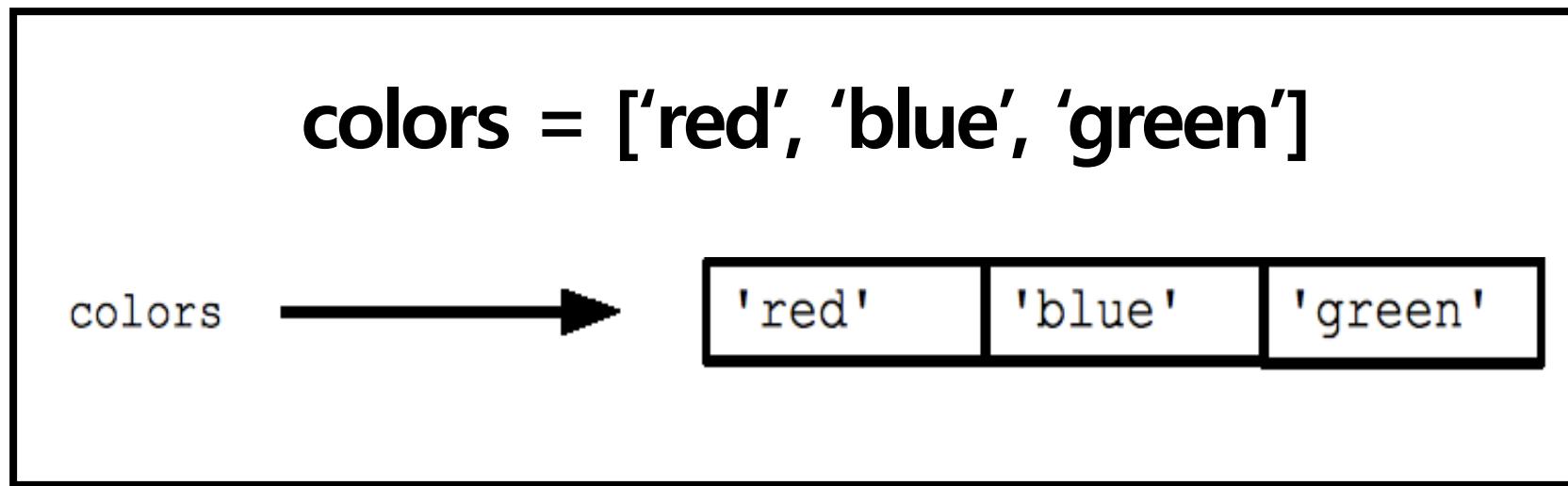
**데이터가 100개 있다면
어떻게 관리할 것인가?**

**100명의 성적 관리를
위한 변수는 몇 개?**

100개? 1개?

List 또는 Array

- 시퀀스 자료형, 여러 데이터들의 집합
- Int, Float 같은 다양한 데이터 Type 포함



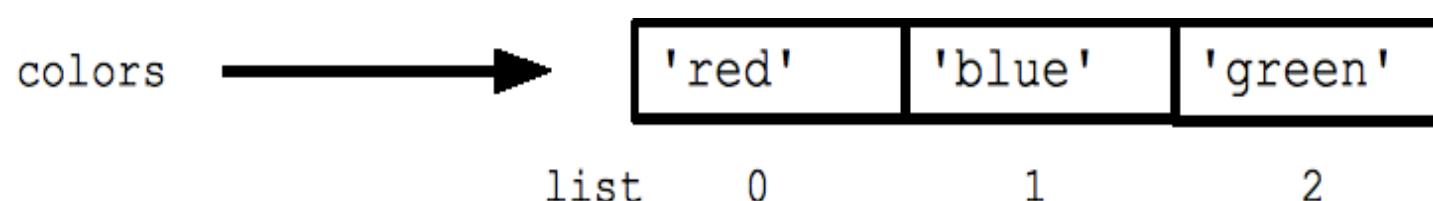
Source: <http://goo.gl/q4VvB1>, <http://goo.gl/JMbHm0>

인덱싱 (Indexing)

list에 있는 값들은 주소(offset)를 가짐,
주소를 사용해 할당된 값을 호출

```
colors = ['red', 'blue', 'green']
print(colors[0])    # red
print(colors[2])    # green
print(len(colors)) # 3
# len은 list의 길이를 반환
```

Editor



Source: <http://goo.gl/q4VvB1>, <http://goo.gl/JMbHm0>

슬라이싱 (Slicing)

- list의 값을 잘라서 쓰는 것이 슬라이싱
- list의 주소 값을 기반으로 부분 값을 반환

Editor

```
cities = ['서울', '부산', '인천', '대구', '대전', '광주', '울산', '수원']
print (cities[0:6], " AND ", a[-9:]) # a 번수의 0부터 5까지, -9부터 끝까지
print (cities[:])    # a변수의 처음부터 끝까지
print (cities[-50:50]) # 범위를 넘어갈 경우 자동으로 최대 범위를 지정
print (cities[::2], " AND ", a[::-1]) # 2칸 단위로, 역으로 슬라이싱
```

리스트의 연산

- 인덱싱, 슬라이싱, 연산 등 활용

```
>>> color = ['red', 'blue', 'green']
>>> color2 = ['orange', 'black', 'white']
>>> print (color + color2)      # 두 리스트 합치기
>>> len(color) # 리스트 길이
>>> color[0] = 'yellow' # 0번째 리스트의 값을 변경
>>> print (color * 2)    # color 리스트 2회 반복
>>> 'blue' in color2    # 문자열 'blue'가 color2 존재 여부 반환
>>> total_color = color + color2
```

Python Shell

리스트 추가와 삭제

- **append, extend, insert, remove, del** 등 활용

```
# 이전 장과 연결 돼서 실행

>>> color.append("white")      # 리스트에 "white" 추가
>>> color.extend(["black","purple"]) # 리스트에 새로운 리스트 추가
>>> color.insert(0,"orange")    # 0번째 주소에 "orange" 추가
>>> print (color)
['orange', 'yellow', 'blue', 'green', 'white', 'black', 'purple']

>>> color.remove("white")      # 리스트에 "white" 삭제
>>> del color[0]                # 0번째 주소 리스트 객체 삭제
>>> print (color)
['yellow', 'blue', 'green', 'black', 'purple']
```

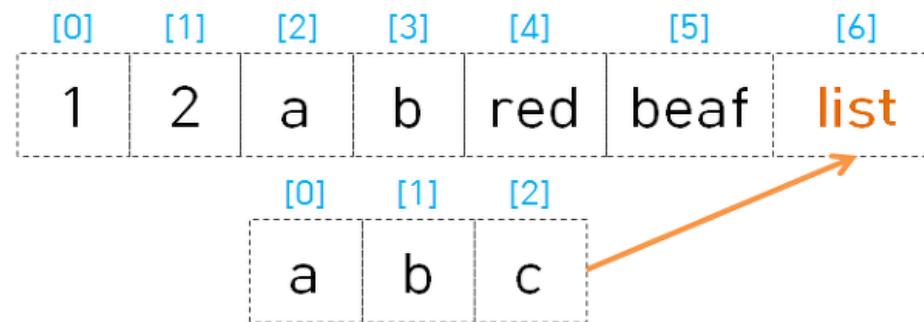
Python Shell

Python 리스트만의 특징

- 다양한 Data Type이 하나에 List에 들어감

```
>>> a = ["color", 1, 0.2]
>>> color = ['yellow', 'blue', 'green', 'black', 'purple']
>>> a[0] = color # 리스트 안에 리스트도 입력 가능
print (a)
[['yellow', 'blue', 'green', 'black', 'purple'],
 1, 0.20000000000000001]
```

Python Shell



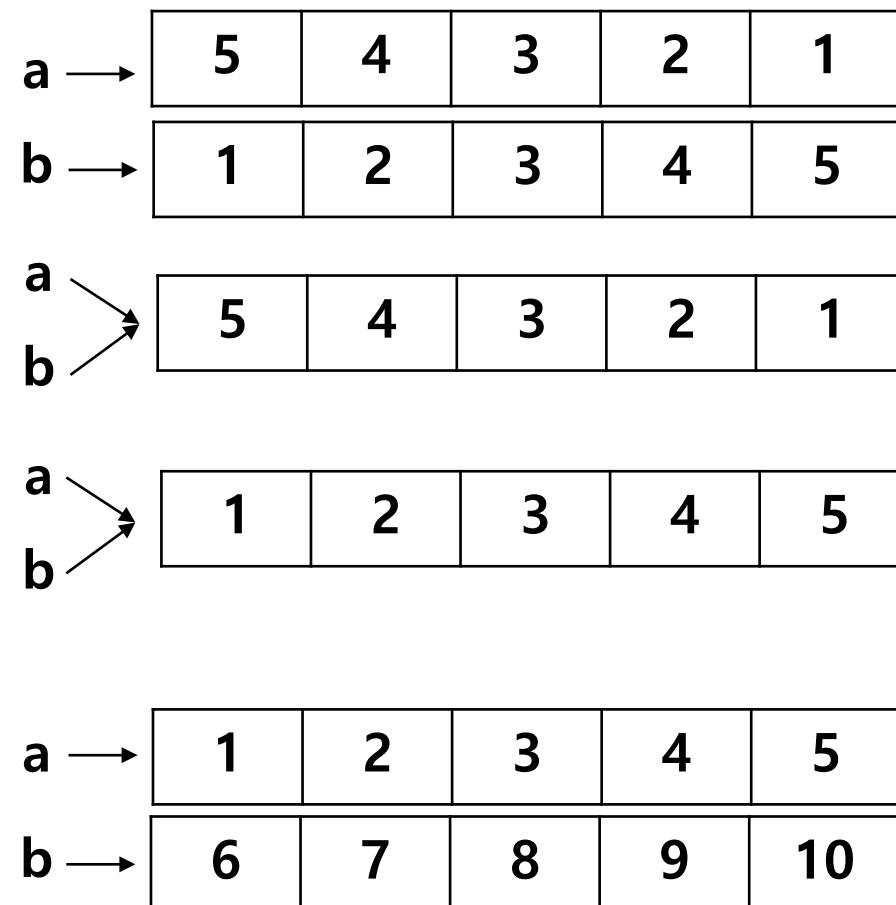
중첩 리스트 시 메모리 구조

Source: <http://goo.gl/FApwnw>

리스트 메모리 저장 방식

```
>>> a = [5, 4, 3, 2, 1]
>>> b = [1, 2, 3, 4, 5]
>>> b = a
>>> print (b)
[5, 4, 3, 2, 1]
>>> a.sort()
>>> print (b)
[1, 2, 3, 4, 5]
>>> b = [6,7,8,9,10]
>>> print (a, b)
[1, 2, 3, 4, 5] [6, 7, 8, 9, 10]
```

Python Shell



“=”의 의미는 같다가 아닌 메모리 주소에 해당 값을 할당(연결)한다는 의미

패킹과 언패킹

- 패킹 : 한 변수에 여러 개의 데이터를 넣는 것
- 언패킹 : 한 변수의 데이터를 각각의 변수로 반환

```
>>> t = [1, 2, 3]          # 1,2,3을 변수 t에 패킹  
  
>>> a , b , c = t        # t에 있는 값 1, 2, 3 을 변수 a, b, c에 언패킹  
  
>>> print(t, a, b, c)   # [1, 2, 3] 1 2 3  
[[1,2,3] 1 2 3]
```

Python Shell

이차원 리스트

- 리스트 안에 리스트를 만들어 행렬(Matrix) 생성

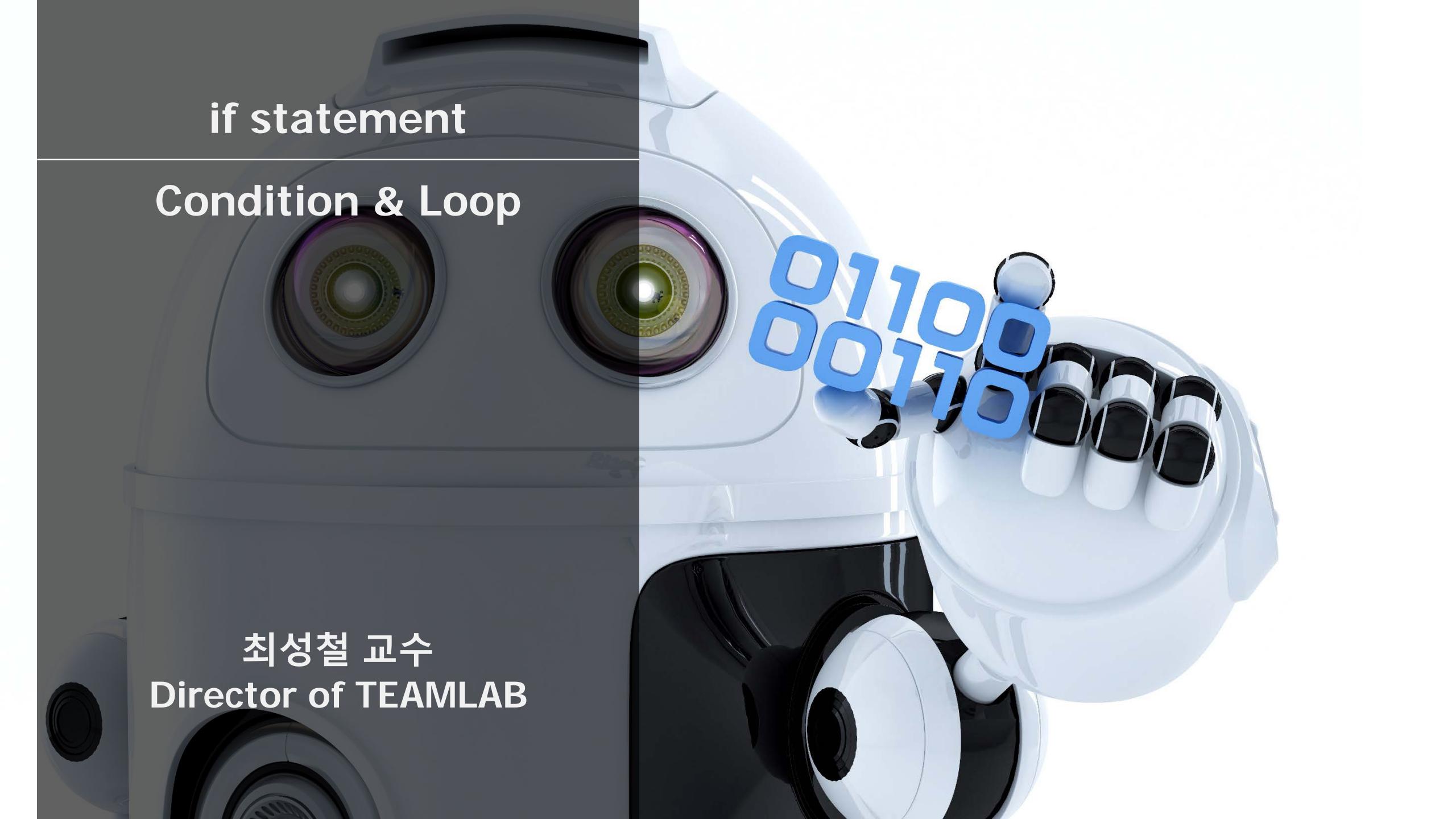
```
>>> kor_score = [49,79,20,100,80]
>>> math_score = [43,59,85,30, 90]
>>> eng_score = [49,79,48,60,100]
>>> midterm_score = [kor_score, math_score, eng_score]
>>> print (midterm_score[0][2])
20
```

Python Shell

	A	B	C	D	E
국어점수	49	79	20	100	80
수학점수	43	59	85	30	90
영어점수	49	79	48	60	100



Human knowledge belongs to the world.



if statement

Condition & Loop

최성철 교수

Director of TEAMLAB

학점 프로그램을 개발해 보자!

점수	학점
38	
37	
7	
16	
95	
71	
63	
48	
49	
66	
37	

- ① 점수에 따른 학점의 기준을 만든다
예) 95점 이상 A+, 60점 미만 F
- ② 기준을 바탕으로 첫번째 줄의 점수를 판단한다
예) 38점은 60점 미만이므로 F
- ③ 다음 줄로 계속 이동하면서 ②를 반복한다
예) 37점은 60점 미만이므로 F
- ④ 더 이상 점수가 없을 때 프로그램을 종료한다.

프로그램 작성 시,
조건에 따른 판단과 반복은 필수

조건문이란?

조건에 따라 특정한 동작을 하게하는 명령어

프로그램 예시 in 생활

- 지하철 앞 차 간격이 10M 이하면 속도를 10km 이하로 늦춰라
- 사용자가 20세 이하면 VOD를 플레이 하지 마라
- 휴대폰 패턴이 5회 틀리면 20초 동안 대기 상태로 만들어라

조건문은 조건을 나타내는 기준과 실행해야 할 명령으로 구성됨

조건의 참, 거짓에 따라 실행해야 할 명령이 수행되거나 되지 않음

파이썬은 조건문으로 if, else, elif 등의 명령 키워드를 사용함

if-else문

가장 기본적인 조건문으로 조건에 따른 명령을 실행

Editor

```
print ("Tell me your age?")
myage = int(input()) # 나이를 입력 받아 myage 변수에 할당
if myage < 30:       # myage 가 30 미만일 때
    print ("Welcome to the Club")
else:                # myage 가 30 이상일 때
    print ("Oh! No. You are not accepted.")
```

입력 받은 값이 “30미만” 이라는 조건에 따른 명령 실행

if-else문 문법

```
if <조건>:          # if를 쓰고 조건 삽입 후 ":" 입력  
    <수행 명령1-1> # 들여쓰기(indentation)후 수행명령 입력  
    <수행 명령1-2> # 같은 조건하에 실행일 경우 들여쓰기 유지  
  
else:                # 조건이 불일치할 경우 수행할 명령 block  
    <수행 명령2-1> # 조건 불일치 시 수행할 명령 입력  
    <수행 명령2-2> # 조건 불일치 시 수행할 명령 들여쓰기 유지
```

- ① 조건 판단 방법
- ② 조건 일치 시 수행 명령 block ":"와 들여쓰기
- ③ 조건 불일치 시 수행 명령 block

조건 판단 방법

- if 다음에 조건을 표기하여 참 또는 거짓을 판단함
- 참/거짓의 구분을 위해서는 비교 연산자를 활용

비교연산자	비교상태	설명
$x < y$	~보다 작음	x과 y보다 작은지 검사
$x > y$	~ 보다 큼	x과 y보다 큰지 검사
$x == y$	같음	x와 y과 같은지 검사 (값과 메모리 주소)
$x \text{ is } y$		
$x != y$	같지 않음	x와 y과 다른지 검사 (값과 메모리 주소)
$x \text{ is not } y$		
$x >= y$	크거나 같음	x과 y보다 이상인지 검사
$x <= y$	작거나 같음	x과 y보다 이하인지 검사

조건 참/거짓의 구분

- 숫자형의 경우는 수학에서의 참/거짓과 동일
- 컴퓨터는 존재하면 참 없으면 거짓이라고 판단함

```
>>> if 1:  
...     print "True"  
... else:  
...     print "False"  
  
...  
True
```

python shell

마찬가지로 if “abc”: 는 참, if “”: 은 거짓임

논리 키워드 사용: and, or, not

- 조건문을 표현할 때 집합의 논리 키워드를 함께 사용하여 참과 거짓을 판단하기도 함

a = 8, b = 5 일 때

if a == 8 and b == 4 # 거짓

if a > 7 or b > 7 # 참

if not (a > 7) # 거짓, a>7인 것이 참 이므로 거짓으로 판단됨

조건 판단 연습 (1/3)

다음 프로그램 수행 결과는?

```
if score >= 90:  
    grade = 'A'  
if score >= 80:  
    grade = 'B'  
if score >= 70:  
    grade = 'C'  
if score >= 60:  
    grade = 'D'  
if score < 60:  
    grade = 'F'  
print grade
```

score	grade
38	
37	
7	
16	
95	
71	
63	
48	
49	
66	
37	

조건 판단 연습 (2/3)

다음 프로그램 수행 결과는?

```
if score >= 90:  
    grade = 'A'  
if score >= 80:  
    grade = 'B'  
if score >= 70:  
    grade = 'C'  
if score >= 60:  
    grade = 'D'  
if score < 60:  
    grade = 'F'  
print grade
```

score	grade
38	F
37	F
7	F
16	F
95	D
71	D
63	D
48	F
49	F
66	D
37	F

조건 판단 연습 (3/3)

```
if score >= 90:  
    grade = 'A'  
if score >= 80:  
    grade = 'B'  
if score >= 70:  
    grade = 'C'  
if score >= 60:  
    grade = 'D'  
if score < 60:  
    grade = 'F'  
print grade
```

- 모든 if문을 순차적으로 실행
- 95는 90초과지만 동시에 60초과이기도 하므로, 마지막 조건문에 따라 grade 값에 "D"가 할당됨
- 이 문제를 해결하기 위해 **elif**와 **else** 구문이 사용됨

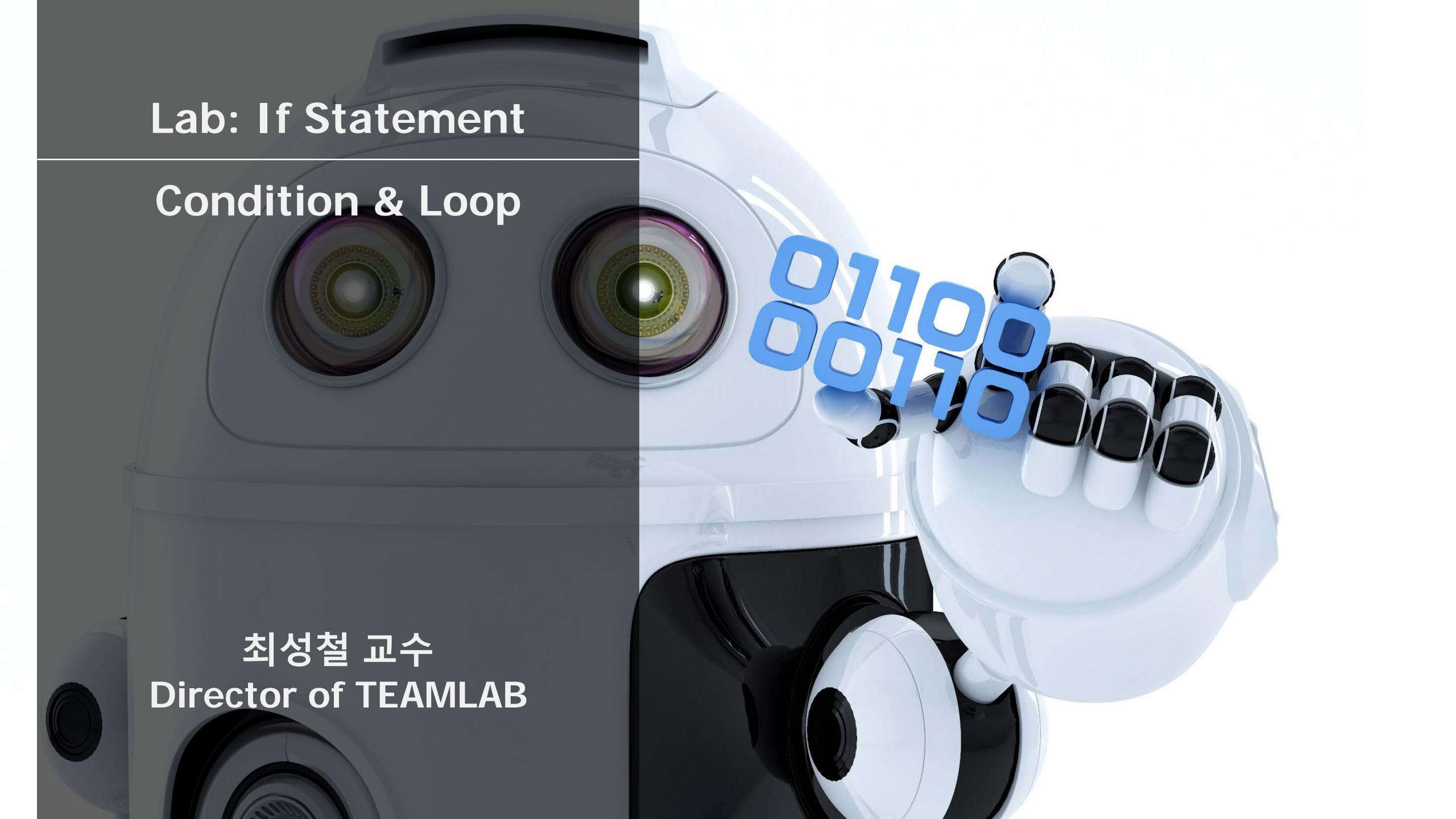
조건 판단 연습 수정

```
if score >= 90: grade = 'A'      # 90 이상일 경우 A
elif score >= 80: grade = 'B'    # 80 이상일 경우 B
elif score >= 70: grade = 'C'    # 70 이상일 경우 C
elif score >= 60: grade = 'D'    # 60 이상일 경우 D
else: grade = 'F'                # 모든 조건에 만족하지 못할 경우 F
```

- 수행할 명령문이 한 줄이면 붙여쓰기 가능
- else 키워드 뒤엔 조건 삭제



Human knowledge belongs to the world.

A white humanoid robot with a black base. It has two large, circular, yellow and green lenses for eyes. Its right hand is extended towards the viewer, with blue 3D-rendered binary digits (01100110) floating around it. The robot's body is mostly white with some black accents.

Lab: If Statement

Condition & Loop

최성철 교수

Director of TEAMLAB

[연습] 무슨 학교 다니세요?

태어난 연도를 계산하여 학교 종류를 맞추는 프로그램

당신이 태어난 연도를 입력하세요

Terminal

1994

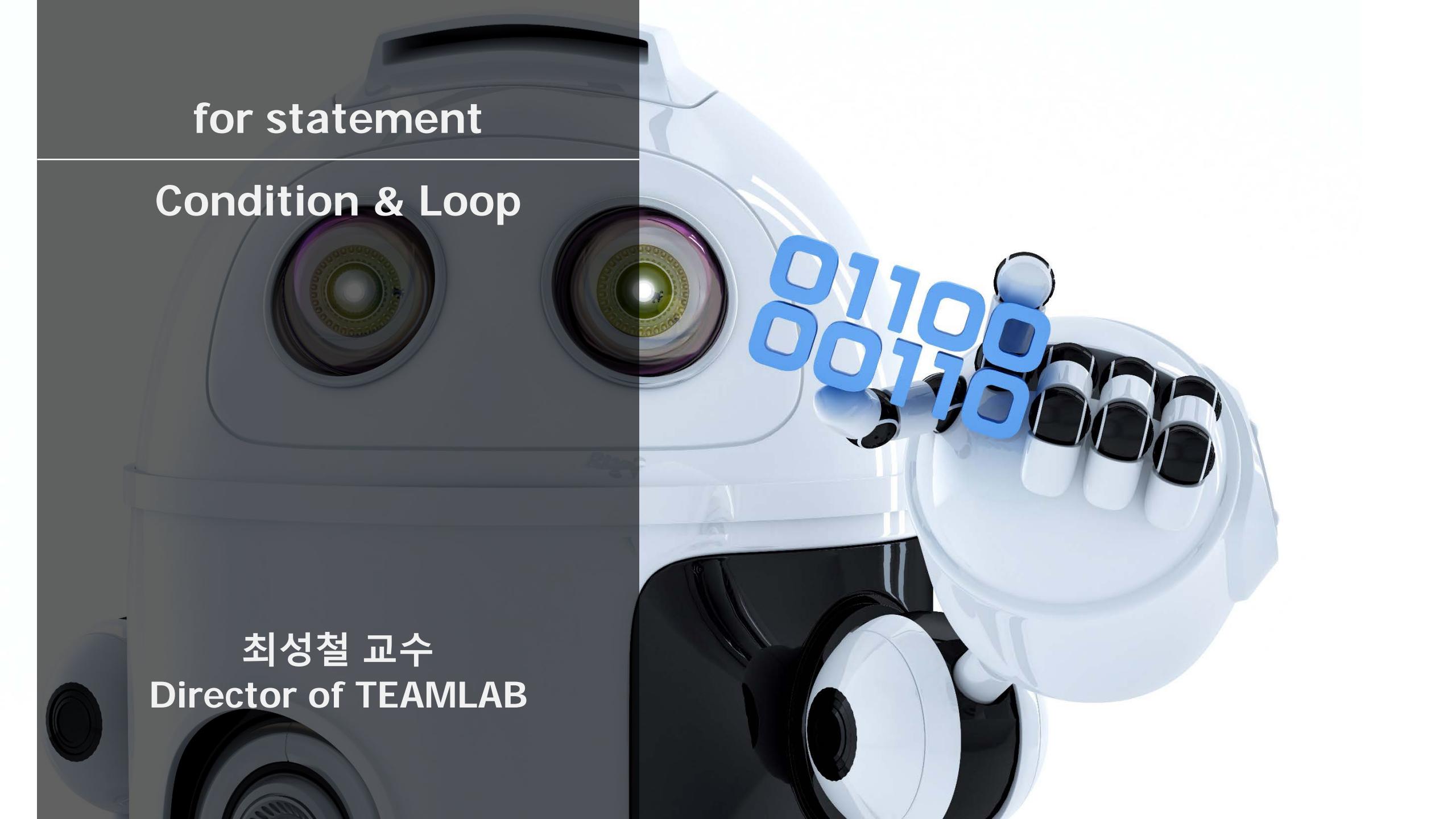
자신이 태어난 연도 입력

대학생

- 나이는 $2017 - \text{태어난 연도} + 1$ 로 계산
- 26세 이하 20세 이상 이면 “대학생”, 20세 미만 17세 이상 이면 “고등학생”
17세 미만 14세 이상 이면 “중학생”, 14세 미만 8세 이상이면 “초등학생”
그 외의 경우는 “학생이 아닙니다” 출력



Human knowledge belongs to the world.

The background features a white humanoid robot with large, expressive eyes and a visible mechanical arm. The robot's hand is held up, displaying the binary code "01100110" in blue. The robot is set against a dark, blurred background.

for statement

Condition & Loop

최성철 교수

Director of TEAMLAB

반복문이란?

정해진 동작을 반복적으로 수행하게 하는 명령문

프로그램 예시 in 생활

- 100명의 학생에 성적을 산출할 때
- 왓챠에서 영화 추천하기
- 워드에서 단어 바꾸기 명령 실행

반복문은 **반복 시작 조건**, **종료 조건**, **수행 명령**으로 구성됨

반복문 역시 반복 구문은 **들여쓰기**와 **block**으로 구분됨

파이썬은 반복문으로 **for**, **while** 등의 명령 키워드를 사용함

for문 (1/2)

기본적인 반복문, 반복 범위를 지정하여 반복문 수행

```
for looper in [1,2,3,4,5]:  
    print ("hello")
```

- ① looper 변수에 1 할당
- ② "Hello" 출력
- ③ 리스트(대괄호속 숫자들) 있는 값 차례로 looper 할당
- ④ 5까지 할당한 후 반복 block 수행 후 종료

```
for looper in [1,2,3,4,5]:  
    print (looper)
```

만약 100번 반복해야 하는 프로그램을 짜야 한다면?

for문 (2/2)

range () 사용하기

Editor

```
for looper in [1,2,3,4,5]:  
    print ("hello")  
for looper in range(0,5):  
    print "hello"
```

왜 range(1,5) 과 아닌 range (0,5) 인가?

: range()는 마지막 숫자 바로 앞까지 리스트를 만들어줌

즉, range(1,5) = [1,2,3,4] 까지 같은 의미

※ range(0,5) = [0,1,2,3,4] = range(5)는 같은 의미

[알아두면 상식] 반복문

반복문 변수명

- ✓ 임시적인 반복 변수는 대부분 i, j, k로 정함
- ✓ 이것은 수학에서 변수를 x, y, z로 정하는 것과 유사한 관례

0부터 시작하는 반복문

- ✓ 반복문은 대부분 0부터 반복을 시작
- ✓ 이것도 일종의 관례로 1부터 시작하는 언어도 존재
- ✓ 2진수가 0부터 시작하기 때문에 0부터 시작하는 걸 권장

무한 loop

- ✓ 반복 명령이 끝나지 않는 프로그램 오류
- ✓ CPU와 메모리 등 컴퓨터의 리소스를 과다하게 점유

for문의 다양한 반복문 조건 표현 (1/2)

문자열을 한자씩 리스트로 처리

```
for i in 'abcdefg':  
    print (i)
```

Editor

각각의 문자열 리스트로 처리

```
for i in ['Americano', 'latte', 'frafuchino']:  
    print (i)
```

Editor

for문의 다양한 반복문 조건 표현 (2/2)

간격을 두고 세기

```
for i in range(1, 10, 2):
    # 1부터 10까지 2씩 증가시키면서 반복문 수행
    print (i)
```

Editor

역순으로 반복문 수행

```
for i in range(10, 1, -1):
    # 10부터 1까지 -1씩 감소시키면서 반복문 수행
    print (i)
```

Editor

while문

조건이 만족하는 동안 반복 명령문을 수행

```
i = 1  
while i < 10:  
    print (i)  
    i += 1
```

- ① i 변수에 1 할당
- ② i가 10 미만인지 판단
- ③ 조건에 만족할 때 i 출력, i에 1을 더함
- ④ i가 10이 되면 반복 종료

Editor

for문은 while문으로 변환 가능

반복 실행횟수를 명확히 알 때

```
for i in range(0,5):  
    print (i)
```

Editor

반복 실행횟수가 명확하지 않을 때

```
i = 0  
while i < 5:  
    print (i)  
    i = i + 1
```

Editor

반복의 제어 – break, continue

break 특정 조건에서 반복 종료

Editor

```
for i in range(10):
    if i == 5: break      # i가 5가 되면 반복 종료
    print (i)
print ("EOP")            # 반복 종료 후 "EOP" 출력
```

continue 특정 조건에서 남은 반복 명령 skip

Editor

```
for i in range(10):
    if i == 5: continue  # i가 5가 되면 i를 출력하지 않음
    print (i)
print ("EOP")            # 반복 종료 후 "EOP" 출력
```

반복의 제어 – else

반복 조건이 만족하지 않을 경우 반복 종료 시 1회 수행

```
for i in range(10):
    print (i),
else:
    print ("EOP")
```

Editor

```
i =0
while i < 10:
    print (i,)
    i += 1
else:
    print ("EOP")
```

Editor

※ break로 종료된 반복문은 else block이 수행되지 않음



Human knowledge belongs to the world.

Lab: Multiple Table Game

Condition & Loop

최성철 교수

Director of TEAMLAB

01100
00110
01100

[연습] 구구단 계산기

아래와 같이 출력되는 프로그램을 만드시오

구구단 몇단을 계산할까요?

CMD

5

구구단 5단을 계산합니다.

$5 \times 1 = 5$

$5 \times 2 = 10$

$5 \times 3 = 15$

.....

$5 \times 8 = 40$

$5 \times 9 = 45$

Loop Review

```
sentence = "I love you"  
reverse_sentence = "  
for char in sentence:  
    reverse_sentence = char + reverse_sentence  
print (reverse_sentence)
```

<u>Loop</u>	<u>reverse_sentence¹</u>	<u>reverse_sentence²</u>	<u>char</u>
0			
1			
2			
3		ol	o
4	ol	vol	v
5	vol	evol	e
6	evol	evol	
7	evol	y evol	y
8	y evol	oy evol	o
9	oy evol	uooy evol	u

```

decimal = 10
result = ""
while (decimal > 0):
    remainder = decimal % 2
    decimal = decimal / 2
    result = str(remainder) + result
print (result)

```

수식

$$\begin{array}{r}
 2 \longdiv{10} \cdots 0 \\
 2 \longdiv{5} \cdots 1 \\
 2 \longdiv{2} \cdots 0 \\
 2 \longdiv{1}
 \end{array}$$

<u>Loop</u>	<u>decimal¹</u>	<u>remainder</u>	<u>decimal²</u>	<u>result¹</u>	<u>result²</u>
0	10	0	5		0
1	5	1	2	0	10
2	2	0	1	10	010
3	1	1	0	010	1010

Debugging Loop

```
print ("input decimal number: ")
decimal = int(input())
result = ""
loop_counter = 0
while (decimal > 0):
    temp_decimal_input = decimal
    temp_result_input = result

    remainder = decimal % 2
    decimal = decimal // 2
    result = str(remainder) + result

    print ("-----", loop_counter, "loop value check -----")
    print ("Initial decimal:", temp_decimal_input,
          ", Remainder:", remainder,
          ", Initial result", temp_result_input)
    print ("Output decimal:", decimal,
          "Output result:", result)
    print ("-----")
    print ("")  
  
    loop_counter += 1
print ("Binary number is", result)
```

binary_converter.py

Editor

Loop 내에 변수들의 값을
Print문으로 확인



Human knowledge belongs to the world.

Lab: Condition and Loop

Condition & Loop

최성철 교수
Director of TEAMLAB

01100
00110

가변적인 중첩 반복문 (variable nested loops)

실제 프로그램에서는 반복문은
사용자의 입력에 따라 가변적으로 반복되고
하나의 반복이 아닌 중복되어 반복이 일어남

[연습] 숫자 찾기 게임

1~100 임의의 숫자를 맞추시오

guess_number.py

```
# -*- coding: utf-8 -*-
import random          # 난수 발생 함수 호출
guess_number = random.randint(1, 100)    # 1~100 사이 정수 난수 발생
print ("숫자를 맞춰보세요 (1 ~ 100)")
users_input = int(input())      # 사용자 입력을 받음
while (users_input is not guess_number):    # 사용자 입력과 난수가 같은지 판단
    if users_input > guess_number:        # 사용자 입력이 클 경우
        print ("숫자가 너무 큽니다")
    else:                                # 사용자 입력이 작은 경우
        print ("숫자가 너무 작습니다")
    users_input = int(input())      # 다시 사용자 입력을 받음
else: print ("정답입니다. ", "입력한 숫자는 ", users_input , "입니다")  # 종료 조건
```

Editor

[연습] 연속적인 구구단 입력

1~9 입력 받아 구구단을 출력, 0을 입력 시 종료

구구단 몇 단을 계산할까요(1~9)?

5

구구단 5단을 계산합니다.

$5 \times 1 = 5$

$5 \times 2 = 10$

$5 \times 3 = 15$

.....

$5 \times 8 = 40$

$5 \times 9 = 45$

구구단 몇 단을 계산할까요(1~9)?

10

0

구구단 게임을 종료합니다

[정답] 연속적인 구구단 입력

nested_gugudan.py

```
print ("구구단 몇 단을 계산할까요(1~9)?")
```

```
x = 1
```

```
while (x is not 0):
```

```
    x = int(input())
```

```
    if x == 0: break
```

```
    if not(1 <= x <= 9):
```

```
        print ("잘못 입력하셨습니다", "1부터 9사이 숫자를 입력해주세요")
```

```
        continue
```

```
else:
```

```
    print ("구구단 " + str(x) + "단을 계산합니다.")
```

```
    for i in range(1,10):
```

```
        print (str(x) + " X " + str(i) + " = " + str(x*i))
```

```
    print ("구구단 몇 단을 계산할까요(1~9)?")
```

```
print ("구구단 게임을 종료합니다")
```

Editor

전체 loop

0이 입력될 때까지
게임이 실행됨

구구단 loop

구구단 계산용 loop

[연습] 이차원 리스트 처리하기

사람 별 평균을 구하라

```
kor_score = [49, 79, 20, 100, 80]
math_score = [43, 59, 85, 30, 90]
eng_score = [49, 79, 48, 60, 100]
midterm_score = [kor_score, math_score, eng_score]
print(midterm_score[0][2])
```

	A	B	C	D	E
국어점수	49	79	20	100	80
수학점수	43	59	85	30	90
영어점수	49	79	48	60	100

[연습] 이차원 리스트처리하기

two_dim_list_loop.py

Editor

```
student_score = [0,0,0,0,0]
i = 0
for subject in midterm_score:
    for score in subject:
        student_score[i] += score      # 각 학생마다 개별로 교과 점수를 저장
        i += 1                         # 학생 index 구분
    i = 0                            # 과목이 바뀔 때 학생 인덱스 초기화
else:
    a, b, c, d, e = student_score      # 학생 별 점수를 unpacking
    student_average = [a/3,b/3,c/3,d/3,e/3] #
print (student_average)
```



Human knowledge belongs to the world.



How to debug code

Condition & Loop

최성철 교수
Director of TEAMLAB

01100
00110



JORGE CHAM © 2005

www.phdcomics.com

<http://phdcomics.com/>

Debugging (디버깅)

- 코드의 오류를 발견하여 수정하는 과정
- 오류의 '원인'을 알고 '해결책'을 찾아야 함
- 문법적 에러를 찾기 위한 에러 메시지 분석
- 논리적 에러를 찾기 위한 테스트도 중요

문법적 에러

초보자에게 흔히 생기는 문법 오류

들여쓰기 (Indentation Error)

```
x = 2  
y = 5 # IndentationError  
print (x+y)
```

초보자에게 흔히 생기는 문법 오류

오탈자

```
pront (x+y) # Not Print, But Pront
```

대소문자 구분 안 함

```
gachon = "ACE"
```

```
print (Gachon) # g는 소문자
```

문법 에러 – Error 메시지 분석

- 에러가 발생하면 파이썬이 알려줌

A screenshot of a terminal window titled "test.py + (~) - VIM". The code in the editor is:

```
1 temperature = float(input("온도를 입력하세요 :"))
2 print(temperature)
```

The terminal output shows the command "python3.4 test.py" followed by an error message:

```
python3.4 test.py
File "test.py", line 1
    temperature = float(input("온도를 입력하세요 :"))
          ^
IndentationError: unexpected indent
```

Annotations in Korean explain the error:

- An arrow points from the text "test.py 라는 파일 첫번째 줄에" to the first line of the code.
- A callout box highlights the line "temperature = float(input("온도를 입력하세요 :"))" with the text "이 부분에 에러가 있어요."
- An arrow points from the text "들여쓰기가 문제가 있네요" to the error message "IndentationError: unexpected indent".

Indentation Error – 흔히 발생하는 에러
들여쓰기를 **Space**로 했나 **Tab**으로 했냐?

문법 에러 – Error 메시지 분석

The screenshot shows a terminal window with two tabs. The top tab is titled "test.py (~) - VIM" and contains the following Python code:

```
1 temperature = float(input("온도를 입력하세요 :"))
2 print (temperatre) temperature != temperatre
```

The bottom tab is titled "cs50@cs50: ~" and shows the output of running the script:

```
cs50@cs50:~$ python3.4 test.py
온도를 입력하세요 :10.3
Traceback (most recent call last):
  File "test.py", line 2, in <module>
    print (temperatre)
               ^      이 부분에 에러가 있어요.
  NameError: name 'tempearatre' is not defined
cs50@cs50:~$
```

Annotations in Korean explain the error:

- An arrow points from the word "temperatre" in the code to the word "tempearatre" in the error message, with the text "이 부분에 에러가 있어요." (There is an error in this part).
- An arrow points from the word "NameError" in the error message to the word "tempearatre", with the text "tempearatre라는 이름이 뭔지 모르겠어요." (I don't know what the name tempearatre means).

오탈자 / 대문자 등에 의한 에러 발생

논리적 에러

논리적 에러

- 논리적 에러 - 뜻대로 실행이 안되는 코드
- 중간 중간 프린터 문을 찍어서 확인
- Loop Review 처럼 해보기!

논리적 에러

<http://dodnet.tistory.com/186>

Trapezoid

Solve for area ▾

$$A = \frac{a+b}{2} h$$

a Base

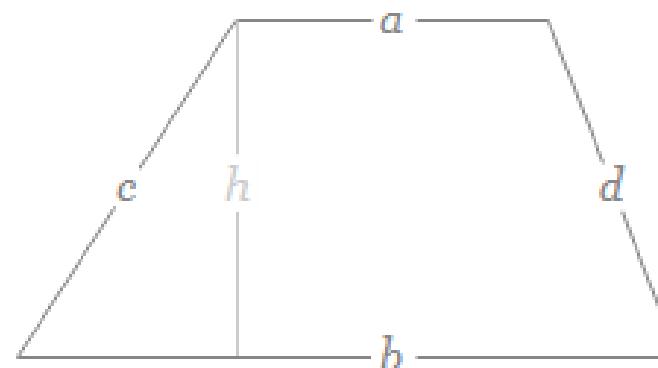
Enter value

b Base

Enter value

h Height

Enter value



논리적 에러 – 함수 Check

```
def addition(x, y):  
    return x+y
```

Editor

```
def multiplication(x, y):  
    return x*y
```

```
def divided_by_2(x):  
    return x/2
```

trapezium_area.py

```
>>> import trapezium_area as ta  
#trapezium_area 파일을 ta라는 이름으로 부름  
>>> ta.addition(10,5)  
15  
>>> ta.multiplication(10,5)  
50  
>>> ta.divided_by_2(50)  
25  
>>>
```

Python Shell

논리적 에러 – 함수 Check Print문

```
def addition(x, y):
```

```
    return x+y
```

Editor

```
def multiplication(x, y):
```

```
    return x*y
```

```
def divided_by_2(x):
```

```
    return x/2
```

Python Shell에서 호출 할 경우 실행되지 않음

```
if __name__ == '__main__':
```

```
    print(addition(10,5))
```

```
    print(multiplication(10,5))
```

```
    print(multiplication(50))
```

trapezium_area_test.py

논리적 에러 – 함수 Check Print문

```
python trapezium_area_test.py
```

Editor

```
addtion : 15
```

```
multiplication : 50
```

```
divided_by_2 : 25.0    #결과가 올바로 출력 됐는지 확인 가능
```

※ if __name__ == "__main__" 차이

```
>>> import trapezium_area_test
```

있을 경우

```
>>>
```

```
>>> import trapezium_area_test
```

없을 경우

```
addtion : 15
```

```
multiplication : 50
```

```
divided_by_2 : 25.0
```

함수 합치기 – 사다리꼴 넓이

```
def addition(x, y):
    return x+y
def divided_by_2(x):
    return x/2

def main():
    base_line = float(input("밑변의 길이는?"))
    upper_edge = float(input("윗변의 길이는?"))
    height = float(input("높이는?"))

    print("넓이는 :", divided_by_2(addition(base_line, upper_edge) *
                                   height))

if __name__ == "__main__":
    main()
```

Editor

스스로 해결 해 보기

인터넷에 물어보기

- 모든 문제는 Google + stack overflow 로 해결 가능
- stack overflow: 전 세계 개발자들의 코딩 Q&A 사이트
코딩계의 네이버 지식인

인터넷에 물어보기

The screenshot shows the Stack Overflow homepage. At the top, there's a navigation bar with links for sign up, log in, tour, help, and a search bar. Below the navigation is the Stack Overflow logo and a menu bar with Questions, Jobs, Documentation Beta, Tags, Users, Badges, and Ask Question.

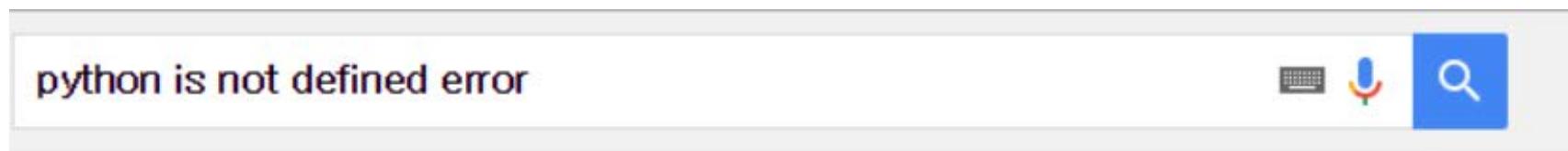
A prominent blue banner at the top left announces "Announcing Stack Overflow Documentation". It says, "We started with Q&A. Technical documentation is next, and we need your help. Whether you're a beginner or an experienced developer, you can contribute." It includes two buttons: "Sign up and start helping →" and "Learn more about Documentation →". To the right of the banner is a screenshot of a software interface with multiple windows and code snippets.

The main content area features a "Top Questions" section. The first question listed is "Wireless Communication and Network Research" with 1 view, 0 votes, and 0 answers. It was asked 6 secs ago by Abdulhameed 49. The second question is "Bootstrap, Modal with link inside data-content once clicked close the current modal and shows another modal?" with 8 views, 0 votes, and 1 answer. It was answered 35 secs ago by Bùi văn Nguyễn 129. The third question is "Hi i'm using swrevealviewcontroller and pushing to other view controller by using the bellow code" with 4 views, 0 votes, and 0 answers. It was modified 1 min ago by NDoc 7,339. There are also sections for "interesting" questions, "454 featured" questions, and filters for hot, week, and month.

On the right side, there's a sidebar titled "Looking for a job?" featuring several job listings:

- Senior Software Engineer** at Buzzvil in Seoul, South Korea, with a salary range of \$50,000 - \$60,000. The listing includes Python as a required skill.
- Full Stack Developer - \$60k** at Crossover with no office location, labeled as REMOTE. The listing includes Python and react-native as required skills.
- 웹서비스 백엔드 개발자 (Web Service Back-End Developer)** at IDK2 (아이디케이스퀘어드) in Seoul, South Korea. The listing includes AWS and node.js as required skills.

인터넷에 물어보기



검색결과 약 12,400,000개 (0.50초)

관련검색: name is not defined python python __file__ is not defined
python global name is not defined global name is not defined python class
global name is not defined python import

[Python input\(\) error - NameError: name '...' is not defined - Stack ...](#)

stackoverflow.com/.../python-input-error-nameerror-name-is-not... ▾ 이 페이지 번역하기

2014. 1. 14. - I am getting an error when I try to run this simple python script: ... TL;DR. input function in Python 2.7, evaluates whatever you enter, as a Python ...

인터넷에 물어보기

Python input() error - NameError: name '...' is not defined



▲ I am getting an error when I try to run this simple python script:

43
▼

```
input_variable = input ("Enter your name: ")
print ("your name is" + input_variable)
```

★
20
Lets say I type in "dude", the error I am getting is:

```
line 1, in <module>
input_variable = input ("Enter your name: ")
File "<string>", line 1, in <module>
NameError: name 'dude' is not defined
```

I am running Mac OS X 10.9.1 and I am using the Python Launcher app that came with the install of python 3.3 to run the script.

Edit: I realized I am somehow running these scripts with 2.7. I guess the real question is how do I run my scripts with version 3.3? I thought if I dragged and dropped my scripts on top of the Python Launcher app that is inside the Python 3.3 folder in my applications folder that it would launch my scripts using 3.3. I guess this method still launches scripts with 2.7. So How do I use 3.3?

인터넷에 물어보기



You are running Python 2, not Python 3. For this to work in Python 2, use `raw_input`.



```
input_variable = raw_input ("Enter your name: ")
print ("your name is" + input_variable)
```

[share](#) [improve this answer](#)

[edited Mar 6 at 21:03](#)

[answered Jan 14 '14 at 19:53](#)



davidism

32.2k • 10 • 45 • 71



llamadillo

171 • 1 • 10

[add a comment](#)



Since you are writing for Python 3.x, you'll want to begin your script with:



```
#!/usr/bin/env python3
```



If you use:

```
#!/usr/bin/env python
```

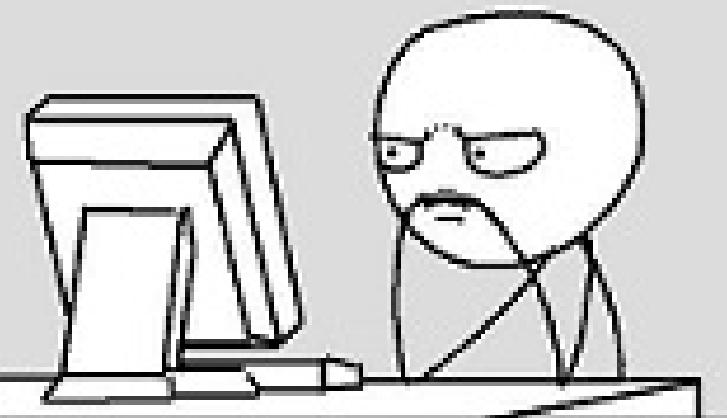
It will default to Python 2.x. These go on the first line of your script, if there is nothing that starts with `#!` (aka the shebang).

If your scripts just start with:

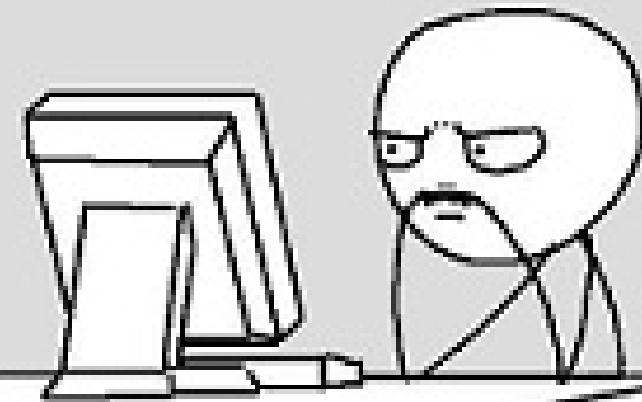
추천 사이트

- 점프 투 파이썬 - <https://wikidocs.net/book/1>
- 코드카데미 - <https://www.codecademy.com/>
- 코드파이트 - <https://codefights.com/>
- 생활 코딩 - <https://opentutorials.org/course/1750>
- Coursera – python 검색

It doesn't work..... why?



It works..... why?



<http://goo.gl/PSvBDp>



Human knowledge belongs to the world.

Function Concepts

Function

최성철 교수
Director of TEAMLAB

01100
00110

**프로그램을 여럿이
개발할 경우 코드를
어떻게 작성해야 할까?**

[생각해보기]

프로그램을 여럿이 개발할 경우 코드를 어떻게 작성해야 할까?

- ① 다같이 모여서 토론하며 한 줄 한 줄
- ② 제일 잘하는 사람이 혼자 작성
- ③ 필요한 부분을 나눠서 작성한 후 합침

[생각해보기]

프로그램을 기능별로 나누는 방법

- ① 함수 ② 객체 ③ 모듈

함수

함수 (Function)

어떤 일을 수행하는 코드의 덩어리

```
# 사각형의 넓이를 구하는 함수  
def calculate_rectangle_area (x , y):  
    return x * y          # 가로, 세로를 곱해서 반환
```

- 반복적인 수행을 1회만 작성 후 호출
- 코드를 논리적인 단위로 분리
- 캡슐화: 인터페이스만 알면 타인의 코드 사용

함수 선언 문법

함수 이름, parameter, return value(optional)

```
def 함수 이름 (parmaeter #1 ... ):  
    수행문 #1(statements)  
    수행문 #2(statements)  
    return <반환값>
```

함수 선언 예시

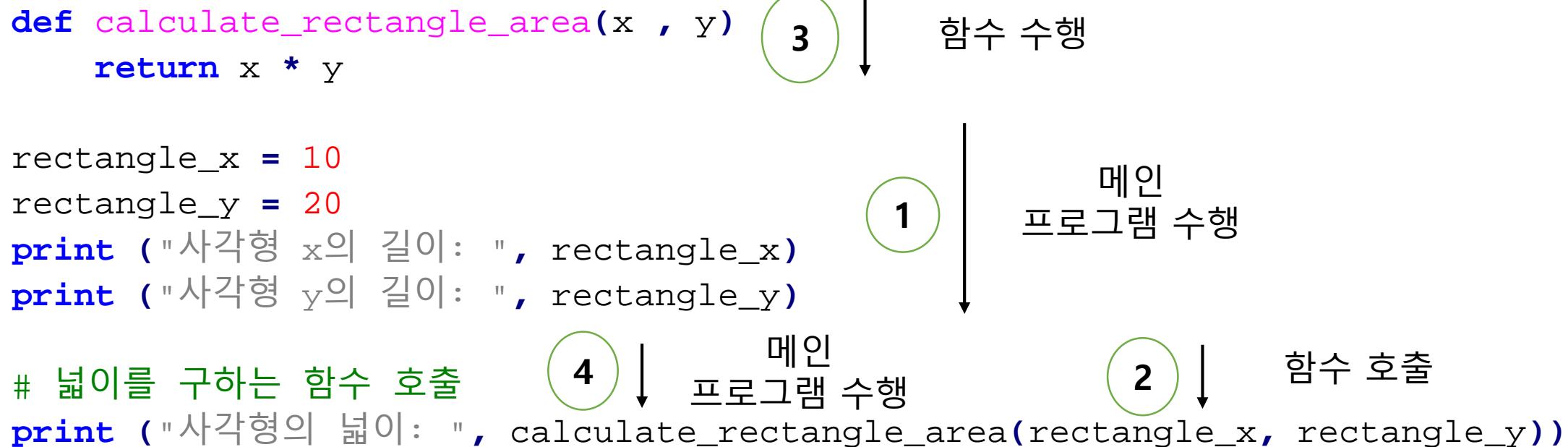
```
def calculate_rectangle_area(x , y)
    return x * y

rectangle_x = 10
rectangle_y = 20
print ("사각형 x의 길이: ", rectangle_x)
print ("사각형 y의 길이: ", rectangle_y)

# 넓이를 구하는 함수 호출
print ("사각형의 넓이: ", calculate_rectangle_area(rectangle_x,
rectangle_y))
```

함수 수행 순서 (w/함수)

- 함수 부분 를 제외한 메인프로그램부터 시작
- 함수 호출 시 함수부분을 수행 후 되돌아옴



[알아두면 상식] 함수 vs 함수 (1/2)

- 프로그래밍의 함수와 수학의 함수는 유사함
- 모두 입력 값과 출력 값으로 이루어짐

$f(x) = 2x + 7$, $g(x) = x^2$ 이고 $x = 2$ 일 때

$f(x) + g(x) + f(g(x)) + g(f(x))$ 의 값은?

$f(2) = 11$, $g(2) = 4$, $f(g(x)) = 15$, $g(f(x))=121$

$11 + 4 + 15 + 121 = 151$

이 공식을 파이썬으로 작성하면?

[알아두면 상식] 함수 vs 함수 (2/2)

```
def f(x):  
    return 2 * x + 7  
  
def g(x):  
    return x ** 2  
  
x = 2  
  
print (f(x) + g(x) + f(g(x)) + g(f(x)))
```

↓ ↓

f(x) 함수 선언 g(x) 함수 선언

Parameter vs. Argument

- Parameter : 함수의 입력 값 인터페이스

```
def f(x):  
    return 2 * x + 7
```

- Argument 실제 Parameter에 대입된 값

```
>>> print(f(2))  
11
```

함수 형태

Parameter 유무, 반환 값(return value)

유무에 따라 함수의 형태가 다름

	Parameter 없음	Parameter 존재
반환 값 없음	함수 내의 수행문만 수행	인자를 사용, 수행문만 수행
반환 값 존재	인자없이, 수행문 수행 후 결과값 반환	인자를 사용하여 수행문 수행 후 결과값 반환

함수 형태 예제

```
def a_rectangle_area():# 인자 x , 리턴 값 x
    print (5 * 7)
def b_rectangle_area(x,y): # 인자 o , 리턴 값 x
    print (x * y)
def c_rectangle_area():      # 인자 x , 리턴 값 o
    return (5 * 7)
def d_rectangle_area(x ,y):# 인자 o , 리턴 값 o
    return (x * y)

a_rectangleArea()
b_rectangleArea(5,7)
print (c_rectangleArea())
print (d_rectangleArea(5,7))
```



Human knowledge belongs to the world.

Function Concepts II

Function

최성철 교수

Director of TEAMLAB

01100
00110



함수 호출 방식

함수 호출 방식 개요 (1/2)

함수의 인자를 전달하는 방식

값에 의한 호출(Call by Value)

참조의 의한 호출(Call by Reference)

함수 호출 방식 개요 (2/2)

Call by Value

함수에 인자를 넘길 때 값만 넘김.

함수 내에 인자 값 변경 시, 호출자에게 영향을 주지 않음

Call by Reference

함수에 인자를 넘길 때 메모리 주소를 넘김.

함수 내에 인자 값 변경 시, 호출자의 값도 변경됨

파이썬 함수 호출 방식(1/2)

파이썬은 **객체의 주소가 함수로 전달되는 방식**

전달된 객체를 참조하여 변경 시 호출자에게 영향을 주나,
새로운 객체를 만들 경우 호출자에게 영향을 주지 않음



파이썬 함수 호출 방식(2/2)

```
def spam(eggs):  
    eggs.append(1) # 기존 객체의 주소값에 [1] 추가  
    eggs = [2, 3] # 새로운 객체 생성  
  
ham = [0]  
spam(ham)  
print(ham) # [0, 1]
```

Function Call Test

```
def test(t):  
    t = 20  
    print ("In Function : ", t)  
  
x = 10  
print ("Before : ", x)      # 10  
test(x)                   # 함수 호출  
print ("After : ", x)      # 10 - 함수 내부의 t는 새로운 주소값을 가짐
```

변수의 범위

변수의 범위 (Scoping Rule)

- 변수가 사용되는 범위 (함수 또는 메인 프로그램)
- 지역변수(local variable) : 함수내에서만 사용
전역변수(Global variable) : 프로그램전체에서 사용

```
def test(t):  
    print(x)  
    t = 20  
    print ("In Function :", t)  
  
x = 10  
test(x)  
print(t)
```

변수의 범위 (Scoping Rule)

- 전역변수는 함수에서 사용가능
- But, 함수 내에 전역 변수와 같은 이름의 변수를 선언하면 새로운 지역 변수가 생김

```
def f():
    s = "I love London!"
    print(s)
```

```
s = "I love Paris!"
f()
print(s)
```

<http://goo.gl/O3vDwy>

변수의 범위 (Scoping Rule)

- 함수 내에서 전역변수 사용 시 `global` 키워드 사용

```
def f():
    global s
    s = "I love London!"
    print(s)

s = "I love Paris!"
f()
print(s)
```

<http://goo.gl/O3vDwy>

변수의 범위 (Scoping Rule)

```
def calculate(x, y):
    total = x + y      # 새로운 값이 할당되어 함수 내 total은 지역변수가 됨
    print ("In Function")
    print ("a:", str(a), "b:", str(b), "a+b:", str(a+b), "total :", str(total))
    return total

a = 5                  # a와 b는 전역변수
b = 7
total = 0              # 전역변수 total
print ("In Program - 1")
print ("a:", str(a), "b:", str(b), "a+b:", str(a+b))

sum = calculate (a,b)
print ("After Calculation")
print ("Total :", str(total), " Sum:", str(sum)) # 지역변수는 전역변수에 영향 x
```

Swap

- 함수를 통해 변수 간의 값을 교환(Swap)하는 함수
- Call By XXXX를 설명하기 위한 전통적인 함수 예시

```
Enter the value of x and y
4
5
Before Swapping
x = 4
y = 5
After Swapping
x = 5
y = 4
```

<http://goo.gl/ZNht49>

Swap

a = [1,2,3,4,5] 일 때 아래 함수 중 실제 swap이 일어나는 함수는?

```
def swap_value (x, y):
    temp = x
    x = y
    y = temp
```

```
def swap_offset (offset_x, offset_y):
    temp = a[offset_x]
    a[offset_x] = a[offset_y]
    a[offset_y] = temp
```

```
def swap_reference (list, offset_x, offset_y):
    temp = list[offset_x]
    list[offset_x] = list[offset_y]
    list[offset_y] = temp
```

Swap

swap_offset: a 리스트의 전역 변수 값을 직접 변경

swap_reference: a 리스트 객체의 주소 값을 받아 값을 변경

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

swap_value(a[1], a[2])
print (a)                      # [1,2,3,4,5]

swap_offset(1,2)
print (a)                      # [1,3,2,4,5]

swap_reference(a, 1, 2)
print (a)                      # [1,3,2,4,5]
```

재귀 함수

재귀함수 (Recursive Function)

- 자기자신을 호출하는 함수
- 점화식과 같은 재귀적 수학 모형을 표현할 때 사용
- 재귀 종료 조건 존재, 종료 조건까지 함수호출 반복

$$n! = n \cdot (n - 1) \cdots 2 \cdot 1 = \prod_{i=1}^n i$$
$$1! = 1$$
$$2! = 2(1) = 2$$
$$3! = 3(2)(1) = 6$$
$$4! = 4(3)(2)(1) = 24$$
$$5! = 5(4)(3)(2)(1) = 120$$

재귀함수 (Recursive Function)

```
def factorial(n):
    if n == 1:
        return 1
    else:
        return n + factorial(n-1)
print(factorial(int(input("Input Number for Factorial
Calculation: "))))
```



Human knowledge belongs to the world.

Function arguments

Function

최성철 교수
Director of TEAMLAB

01100
00110

Passing arguments

Passing arguments

- 함수에 입력되는 arguments는 다양한 형태를 가짐
 - 1) Keyword arguments
 - 2) Default arguments
 - 3) Variable-length arguments

Keyword arguments

- 함수에 입력되는 parameter의 변수명을 사용, arguments를 넘김

```
def print_somthing(my_name, your_name):  
    print("Hello {0}, My name is {1}" .format(your_name, my_name))  
  
print_somthing( "Sungchul" , "TEAMLAB" )  
print_somthing(your_name="TEAMLAB" , my_name="Sungchul" )
```

Default arguments

- parameter의 기본 값을 사용, 입력하지 않을 경우 기본값 출력

```
def print_somthing_2(my_name, your_name="TEAMLAB"):  
    print("Hello {0}, My name is {1}".format(your_name, my_name))  
  
print_somthing_2("Sungchul", "TEAMLAB")  
print_somthing_2("Sungchul")
```

Variable-length asterisk

함수의 parameter가 정해지지 않았다?

다항 방정식? 마트 물건 계산 함수?

가변인자 using Asterisk

가변인자 (Variable-length)

- 개수가 정해지지 않은 변수를 함수의 parameter로 사용하는 법
- Keyword arguments와 함께, argument 추가가 가능
- Asterisk(*) 기호를 사용하여 함수의 parameter를 표시함
- 입력된 값은 tuple type으로 사용할 수 있음
- 가변인자는 오직 한 개만 맨 마지막 parameter 위치에 사용가능

가변인자 (Variable-length)

- 가변인자는 일반적으로 *args를 변수명으로 사용
- 기존 parameter 이후에 나오는 값을 tuple로 저장함

```
def asterisk_test(a, b, *args):  
    return a+b+sum(args)  
  
print(asterisk_test(1, 2, 3, 4, 5))
```

가변인자 (Variable-length)

```
def asterisk_test_2(*args):
    x, y, z = args
    return x, y, z

print(asterisk_test_2(3, 4, 5))
```

키워드 가변인자 (Keyword variable-length)

- Parameter 이름을 따로 지정하지 않고 입력하는 방법
- Asterisk(*) 두개를 사용하여 함수의 parameter를 표시함
- 입력된 값은 dict type으로 사용할 수 있음
- 가변인자는 오직 한 개만 기존 가변인자 다음에 사용

키워드 가변인자 (Keyword variable-length)

```
def kwargs_test_1(**kwargs):
    print(kwargs)

def kwargs_test_2(**kwargs):
    print(kwargs)
    print("First value is {first}".format(**kwargs))
    print("Second value is {second}".format(**kwargs))
    print("Third value is {third}".format(**kwargs))
```

키워드 가변인자 (Keyword variable-length)

```
def kwargs_test_3(one, two, *args, **kwargs):
    print(one+two+sum(args))
    print(kwargs)

kwargs_test_3(3,4,5,6,7,8,9, first=3, second=4, third=5)
```



Human knowledge belongs to the world.



How to write GOOD code

Function

01100
00110



<http://goo.gl/4OxqtO>

프로그래밍 = 팀 플레이

좋은 팀을 위한 규칙

사람을 위한 코드



<http://goo.gl/u21VvR>

컴퓨터가 이해할 수 있는
코드는 어느 바보나 다 짤 수 있다.

좋은 프로그래머는 사람이
이해할 수 있는 코드를 짠다.

- 마틴 파울러 -

사람의 이해를 돋기 위해

규칙이 필요함

우리는 그 규칙을
코딩 컨벤션
이라고 함

파이썬 코딩 컨벤션

파이썬 코딩 컨벤션

- 명확한 규칙은 없음
- 때로는 팀마다, 프로젝트마다 따로
- 중요한 건 **일관성!!!**
- 읽기 좋은 코드가 좋은 코드

<http://goo.gl/OCQJ35>

파이썬 코딩 컨벤션의 예시

- 들여쓰기는 **Tab or 4 Space** 논쟁!
- 일반적으로 4 Space를 권장함
- 중요한 건 **혼합하지 않으면 됨**

<http://goo.gl/2x3G51>

PEP8 – 파이썬 코딩 컨벤션의 기준

- PEP (Python Enhance Proposal)
- 파이썬 개선을 위한 제안서
- PEP 8은 파이썬 코딩의 기준을 제시

<http://goo.gl/2x3G51>

PEP8 – 파이썬 코딩 컨벤션의 기준

- 들여쓰기 공백 4칸을 권장
- 한 줄은 최대 79자까지
- 불필요한 공백은 피함

```
def factorial( n ):  
    if n == 1:  
        return 1
```

PEP8 – 파이썬 코딩 컨벤션의 기준

- = 연산자는 1칸 이상 안 띄움

```
variable_example = 12  
variable_example = 12
```

- 주석은 항상 갱신, 불필요한 주석은 삭제

PEP8 – 파이썬 코딩 컨벤션의 기준

- 소문자 I, 대문자 O, 대문자 l 금지

IIOO = "Hard to Understand"

- 함수명은 소문자로 구성, 필요하면 밑줄로 나눔

PEP8 – 파이썬 코딩 컨벤션의 기준

- "flake8" 모듈로 체크 – flake8 <파일명>
- conda install -c anaconda flake8

```
lL0O = "123"  
for i in 10 :  
    print ("Hello")
```

```
flake8 flake8_test.py  
flake8_test.py:2:12: E203 whitespace before ':'  
flake8_test.py:3:10: E211 whitespace before '('
```

함수 개발 가이드라인

함수 작성 가이드 라인

- 함수는 가능하면 짧게 작성할 것 (줄 수를 줄일 것)
- 함수 이름에 함수의 역할, 의도가 명확히 들어낼 것

```
def print_hello_world():
    print("Hello, World")

def get_hello_world():
    return "Hello, World"
```

함수 작성 가이드 라인

- 하나의 함수에는 유사한 역할을 하는 코드만 포함

```
def add_variables(x,y):  
    return x + y
```

```
def add_variables(x,y):  
    print (x, y)  
    return x + y
```

함수 작성 가이드 라인

- 인자로 받은 값 자체를 바꾸진 말 것 (임시변수 선언)

```
def count_word(string_variable):  
    string_variable = list(string_variable)  
    return len(string_variable)
```



```
def count_word(string_variable):  
    return len(string_variable)
```

함수는 언제 만드는가?

- 공통적으로 사용되는 코드는 함수로 변환
- 복잡한 수식 → 식별 가능한 이름의 함수로 변환
- 복잡한 조건 → 식별 가능한 이름의 함수로 변환

공통 코드는 함수로

```
a = 5
if (a > 3):
    print "Hello World"
    print "Hello Gachon"
if (a > 4):
    print "Hello World"
    print "Hello Gachon"
if (a > 5):
    print "Hello World"
    print "Hello Gachon"
```

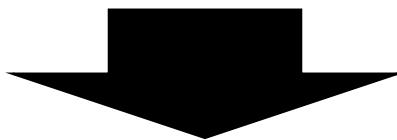


```
def print_hello():
    print "Hello World"
    print "Hello Gachon"

a = 5
if (a > 3):
    helloMethod()
if (a > 4):
    helloMethod()
if (a > 5):
    helloMethod()
```

복잡한 수식은 함수로

```
import math  
a = 1; b = -2; c = 1  
  
print((-b + math.sqrt(b ** 2 - (4 * a * c))) / (2 * a))  
print((-b - math.sqrt(b ** 2 - (4 * a * c))) / (2 * a))
```



```
import math  
  
def get_result_quadratic_equation(a, b, c):  
    values = []  
    values.append((-b + math.sqrt(b ** 2 - (4 * a * c))) / (2 * a))  
    values.append((-b - math.sqrt(b ** 2 - (4 * a * c))) / (2 * a))  
    return values  
  
print(get_result_quadratic_equation(1, -2, 1))
```



Human knowledge belongs to the world.

String Overview

String

최성철 교수
Director of TEAMLAB

01100
00110

문자열 (string)

- 시퀀스 자료형으로 문자형 data를 메모리에 저장
- 영문자 한 글자는 1byte의 메모리공간을 사용

Python shell

```
>>> import sys          # sys 모듈을 호출
>>> print (sys.getsizeof("a"), sys.getsizeof("ab"), sys.getsizeof("abc"))
50 51 52
# "a", "ab", "abc"의 각 메모리 사이즈 출력
```

문자열 (string)

- string은 1byte 크기로 한 글자씩 메모리 공간이 할당됨

```
a = "abcde"
```

a	0100 1001
b	0100 1010
c	0100 1011
d	0100 1100
e	0100 1101

1Byte의 메모리 공간???

- 컴퓨터는 2진수로 데이터를 저장
- 이진수 한 자릿수는 1bit로 저장됨
- 즉 1bit 는 0 또는 1
- 1 byte = 8 bit = 2^8 = 256 까지 저장 가능

Source: <http://goo.gl/xr4Az2>

1Byte의 메모리 공간???

- 컴퓨터는 문자를 직접적으로 인식 X
→ 모든 데이터는 2진수로 인식
- 이를 위해 2진수를 문자로 변환하는 표준 규칙을 정함
- 이러한 규칙에 따라 문자를 2진수로 변환하여 저장하거나
저장된 2진수를 숫자로 변환하여 표시함
- 예) 대문자 U는 이진수로 “1000011” 변환됨 (UTF-8기준)

1Byte의 메모리 공간???

000	(nul)	016 ► (dle)	032 sp	048 ߱	064 ߲	080 P	096 ߳	112 p
001 ☺ (soh)	017 ◀ (dc1)	033 !	049 ߱	065 A	081 Q	097 a	113 q	
002 ☻ (stx)	018 ‡ (dc2)	034 " "	050 ߱	066 B	082 R	098 b	114 r	
003 ♥ (etx)	019 !! (dc3)	035 #	051 ߱	067 C	083 S	099 c	115 s	
004 ♦ (eot)	020 ¶ (dc4)	036 \$	052 ߱	068 D	084 T	100 d	116 t	
005 ♣ (enq)	021 \$ (nak)	037 %	053 ߱	069 E	085 U	101 e	117 u	
006 ♤ (ack)	022 – (syn)	038 &	054 ߱	070 F	086 V	102 f	118 v	
007 • (bel)	023 † (etb)	039 *	055 ߱	071 G	087 W	103 g	119 w	
008 ☐ (bs)	024 † (can)	040 (056 ߱	072 H	088 X	104 h	120 x	
009 (tab)	025 ↓ (em)	041)	057 ߱	073 I	089 Y	105 i	121 y	
010 (lf)	026 (eof)	042 *	058 :	074 J	090 Z	106 j	122 z	
011 ☏ (vt)	027 ← (esc)	043 +	059 ;	075 K	091 [107 k	123 {	
012 ☓ (np)	028 ↂ (fs)	044 ,	060 <	076 L	092 \	108 l	124	
013 (cr)	029 ↔ (gs)	045 -	061 =	077 M	093]	109 m	125 }	
014 ☔ (so)	030 ▲ (rs)	046 .	062 >	078 N	094 ^	110 n	126 ~	
015 ☕ (si)	031 ▼ (us)	047 /	063 ?	079 O	095 _	111 o	127 △	

프로그램 언어에서 데이터 타입

- 각 타입 별로 메모리 공간을 할당 받은 크기가 다름

종류	타입	크기	표현 범위 (32bit)
정수형	int	4바이트	$-2^{31} \sim 2^{31}-1$
	long	무제한	무제한
실수형	float	8바이트	약 $10^{-308} \sim 10^{+308}$

<http://goo.gl/JSI4ZP>

- 메모리 공간에 따라 표현할 수 있는 숫자범위가 다름

예) 4byte = 32bit = $2^{32} = 4,294M = -2,147M \sim + 2.147M$ 까지 표시

- 데이터 타입은 메모리의 효율적 활용을 위해 매우 중요

문자열의 특징

인덱싱 (Indexing)

- 문자열의 각 문자는 개별 주소(offset)를 가짐
- 이 주소를 사용해 할당된 값을 가져오는 것이 인덱싱
- List와 같은 형태로 데이터를 처리함

```
>>> a = "abcde"
>>> print (a[0], a[4])          # a 변수의 0번째, 4번째 주소에 있는 값
a e
>>> print (a[-1], a[-5])       # a 변수의 오른쪽에서 0번째, 4번째 주소에 있는 값
e a
```

Python shell

Source: <http://goo.gl/XjPSpP>

인덱싱 (Indexing)

Hello

0 1 2 3 4
-5 -4 -3 -2 -1

각 문자의 오프셋은

왼쪽에선 0부터 오른쪽에선 -1부터 시작함

```
>>> a = "abcde"
>>> print (a[0], a[4])          # a 변수의 0번째, 4번째 주소에 있는 값
a e
>>> print (a[-1], a[-5])       # a 변수의 오른쪽에서 0번째, 4번째 주소에 있는 값
e a
```

Python shell

Source: <http://goo.gl/XjPSP>

슬라이싱 (Slicing)

- 문자열의 주소값을 기반으로 문자열의 부분값을 반환

Python shell

```
>>> print (a[0:6], " AND ", a[-9:]) # a 변수의 0부터 5까지, -9부터 끝까지
```

Gachon AND University

```
>>> print (a[:]) # a변수의 처음부터 끝까지
```

Gachon University

```
>>> print (a[-50:50]) # 범위를 넘어갈 경우 자동으로 최대 범위를 지정
```

Gachon University

```
>>> print (a[::-2], " AND ", a[::-1]) # 2칸 단위로, 역으로 슬라이싱
```

Gco nest AND ytisrevnU nohcaG

문자열 연산 및 포함여부 검사

- 덧셈과 뺄셈 연산 가능, `in` 명령으로 포함여부 체크

```
>>> a = "TEAM"
>>> b = "LAB"
>>> print (a + " " + b)      # 덧셈으로 a와 b 변수 연결하기
```

TEAM LAB

```
>>> print (a * 2 + " " + b * 2)
TEAMTEAM LABLAB      # 곱하기로 반복 연산 가능
>>> if 'A' in a: # 'U'가 a에 포함되었는지 확인
```

```
...   print (a)
```

```
... else:
```

```
...   print (b)
```

```
...
```

TEAM

Python shell

문자열 함수 (1/2)

함수명	기능
len(a)	문자열의 문자 개수를 반환
a.upper()	대문자로 변환
a.lower()	소문자로 변환
a.capitalize()	첫 문자를 대문자로 변환
a.title()	제목형태로 변환 띄워쓰기 후 첫 글자만 대문자
a.count('abc')	문자열 a에 'abc'가 들어간 횟수 반환
a.find('abc') a.rfind('abc')	문자열 a에 'abc'가 들어간 위치(오프셋) 반환
a.startswith('ab'c)	문자열 a는 'abc'로 시작하는 문자열여부 반환
a.endswith('abc')	문자열 a는 'abc'로 끝나는 문자열여부 반환

문자열 함수 (2/2)

함수명	기능
a.strip()	좌우 공백을 없앰
a.rstrip()	오른쪽 공백을 없앰
a.lstrip()	왼쪽 공백을 없앰
a.split()	공백을 기준으로 나눠 리스트로 반환
a.split('abc')	abc를 기준으로 나눠 리스트로 반환
a.isdigit()	문자열이 숫자인지 여부 반환
a.islower()	문자열이 소문자인지 여부 반환
a.isupper()	문자열이 대문자인지 여부 반환

[예제] 문자열 함수

```
>>> title = "TEAMLAB X Inflearn"
>>> title.upper()
'TEAMLAB X INFLEARN'
>>> title.lower()
'teamlab x inflearn'
>>> title.split(" ")
['TEAMLAB', 'X', 'Inflearn']
>>> title.isdigit()
False
>>> title.title()
'Teamlab X Inflearn'
>>> title.startswith("a")
False
>>> title.count("a")
1
>>> title.upper().count("a")
0
```

```
>>> "12345".isdigit()
True
>>> title.find("Gachon")
0
>>> title.upper().find("Gachon")
-1
>>> "Hello ".strip()
'Hello'
>>> "A-B-C-D-E-F".split("-")
['A', 'B', 'C', 'D', 'E', 'F']
```

Python shell

다양한 문자열 표현

문자열 선언은 큰따옴표(“”)나 작은 따옴표 (‘’)를 활용

It's OK 이라는 문자열은 어떻게 표현할까?

① a = 'It'₩' ok.'

₩는 문자열 구분자가 아닌 출력 문자로 처리

② a = "It's ok." #

큰따옴표로 문자열 선언 후 작은 따옴표는 출력 문자로 사용

다양한 문자열 표현

두줄 이상은 어떻게 저장할까?

- ① **₩n #₩n**은 줄바꿈을 의미하는 특수 문자
- ② 큰따옴표 또는 작은 따옴표 세 번 연속 사용

예) a= "It's Ok"

I'm Happy.

See you. " "

특수 문자

문자열을 표시할 때 백슬래시 “₩”를 사용하여
키보드로 표시하기 어려운 문자들을 표현함

문자	설명	문자	설명
₩ [Enter]	다음 줄과 연속임을 표현	₩n	줄 바꾸기
<u>₩₩</u>	₩ 문자 자체	₩t	TAB 키
₩`	` 문자	₩e	ESC 키
₩"	" 문자		
₩b	백 스페이스		



Human knowledge belongs to the world.

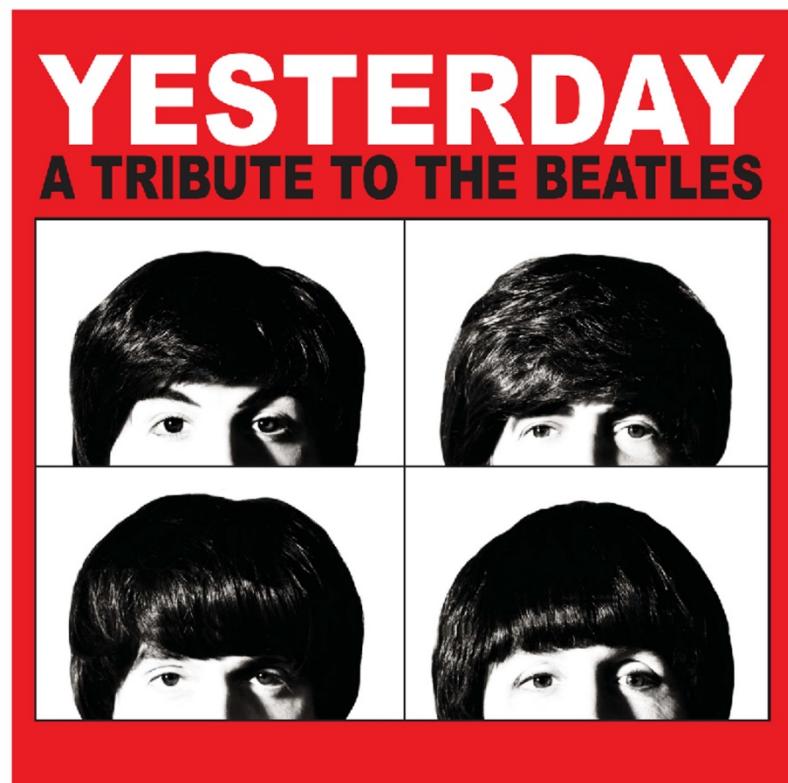
Lab: Yesterday Counter

String

최성철 교수
Director of TEAMLAB

01100
00110

Yesterday



Yesterday 노래엔
Yesterday 라는 말이 몇 번 나올까?

<http://goo.gl/UqNp1r>

Lab: Yesterday Counter

“Yesterday” 노래에 “Yesterday” 단어의 개수?

```
wget https://raw.githubusercontent.com/TeamLab/cs50_example_code/master/12_string/yesterday.txt
```

CMD Window

```
f = open("yesterday.txt", 'r')
yesterday_lyric = ""

while 1:
    line = f.readline()
    if not line:
        break
    yesterday_lyric = yesterday_lyric + line.strip() + "\n"
f.close()

n_of_yesterday = yesterday_lyric.upper().count("YESTERDAY")
print ("Number of a Word 'Yesterday' ", n_of_yesterday)
```

Python shell

yesterday_counter.py

Lab: Yesterday Counter II

대소문자를 구분하여 “Yesterday”와 “yesterday”의
개수를 나눠서 세는 프로그램을 작성하세요.



Human knowledge belongs to the world.

Data Structure Overview

Data Structure

01100
00110



**특징이 있는 정보는
어떻게 저장하면 좋을까?**

생각해보기

- 전화번호부 정보는 어떻게 저장하면 좋을까?
- 은행 번호표 정보는 어떻게 처리하면 좋을까?
- 서적 정보는 어떻게 관리하면 좋을까?
- 창고에 쌓인 수화물의 위치를 역순으로 찾을 때?

데이터 구조(Data Structure)

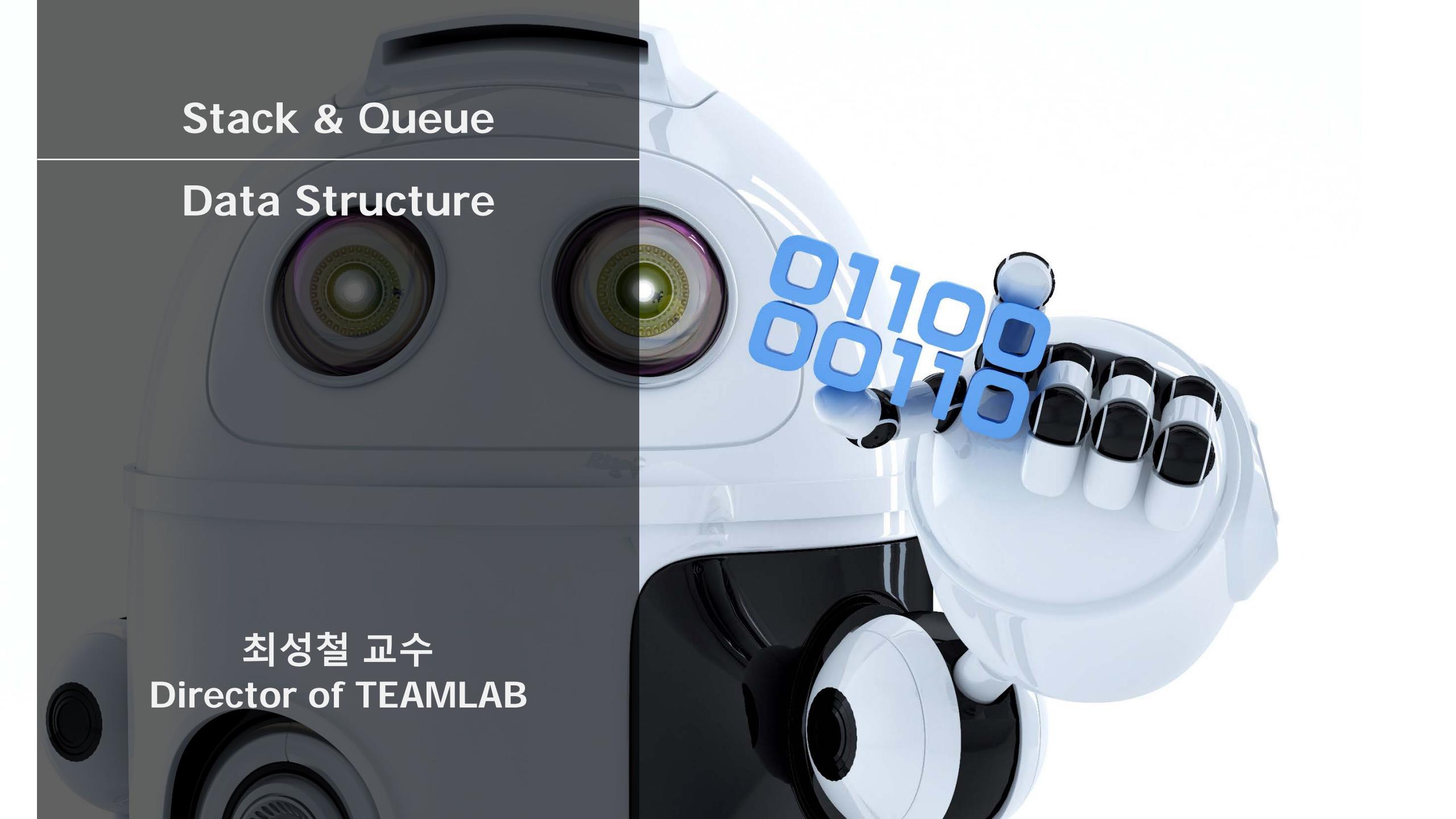
- 메모리상에 데이터를 효율적으로 관리하는 방법
- 검색, 저장 등의 작업에서 효율을 고려하여 메모리 사용량과 실행시간 등을 최소화 함
- 파이썬에서는 리스트, 튜플, 집합(Set), 사전(dictionary) 등의 기본 데이터 구조를 제공함

데이터 구조(Data Structure)

- 스택과 큐(stack & queue)
- 튜플과 집합(tuple & set)
- 사전(dictionary)

END

감사합니다.



Stack & Queue

Data Structure

최성철 교수

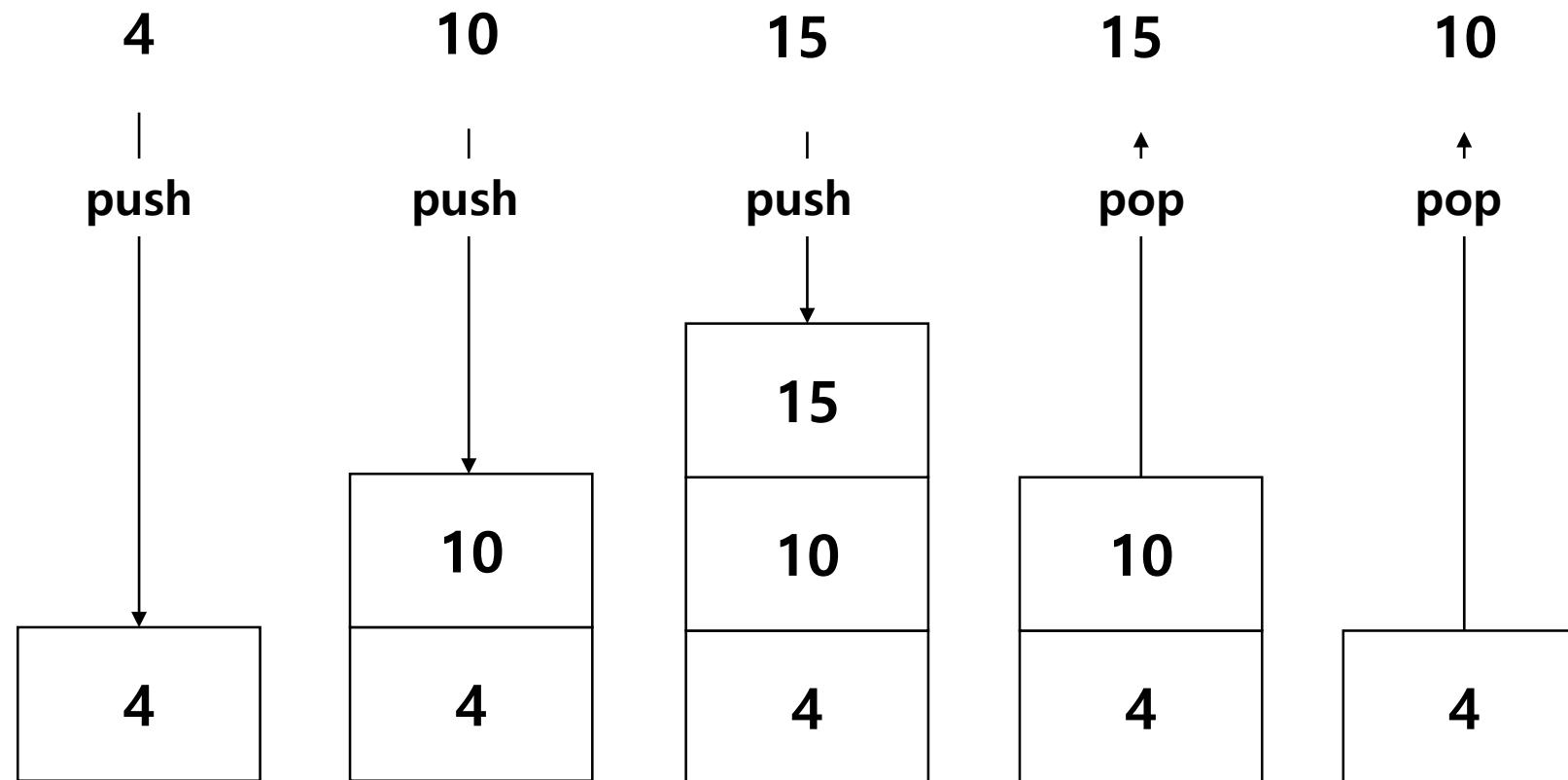
Director of TEAMLAB

Stack

스택 (Stack)

- 나중에 넣은 데이터를 먼저 반환하도록 설계된 메모리 구조로
Last In First Out (LIFO)로 구현됨
- Data의 입력을 **Push**, 출력을 **Pop**이라고 함

스택 (Stack)



스택 (Stack) in 파이썬

- 파이썬은 리스트를 사용하여 스택 구조를 활용
- push를 append(), pop을 pop()를 사용

```
>>> a = [1,2,3,4,5]
>>> a.append(10)
>>> a.append(20)
>>> a.pop()          # 20 출력
20
>>> a.pop()          # 10 출력
10
>>>
```

Python shell

스택 (Stack) Example

스택 구조를 활용, 입력된 글자를 역순으로 출력

Editor

```
word = input("Input a word : ")          # Input Word
word_list = list(word)                  # String to List
for i in range(len(word_list)):
    print (word_list.pop())            # 하나씩 빼면서 출력
```

Queue

큐 (Queue)

- 먼저 넣은 데이터를 먼저 반환하도록 설계된 메모리 구조로
First In First Out으로 구현됨
- Stack과 반대되는 개념

	PUT(A)	PUT(B)	PUT(C)	GET()	PUT(D)	GET()
					D	D
			C	C	C	C
		B	B	B	B	
A	A	A				

Source: <http://goo.gl/2Si0RS>

큐 (Queue) in 파이썬

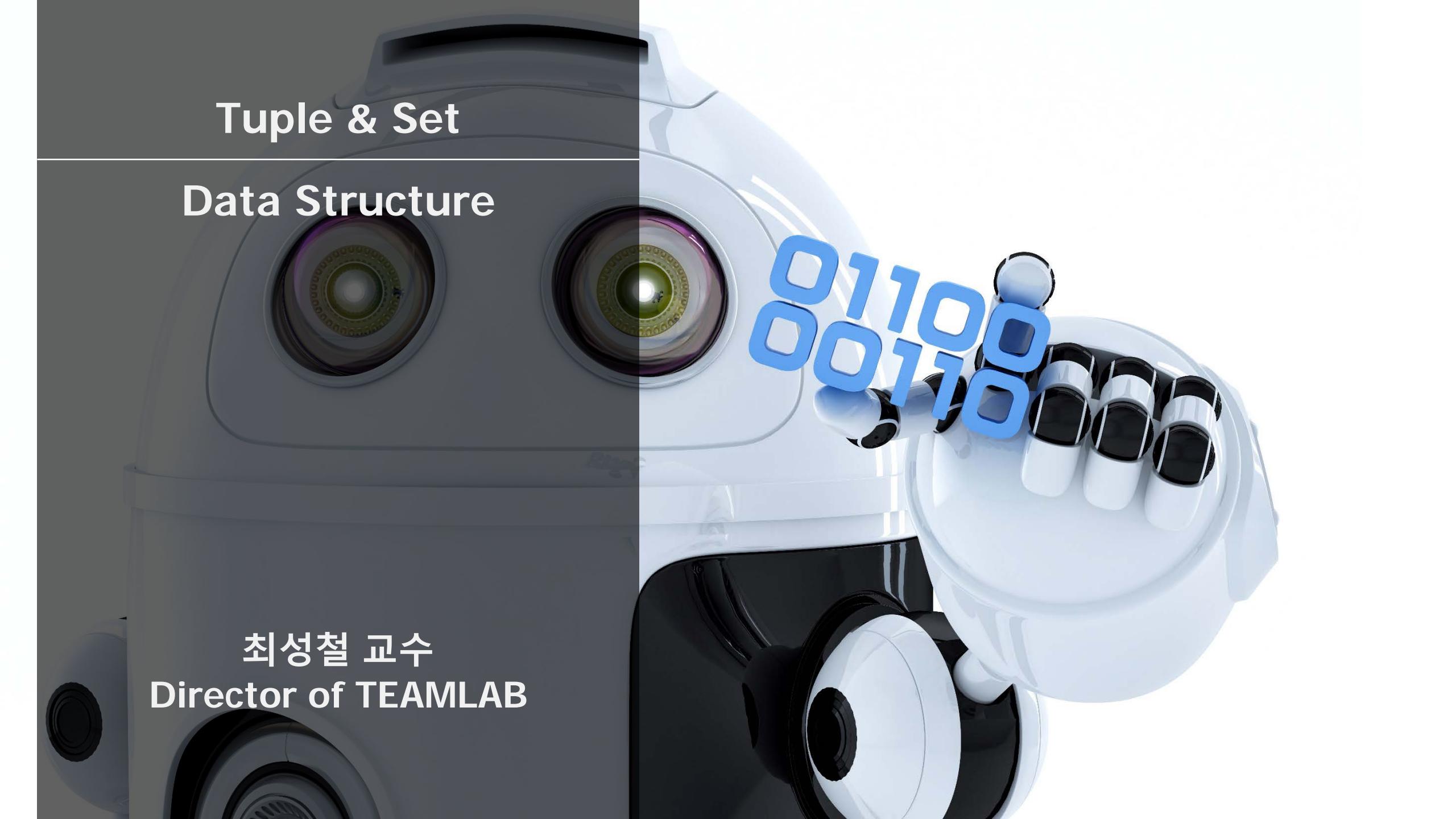
- 파이썬은 리스트를 사용하여 큐 구조를 활용
- put를 append(), get을 pop(0)를 사용

```
>>> a = [1,2,3,4,5]
>>> a.append(10)
>>> a.append(20)
>>> a.pop(0) # 1 출력
1
>>> a.pop(0) # 2 출력
2
>>>
```

Python shell



Human knowledge belongs to the world.



Tuple & Set

Data Structure

최성철 교수

Director of TEAMLAB

Tuple

튜플 (tuple)

- 값의 변경이 불가능한 리스트
- 선언 시 “[]” 가 아닌 “()”를 사용
- 리스트의 연산, 인덱싱, 슬라이싱 등을 동일하게 사용

```
>>> t = (1,2,3)  
>>> print (t + t , t * 2) # (1, 2, 3, 1, 2, 3) (1, 2, 3, 1, 2, 3)  
(1, 2, 3, 1, 2, 3) (1, 2, 3, 1, 2, 3)
```

```
>>> len(t) # 3
```

```
3
```

```
>>> t[1] = 5 # Error 발생
```

Traceback (most recent call last):

 File "<stdin>", line 1, in <module>

TypeError: 'tuple' object does not support item assignment

```
>>>
```

Python Shell

튜플 (tuple)

Python Shell

```
>>> t = (1)      # 일반정수로 인식  
1  
>>> t = (1, )  # 값이 하나인 Tuple은 반드시 "," 를 붙여야 함  
(1,)
```

왜 쓸까?

프로그램을 작동하는 동안 변경되지 않은 데이터의 저장
Ex) 학번, 이름, 우편번호 등등

Set

집합 (set)

- 값을 순서없이 저장, 중복 불허 하는 자료형
- set 객체 선언을 이용하여 객체 생성

```
>>> s = set([1,2,3,1,2,3])      # set 함수를 사용 1,2,3을 집합 객체 생성
>>> s
{1, 2, 3}
>>> s.add(1)                  # 한 원소 1만 추가, 추가, 중복불허로 추가 되지 않음
>>> s
{1, 2, 3}
>>> s.remove(1)               # 1 삭제
>>> s
{2, 3}
>>> s.update([1,4,5,6,7])     # [1,4,5,6,7] 추가
>>> s
{1, 2, 3, 4, 5, 6, 7}
>>> s.discard(3)              # 3 삭제
>>> s
{1, 2, 4, 5, 6, 7}
>>> s.clear()                 # 모든 원소 삭제
```

Python Shell

집합의 연산

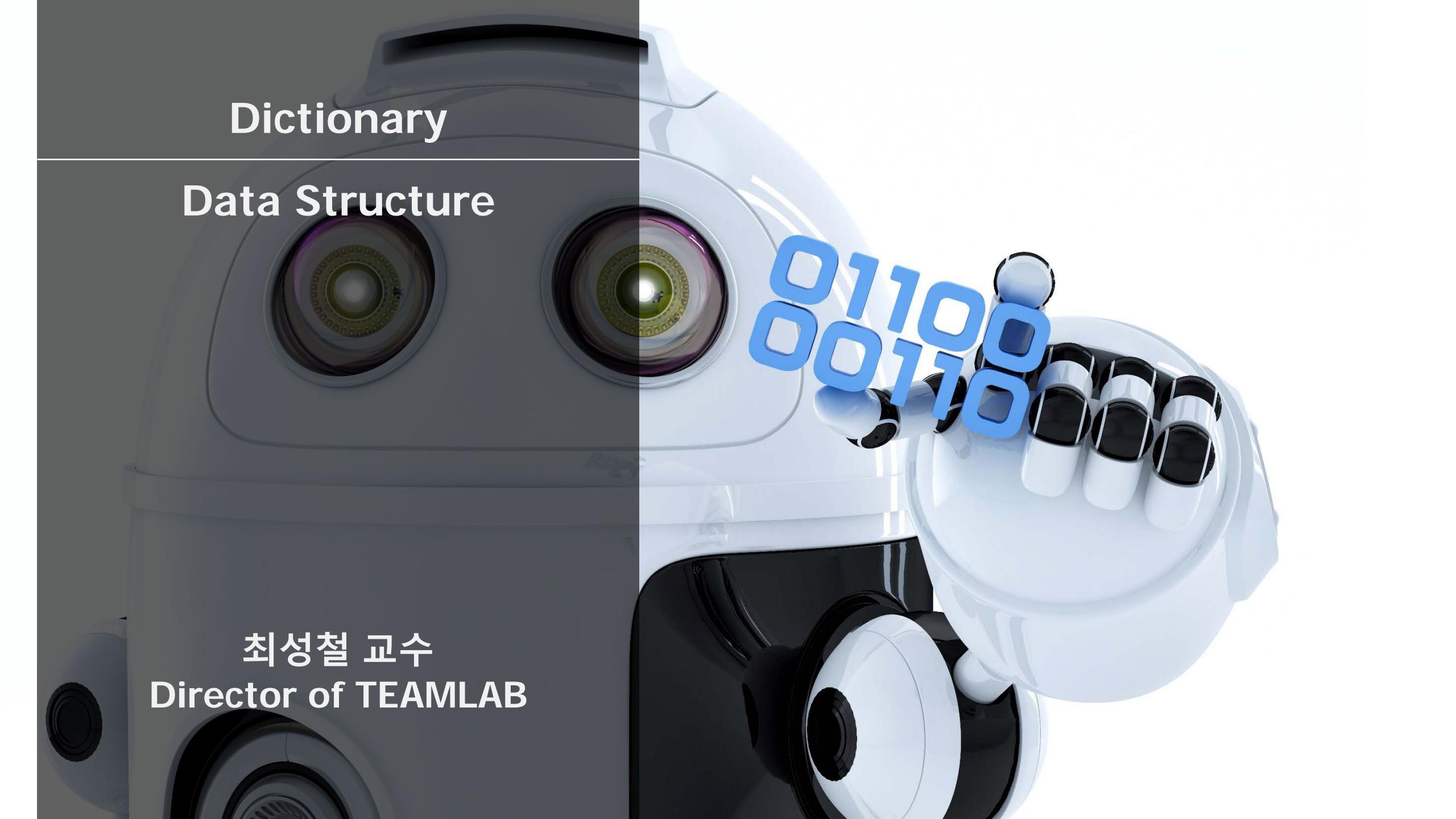
- 수학에서 활용하는 다양한 집합연산 가능

```
>>> s1 = set([1,2,3,4,5])
>>> s2 = set([3,4,5,6,7])
>>> s1.union(s2)           # s1 과 s2의 합집합
{1, 2, 3, 4, 5, 6, 7}
>>> s1 | s2               # set([1, 2, 3, 4, 5, 6, 7])
{1, 2, 3, 4, 5, 6, 7}
>>> s1.intersection(s2)  # s1 과 s2의 교집합
{3, 4, 5}
>>> s1 & s2               # set([3, 4, 5])
{3, 4, 5}
>>> s1.difference(s2)    # s1 과 s2의 차집합
{1, 2}
>>> s1 - s2               # set([1, 2])
{1, 2}
```

Python Shell



Human knowledge belongs to the world.



Dictionary

Data Structure

최성철 교수

Director of TEAMLAB

사전 (dictionary)

- 데이터를 저장 할 때는 구분 지을 수 있는 값을 함께 저장

- 구분을 위한 데이터 고유 값을 Identifier 또는 Key 라고 함
 - Key 값을 활용하여, 데이터 값(Value)를 관리함

사전 (dictionary)

학번	이름	생년월일	주소
20150230	홍길동	1995-04-03	서울시 동대문구
20150233	김가천	1995-04-20	성남시 분당구
20150234	오영심	1996-01-03	성남시 중원구
20150236	최성철	1995-12-27	인천시 계양구

사전 (dictionary)

- key와 value를 매칭하여 key로 value를 검색
- 다른 언어에서는 Hash Table 이라는 용어를 사용
- {Key1:Value1, Key2:Value2, Key3:Value3 ...} 형태

```
student_info = {20140012:'Sungchul',  
20140059:'Jiyong',20140058:'JaeHong'}
```

```
student_info[20140012]  
student_info[20140012] = 'Janhyeok'  
student_info[20140012]  
student_info[20140039] = 'wonchul'  
student_info
```

Key	Value
20140012	Janhyeok
20140059	Jiyong
20140058	JaeHong
20140039	Wonchul

사전 (dictionary) 다루기 1

```
>>> country_code = {} # Dict 생성
>>> country_code = {"America": 1, "Korea": 82, "China": 86, "Japan": 81}
>>> country_code
{'America': 1, 'China': 86, 'Korea': 82, 'Japan': 81}
>>> country_code.items() # Dict 데이터 출력
dict_items([('America', 1), ('China', 86), ('Korea', 82), ('Japan', 81)])
>>> country_code.keys() # Dict 키 값만 출력
dict_keys(['America', 'China', 'Korea', 'Japan'])
>>> country_code["Gernman"] = 49 # Dict 추가
>>> country_code
{'America': 1, 'Gernman': 49, 'China': 86, 'Korea': 82, 'Japan': 81}
>>> country_code.values() # Dict Value만 출력
dict_values([1, 49, 86, 82, 81])
```

Python Shell

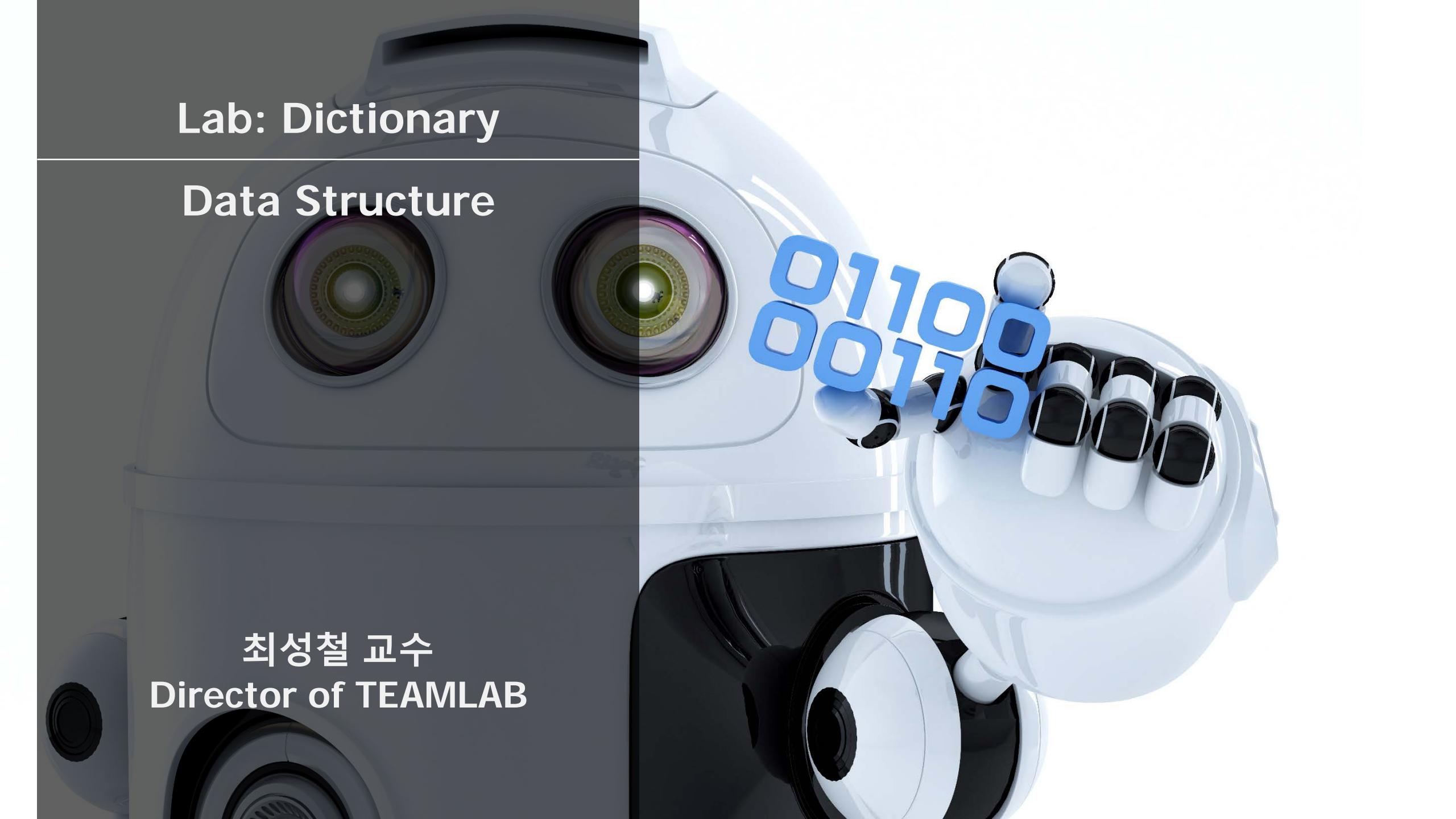
사전 (dictionary) 다루기 2

```
>>> for k,v in country_code.items():
...   print ("Key : ", k)
...   print ("Value : ", v)
...
Key : America
Value : 1
Key : Gernman
Value : 49
Key : China
Value : 86
Key : Korea
Value : 82
Key : Japan
Value : 81
>>> "Korea" in country_code.keys()      # Key값에 "Korea"가 있는지 확인
True
>>> 82 in country_code.values()    # Value값에 82가 있는지 확인
True
```

Python Shell



Human knowledge belongs to the world.



Lab: Dictionary

Data Structure

최성철 교수

Director of TEAMLAB

Command Analyzer

- Command: 사용자가 서버에 명령어를 입력한 명령어

A	B
command	id
_raw	
vi gowithflow.py	source
python3.4 submit_assignment.py -submit gowithflow.py	tiana
vi gowithflow.py	tiana
python3.4 gowithflow.py	white_rabbit
vi gowithflow.py	white_rabbit
python3.4 gowithflow.py	white_rabbit
vi gowithflow.py	tiana
python3.4 submit_assignment.py -submit gowithflow.py	tiana
vi gowithflow.py	white_rabbit
python3.4 gowithflow.py	white_rabbit
python3.4 submit_assignment.py -submit gowithflow.py	tiana
vi gowithflow.py	flo

어떤 사용자가
얼마나 많이
명령어를 입력하였을 까?

Data source:
<https://goo.gl/VpbzJ8>

Code

command_analyzer.py

Editor

```
import csv

def getKey(item):          # 정렬을 위한 함수
    return item[1]          # 신경 쓸 필요 없음

command_data = []           # 파일 읽어오기
with open("command_data.csv", "r") as csvfile:
    spamreader = csv.reader(csvfile, delimiter=',', quotechar='"')
    for row in spamreader:
        command_data.append(row)

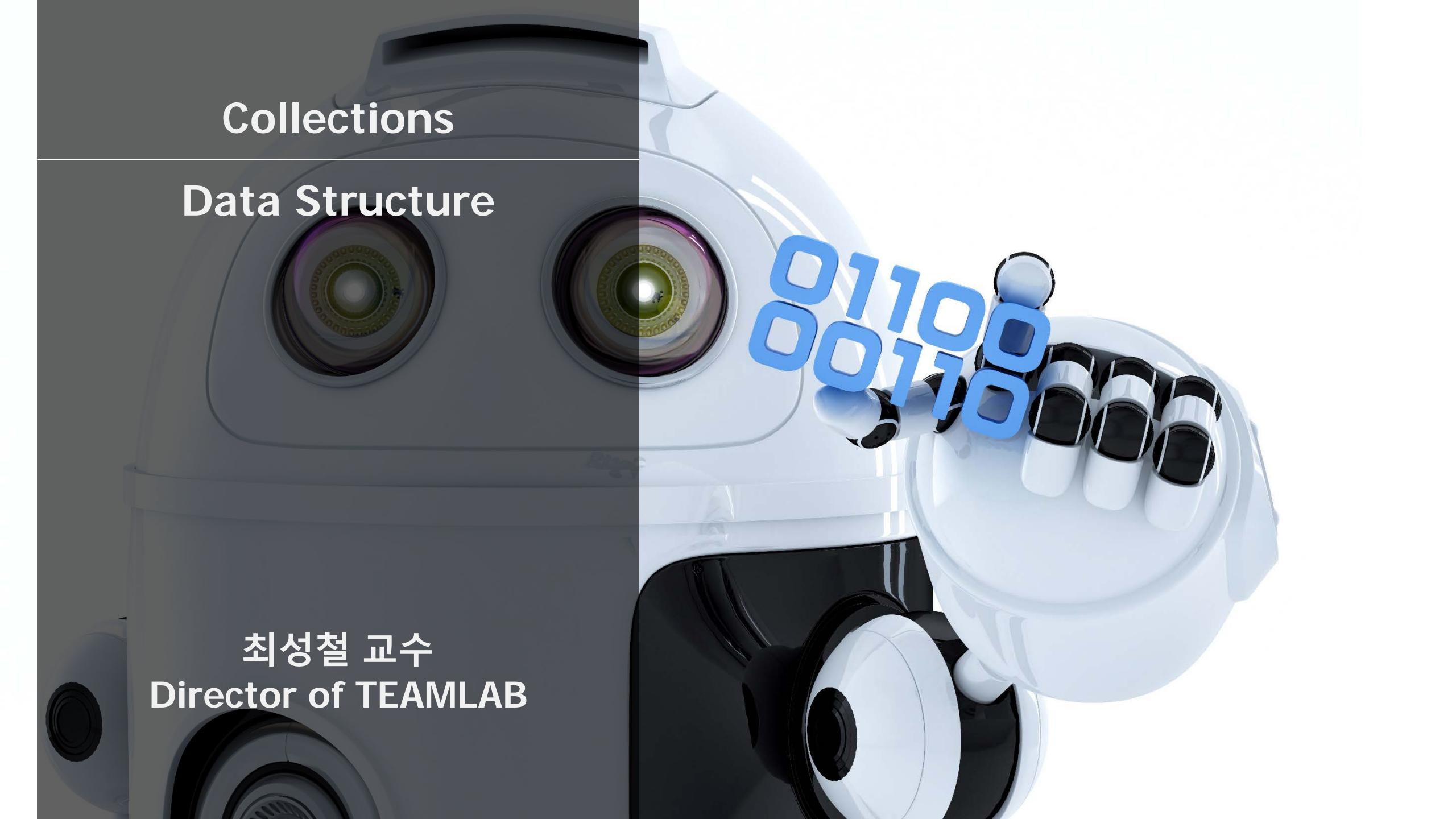
command_counter = {}         # dict 생성, 아이디를 key값, 입력줄수를 value값
for data in command_data:
    if data[1] in command_counter.keys():      # list 데이터를 dict로 변경
        command_counter[data[1]] += 1            # 아이디가 이미 Key값으로 변경되었을 때
    else:                                       # 기존 출현한 아이디
        command_counter[data[1]] = 1             # 처음 나온 아이디

dictlist = []                  # dict를 list로 변경
for key, value in command_counter.items():
    temp = [key,value]
    dictlist.append(temp)

sorted_dict= sorted(dictlist, key=getKey, reverse=True) # list를 입력 줄 수로 정렬
print (sorted_dict[:100])                                # List의 상위 10개값만 출력
```



Human knowledge belongs to the world.



Collections

Data Structure

최성철 교수

Director of TEAMLAB

Collections

Collections

- List, Tuple, Dict에 대한 Python Built-in 확장 자료 구조(모듈)
- 편의성, 실행 효율 등을 사용자에게 제공함
- 아래의 모듈이 존재함

```
from collections import deque
from collections import Counter
from collections import OrderedDict
from collections import defaultdict
from collections import namedtuple
```

deque

deque

- Stack과 Queue 를 지원하는 모듈
- List에 비해 효율적인 자료 저장 방식을 지원함

```
from collections import deque
```

```
deque_list = deque()  
for i in range(5):  
    deque_list.append(i)  
print(deque_list)
```

```
deque_list.appendleft(10)  
print(deque_list)
```

deque

- **rotate, reverse**등 Linked List의 특성을 지원함
- 기존 list 형태의 함수를 모두 지원함

```
deque_list.rotate(2)
```

```
print(deque_list)
```

```
deque_list.rotate(2)
```

```
print(deque_list)
```

```
print(deque_list)
```

```
print(deque(reversed(deque_list)))
```

```
deque_list.extend([5, 6, 7])
```

```
print(deque_list)
```

```
deque_list.extendleft([5, 6, 7])
```

```
print(deque_list)
```

deque

- deque 는 기존 list보다 효율적인 자료구조를 제공
- 효율적 메모리 구조로 처리 속도 향상

deque

```
from collections import deque
import time

start_time = time.clock()
deque_list = deque()
# Stack
for i in range(10000):
    for i in range(10000):
        deque_list.append(i)
        deque_list.pop()
print(time.clock() - start_time, "seconds")
```

general list

```
import time

start_time = time.clock()
just_list = []
for i in range(10000):
    for i in range(10000):
        just_list.append(i)
        just_list.pop()
print(time.clock() - start_time, "seconds")
```

OrderedDict

OrderedDict

- Dict와 달리, 데이터를 입력한 순서대로 dict를 반환함

```
from collections import OrderedDict
```

```
d = {}  
d['x'] = 100  
d['y'] = 200  
d['z'] = 300  
d['l'] = 500
```

l	500	x
x	100	
y	200	
z	300	

```
for k, v in d.items():  
    print(k, v)
```

```
d = OrderedDict()  
d['x'] = 100  
d['y'] = 200  
d['z'] = 300  
d['l'] = 500
```

x	100	x
y	200	
z	300	
l	500	

```
for k, v in d.items():  
    print(k, v)
```

OrderedDict

- Dict type의 값을, value 또는 key 값으로 정렬할 때 사용 가능

```
for k, v in OrderedDict(sorted(d.items(), key=lambda t: t[0])).items():
    print(k, v)
```

l	500	x
x	100	
y	200	
z	300	

```
for k, v in OrderedDict(sorted(d.items(), key=lambda t: t[1])).items():
    print(k, v)
```

x	100	x
y	200	
z	300	
l	500	

defaultdict

defaultdict

- Dict type의 값에 기본 값을 지정, 신규값 생성시 사용하는 방법

```
d = dict()  
print(d["first"])
```

```
-----  
KeyError
```

```
<ipython-input-1-3b4ee8f5b4c3> in <module>()
```

```
    1 d = dict()  
----> 2 print(d["first"])
```

```
Traceback (most recent call l  
x
```

```
KeyError: 'first'
```

defaultdict

- Dict type의 값에 기본 값을 지정, 신규값 생성시 사용하는 방법

```
from collections import defaultdict  
  
d = defaultdict(object)    # Default dictionary를 생성  
  
d = defaultdict(lambda: 0) # Default 값을 0으로 설정합  
  
print(d["first"])
```

defaultdict

- 하나의 지문에 각 단어들이 몇 개나 있는지 세고 싶을 경우?
- **Text-mining 접근법 - Vector Space Model**

text = """A press release is the quickest and easiest way to get free publicity. If well written, a press release can result in multiple published articles about your firm and its products. And that can mean new prospects contacting you asking you to sell to them.""".lower().split()

```
print(text)
```

```
['a', 'press', 'release', 'is', 'the', 'quickest', 'and', 'easiest', 'way', 'to', 'get', 't']  
x
```

defaultdict

```
from collections import OrderedDict

word_count = defaultdict(object) # Default dictionary를 생성
word_count = defaultdict(lambda: 0) # Default 값을 0으로 설정합

for word in text:
    word_count[word] += 1

for i, v in OrderedDict(sorted(
    word_count.items(), key=lambda t: t[1],
    reverse=True)).items():
    print(i, v)
```

a 12
to 10
and 9
the 9
press 8
release 8
that 7
of 5
your 4

Counter

Counter

- Sequence type의 data element들의 갯수를 dict 형태로 반환

```
from collections import Counter

c = Counter()                      # a new, empty counter
c = Counter('gallahad')            # a new counter from an iterable
print(c)  Counter({'a': 3, 'l': 2, 'g': 1, 'd': 1, 'h': 1})
```

Counter

- Dict type, keyword parameter 등도 모두 처리 가능

```
c = Counter({'red': 4, 'blue': 2})      # a new counter from a mapping  
print(c)  
print(list(c.elements()))
```

```
Counter({'red': 4, 'blue': 2})  
['blue', 'blue', 'red', 'red', 'red', 'red']
```

```
c = Counter(cats=4, dogs=8)           # a new counter from keyword args
print(c)
print(list(c.elements()))
```

Counter

- Set의 연산들을 지원함

```
c = Counter(a=4, b=2, c=0, d=-2)
d = Counter(a=1, b=2, c=3, d=4)
c.subtract(d) # c - d
print(c) Counter({'a': 3, 'b': 0, 'c': -3, 'd': -6})
```

Counter

- Set의 연산들을 지원함

```
c = Counter(a=4, b=2, c=0, d=-2)
d = Counter(a=1, b=2, c=3, d=4)
print(c + d)
print(c & d)
print(c | d)
```

Counter({'a': 5, 'b': 4, 'c': 3, 'd': 2})	x
Counter({'b': 2, 'a': 1})	
Counter({'a': 4, 'd': 4, 'c': 3, 'b': 2})	

Counter

- word counter의 기능도 손쉽게 제공함

text = """A press release is the quickest and easiest way to get free publicity. If well written, a press release can result in multiple published articles about your firm and its products. And that can mean new prospects contacting you asking you to sell to them.""".lower().split()

```
print(Counter(text))
print(Counter(text)[ "a" ])
```

```
Counter({'a': 12, 'to': 10, 'the': 9, 'and': 9, 'release': 8, 'press': 8, 'that': 7, 'of': 5,
12
```

namedtuple

namedtuple

- Tuple 형태로 Data 구조체를 저장하는 방법
- 저장되는 data의 variable을 사전에 지정해서 저장함

```
from collections import namedtuple          x, y = p
Point = namedtuple('Point', ['x', 'y'])      print(x, y)
p = Point(11, y=22)                        print(p.x + p.y)
print(p[0] + p[1])                          print(Point(x=11, y=22))
```

```
from collections import namedtuple
import csv
f = open("./code/7/collections/users.csv", "r")
next(f)
reader = csv.reader(f)
student_list = []
for row in reader:
    student_list.append(row)
    print(row)
```

```
columns = ["user_id", "integration_id", "login_id", "password", "first_name",
           "last_name", "full_name", "sortable_name", "short_name",
           "email", "status"]
Student = namedtuple('Student', " ".join(columns))
student_namedtuple_list = []
for row in student_list:
    student = Student(*row)
    student_namedtuple_list.append(student)
print(student_namedtuple_list)
print(student_namedtuple_list[0].full_name)
```



Human knowledge belongs to the world.

The background of the slide features a white humanoid robot with a smooth, rounded body. It has two large, circular eyes with yellow and green pupils, giving it a friendly appearance. A hand is visible on the right side, holding a blue digital tablet. The screen of the tablet displays the binary code "01100110" in a large, bold, blue font.

Overview

Pythonic Code

최성철 교수

Director of TEAMLAB

파이썬 스타일 코드 (Pythonic Code)

Pythonic Code

예시: 여러 단어들을 하나로 붙일 때

Python shell

```
>>> colors = ['red', 'blue', 'green', 'yellow']
>>> result = ''
>>> for s in colors:
    result += s
```

Pythonic Code

예시: 여러 단어들을 하나로 붙일 때

Python shell

```
>>> colors = ['red', 'blue', 'green', 'yellow']
>>> result = ''.join(colors)
>>> result
'redbluegreenyellow'
```

Good Case

Source: <http://python.net/~goodger/projects/pycon/2007/idiomatic/handout.html>

Pythonic Code

- 파이썬 스타일의 코딩 기법
- 파이썬 특유의 문법을 활용하여 효율적으로 코드를 표현함
- 고급 코드를 작성 할 수록 더 많이 필요해짐

Pythonic Code

- Split & Join
- List Comprehension
- Enumerate & Zip

Why Pythonic Code?

남 코드에 대한 이해도

많은 개발자들이 python 스타일로 코딩한다

효율

단순 for loop append보다 list가 조금 더 빠르다

익숙해지면 코드도 짧아진다.

간지

쓰면 왠지 코드 잘 짜는 거처럼 보인다

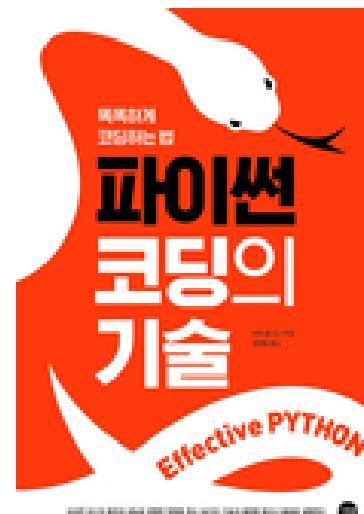
Supplement Materials



Fluent Python – 전문가를 위한 파이썬

(한빛미디어, 2016)

Part 2 부분 – 시퀀스 & 딕션너리



Effective Python - 파이썬 코딩의 기술

(길벗, 2016)

Supplement Materials



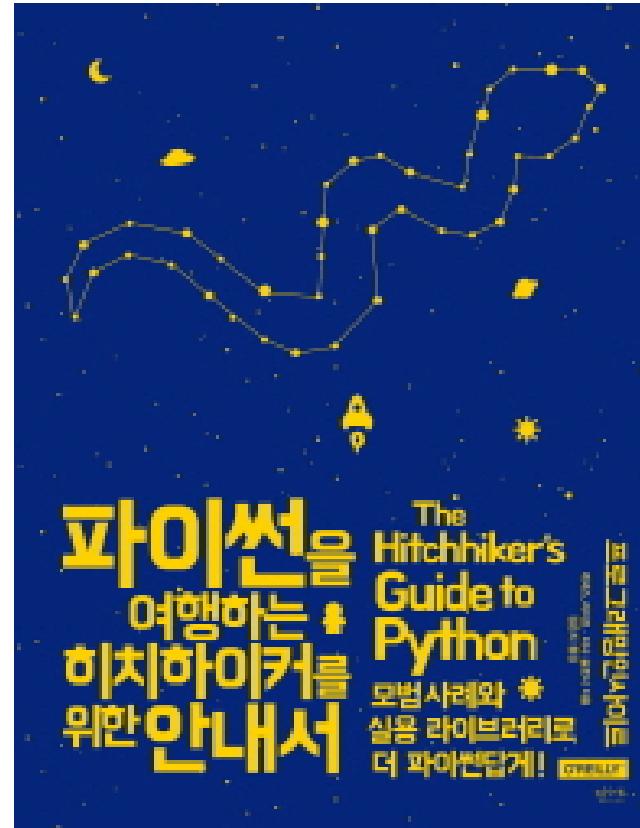
10 Tips for Pythonic Code (M. Kennedy, 2016)

https://www.youtube.com/watch?v=_O23jlXsshs

Supplement Materials

The
Hitchhiker's

Guide
to
Python



<http://python-guide-kr.readthedocs.io/ko/latest/>

<http://docs.python-guide.org/en/latest/writing/style/>



Human knowledge belongs to the world.



Split & Join

Pythonic Code

최성철 교수

Director of TEAMLAB

Split

Split 함수

- String Type의 값을 나눠서 List 형태로 변환

Python shell

```
>>> items = 'zero one two three'.split() # 빈칸을 기준으로 문자열 나누기
>>> print(items)
['zero', 'one', 'two', 'three']

>>> example = 'python,jquery,javascript' # "," 을 기준으로 문자열 나누기
>>> example.split(",")
['python', 'jquery', 'javascript']

>>> a, b, c = example.split(",")
# 리스트에 있는 각 값을 a,b,c 변수로 unpacking
>>> example = 'cs50.gachon.edu'
>>> subdomain, domain, tld = example.split('.')
# "." 을 기준으로 문자열 나누기 → Unpacking
```

Join 함수

- String List를 합쳐 하나의 String으로 반환할 때 사용

```
>>> colors = ['red', 'blue', 'green', 'yellow']
>>> result = ''.join(colors)
>>> result
'redbluegreenyellow'

>>> result = ' '.join(colors) # 연결 시 빈칸 1칸으로 연결
>>> result
'red blue green yellow'

>>> result = ', '.join(colors) # 연결 시 ", "으로 연결
>>> result
'red, blue, green, yellow'

>>> result = '-'.join(colors) # 연결 시 "-"으로 연결
>>> result
'red-blue-green-yellow'
```

Python shell



Human knowledge belongs to the world.



List Comprehension

Pythonic Code

최성철 교수

Director of TEAMLAB

List comprehensions

- 기존 List 사용하여 간단히 다른 List를 만드는 기법
- 포괄적인 List, 포함되는 리스트라는 의미로 사용됨
- 파이썬에서 가장 많이 사용되는 기법 중 하나
- 일반적으로 for + append 보다 속도가 빠름

List comprehensions (1/4)

```
>>> result = []
>>> for i in range(10):
...     result.append(i)
...
>>> result
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Python shell

General Style

```
>>> result = [i for i in range(10)]
>>> result
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> result = [i for i in range(10) if i % 2 == 0]
>>> result
[0, 2, 4, 6, 8]
```

Python shell

List Comprehension

List comprehensions (2/4)

Python shell

```
>>> word_1 = "Hello"
>>> word_2 = "World"
>>> result = [i+j for i in word_1 for j in word_2]
      # Nested For loop
>>> result
['HW', 'Ho', 'Hr', 'Hl', 'Hd', 'eW', 'eo', 'er',
 'el', 'ed', 'lw', 'lo', 'lr', 'll', 'ld', 'lw',
 'lo', 'lr', 'll', 'ld', 'ow', 'oo', 'or', 'ol', 'od']
```

List comprehensions (3/4)

Python shell

```
>>> case_1 = ["A", "B", "C"]
>>> case_2 = ["D", "E", "A"]
>>> result = [i+j for i in case_1 for j in case_2]
>>> result
['AD', 'AE', 'AA', 'BD', 'BE', 'BA', 'CD', 'CE', 'CA']
>>> result = [i+j for i in case_1 for j in case_2 if not(i==j)]
# Filter: i랑 j과 같다면 List에 추가하지 않음
>>> result
['AD', 'AE', 'BD', 'BE', 'BA', 'CD', 'CE', 'CA']
>>> result.sort()
>>> result
['AD', 'AE', 'BA', 'BD', 'BE', 'CA', 'CD', 'CE']
```

List comprehensions (4/4)

```
>>> words = 'The quick brown fox  
jumps over the lazy dog'.split()  
# 문장을 빈칸 기준으로 나눠 list로 변환  
  
>>> print (words)  
['The', 'quick', 'brown', 'fox',  
'jumps', 'over', 'the', 'lazy', 'dog']  
>>>  
>>> stuff = [[w.upper(), w.lower(),  
len(w)] for w in words]  
  
# list의 각 element들을 대문자, 소문자, 길이  
로 변환하여 two dimensional list로 변환
```

```
>>> for i in stuff:  
...     print (i)  
...  
['THE', 'the', 3]  
['QUICK', 'quick', 5]  
['BROWN', 'brown', 5]  
['FOX', 'fox', 3]  
['JUMPS', 'jumps', 5]  
['OVER', 'over', 4]  
['THE', 'the', 3]  
['LAZY', 'lazy', 4]  
['DOG', 'dog', 3]
```

Two dimensional vs One dimensional

Python shell

```
>>> case_1 = ["A", "B", "C"]
>>> case_2 = ["D", "E", "A"]
['AD', 'AE', 'AA', 'BD', 'BE', 'BA', 'CD', 'CE', 'CA']
>>> result = [i+j for i in case_1 for j in case_2]
>>> result
>>> result = [ [i+j for i in case_1] for j in case_2]
>>> result
```



Human knowledge belongs to the world.

The background features a white humanoid robot with large, expressive eyes and a visible mechanical arm. The robot's hand is held up, displaying the binary code '01100110' in large blue digits.

Enumerate & Zip

Pythonic Code

최성철 교수
Director of TEAMLAB

Enumerate

Enumerate

- List의 element를 추출할 때 번호를 붙여서 추출

```
>>> for i, v in enumerate(['tic', 'tac', 'toe']):  
# list의 있는 index와 값을 unpacking  
...     print(i, v)  
...  
0 tic  
1 tac  
2 toe  
  
>>> mylist = ["a", "b", "c", "d"]  
>>> list(enumerate(mylist)) # list의 있는 index와 값을 unpacking하여 list로 저장  
[(0, 'a'), (1, 'b'), (2, 'c'), (3, 'd')]  
>>> {i:j for i,j in enumerate('Gachon University is an academic institute  
located in South Korea.'.split())}  
# 문장을 list로 만들고 list의 index와 값을 unpacking하여 dict로 저장  
{0: 'Gachon', 1: 'University', 2: 'is', 3: 'an', 4: 'academic', 5: 'institute',  
6: 'located', 7: 'in', 8: 'South', 9: 'Korea.'}
```

Python shell

Zip

Zip

두 개의 list의 값을 병렬적으로 추출함

```
>>> alist = ['a1', 'a2', 'a3']
>>> blist = ['b1', 'b2', 'b3']
>>> for a, b in zip(alist, blist): # 병렬적으로 값을 추출
...     print (a,b)
...
a1 b1
a2 b2
a3 b3

>>> a,b,c =zip((1,2,3),(10,20,30),(100,200,300)) #각 tuple의 같은 index 끼리
묶음
(1, 10, 100) (2, 20, 200) (3, 30, 300)

>>> [sum(x) for x in zip((1,2,3), (10,20,30), (100,200,300))]
# 각 Tuple 같은 index를 묶어 합을 list로 변환
[111, 222, 333]
```

Python shell

Enumerate & Zip

Python shell

```
>>> alist = ['a1', 'a2', 'a3']
>>> blist = ['b1', 'b2', 'b3']
>>>
>>> for i, (a, b) in enumerate(zip(alist, blist)):
...     print(i, a, b) # index alist[index] blist[index] 표시
...
0 a1 b1
1 a2 b2
2 a3 b3
```



Human knowledge belongs to the world.

The background features a white humanoid robot with a smooth, rounded body. It has two large, circular eyes with complex internal patterns. A hand is extended from its right arm, holding a series of blue, 3D-rendered binary digits (0s and 1s) between the thumb and forefinger. The robot's body is mostly white with some dark grey or black accents on its joints and the base of its legs.

Lambda & MapReduce

Pythonic Code

최성철 교수

Director of TEAMLAB

Lambda

Lambda

- 함수 이름 없이, 함수처럼 쓸 수 있는 익명함수
- 수학의 람다 대수에서 유래함

General function

```
def f(x, y):  
    return x + y
```

```
print(f(1, 4))
```

Lambda function

```
f = lambda x, y: x + y  
print(f(1, 4))
```

Lambda

- Python 3부터는 권장하지는 않으나 여전히 많이 쓰임

```
f = lambda x, y: x + y  
print(f(1, 4))
```

```
f = lambda x: x ** 2  
print(f(3))
```

```
f = lambda x: x / 2  
print(f(3))
```

```
print((lambda x: x + 1)(5))
```

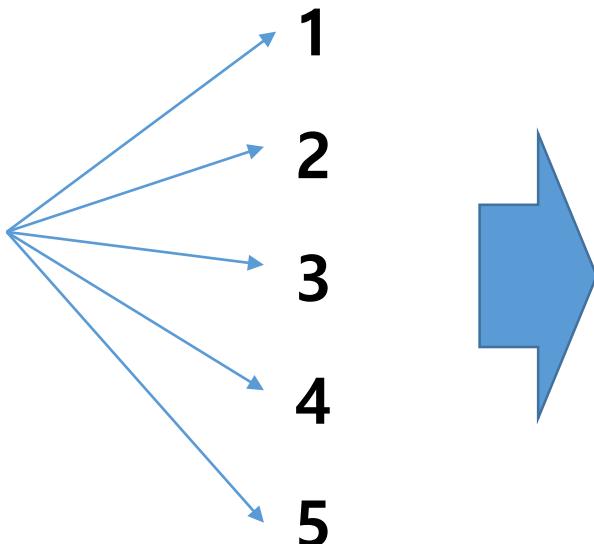
Map & Reduce

Map function

- Sequence 자료형 각 element에 동일한 function을 적용함

`map(function_name, list_data)`

`lambda x: x ** 2`



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

`f = lambda x: x ** 2`

`print(list(map(f, ex)))`

`f = lambda x, y: x + y`

`print(list(map(f, ex, ex)))`

Map function

- 두 개 이상의 list에도 적용 가능함, if filter도 사용 가능

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

```
f = lambda x, y: x + y
```

```
print(list(map(f, ex, ex)))
```

```
list(  
    map(  
        lambda x: x ** 2 if x % 2 == 0  
        else x,  
        ex)  
)
```

Map function

- python3 는 iteration을 생성 → list을 붙여줘야 list 사용 가능
- 실행시점의 값을 생성, 메모리 효율적

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

```
print(list(map(lambda x: x+x, ex)))
```

```
print((map(lambda x: x+x, ex)))
```

```
f = lambda x: x ** 2
```

```
print(map(f, ex))
```

```
for i in map(f, ex):
```

```
    print(i)
```

```
result = map(f, ex)
```

```
print(next(result))
```

Reduce function

- map function과 달리 list에 똑같은 함수를 적용해서 통합

```
from functools import reduce
```

```
print(reduce(lambda x, y: x+y, [1, 2, 3, 4, 5]))
```

1	2	3	4	5
---	---	---	---	---

Summary

- Lambda, map, reduce는 간단한 코드로 다양한 기능을 제공
- 그러나 코드의 직관성이 떨어져서 lambda나 reduce는 **python3에서 사용을 권장하지 않음**
- Legacy library나 다양한 머신러닝 코드에서 여전히 사용중



Human knowledge belongs to the world.

A white humanoid robot with large, expressive eyes and a hand held up towards the right side of the frame. The hand is open, with fingers slightly curled, holding a large, three-dimensional blue digital display showing the binary sequence '011000110110'. The robot's body is white and has a smooth, rounded design.

Asterisk

Pythonic Code

최성철 교수

Director of TEAMLAB

Asterisk

- 흔히 알고 있는 * 를 의미함
- 단순 곱셈, 제곱연산, 가변 인자 활용 등 다양하게 사용됨

*args

```
def asterisk_test(a, *args):  
    print(a, args)  
    print(type(args))  
  
asterisk_test(1,2,3,4,5,6)
```

**kargs

```
def asterisk_test(a, **kargs):  
    print(a, kargs)  
    print(type(kargs))  
  
asterisk_test(1, b=2, c=3,  
d=4, e=5, f=6)
```

Asterisk – unpacking a container

- tuple, dict 등 자료형에 들어가 있는 값을 unpacking
- 함수의 입력값, zip 등에 유용하게 사용 가능

```
def asterisk_test(a, *args):  
    print(a, args)  
    print(type(args))
```

```
asterisk_test(1, *(2,3,4,5,6))
```

```
def asterisk_test(a, args):  
    print(a, *args)  
    print(type(args))
```

```
asterisk_test(1, (2,3,4,5,6))
```

Asterisk – unpacking a container

```
a, b, c = ([1, 2], [3, 4], [5, 6])  
print(a, b, c)
```

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

```
def asterisk_test(a, b, c, d,):  
    print(a, b, c, d)  
  
data = {"b":1 , "c":2, "d":3}  
asterisk_test(10, **data)
```

```
for data in zip(*([1, 2], [3, 4], [5, 6])):  
    print(data)
```



Human knowledge belongs to the world.

The background features a white humanoid robot with a smooth, rounded body. It has two large, circular eyes with complex internal patterns, giving it a thoughtful appearance. A hand is extended from its right arm, holding a series of blue, 3D-rendered binary digits (0s and 1s) between the thumb and forefinger. The robot's skin is a light grey or white, and it appears to be made of a soft, possibly fabric-like material. The overall aesthetic is clean and futuristic.

Linear algebra concepts

Pythonic Code

최성철 교수

Director of TEAMLAB

Linear Algebra

선형 대수

Linear Algebra

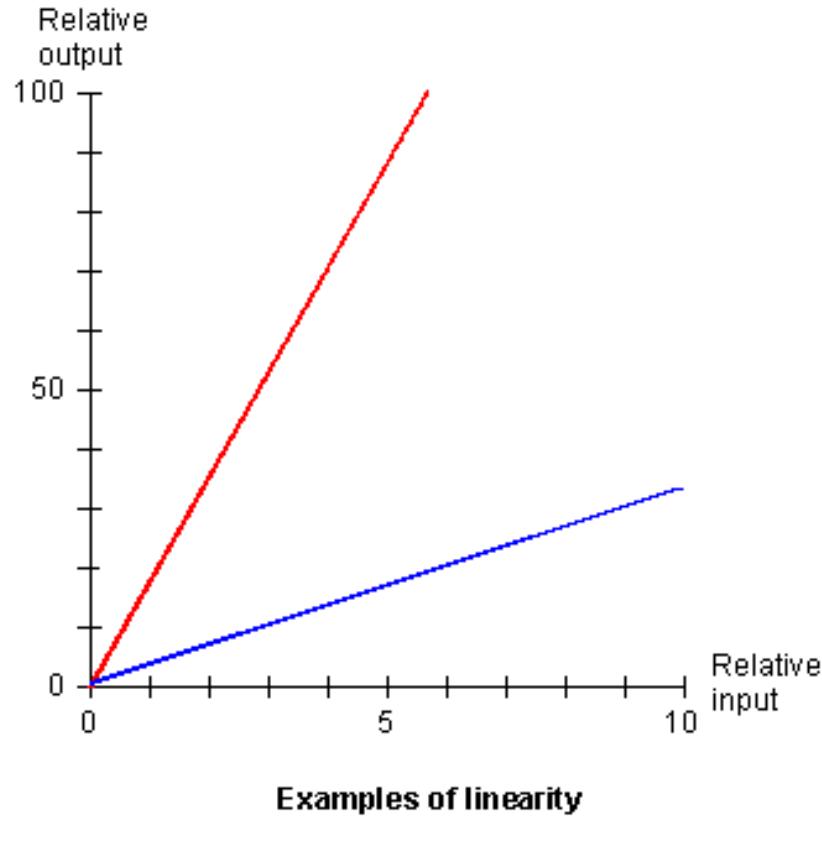
선형(線型)

- 직선 또는 그와 비슷한 성질을 가진 대상
- 일반적으로 1차함수 형태 = 선형성

대수학(代數學)

- 문자에 숫자를 대입하여 푸는 문제
- 1개 이상의 변수를 가진 다항방정식을 푸는 문제

Linear Algebra



Examples:

$$1) \text{ orange } 3a + 6y \\ = 3 (a + 2y)$$

$$2) \text{ orange } fs + qr - fr - qs \\ = fs - fr + qr - qs \\ = f(s - r) + q(r - s) \\ = (f - q)(s - r)$$

Vector

Vector

- “배달, 운반하다”의 의미를 가진 단어가 어원
- 크기와 방향을 모두 가지는 것을 벡터(vector)
- 크기만 가지는 경우 스칼라(scalar)라고 지칭함

Vector

$$u = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, v = \begin{bmatrix} 1 \\ -3 \end{bmatrix}, w = \begin{bmatrix} -1 \\ -2 \end{bmatrix}$$

일반적인 vector의 표현방법

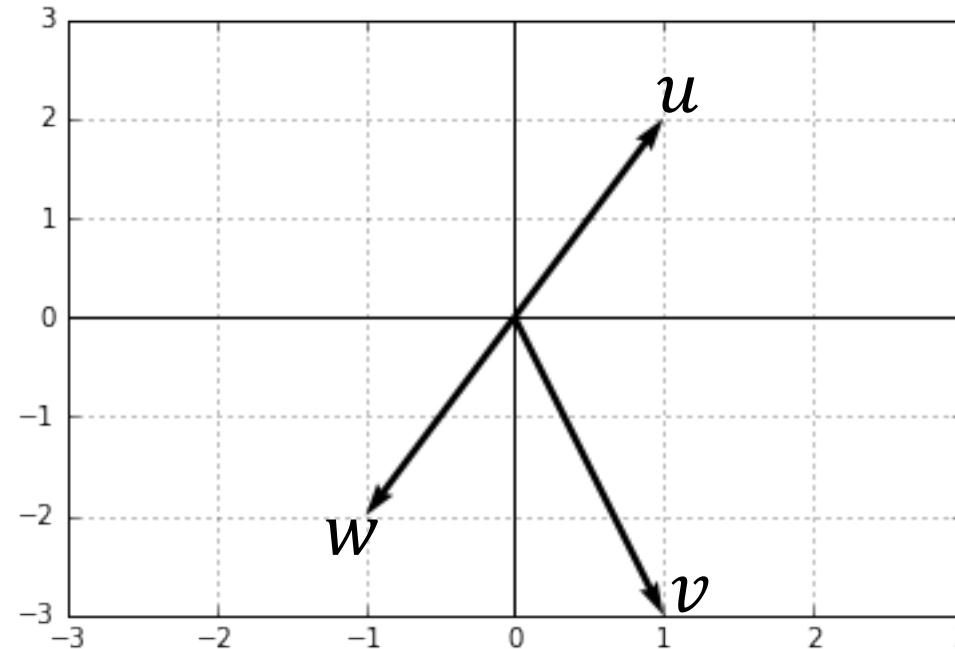
Vector

- 고등학교때는 평면 좌표 위의 Vector를 배움
- Vector라 하면 다음과 같이 상상됨

$$u = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$v = \begin{bmatrix} 1 \\ -3 \end{bmatrix}$$

$$w = \begin{bmatrix} -1 \\ -2 \end{bmatrix}$$



Vector

- 그러나 일반적인 연산에서는 2개 이상의 변수를 처리
- 이 변수 하나하나가 **vector**의 **element**로 표시됨

$$\mathbb{R}^4 \ni [1, 2, -1.0, 3.14] \quad \text{또는} \quad \mathbb{R}^4 \ni \begin{bmatrix} 1 \\ 2 \\ -1.0 \\ 3.14 \end{bmatrix}$$

Vector

- $[1, 2, -1.0, 3.14]$ 은 \mathbb{R} 상의 4-Vector 또는 4-dimentional vector
- 이 변수 하나하나가 vector의 element로 표시됨
- \mathbb{R}^4 : 4개의 실수를 원소를 가지는 벡터 집합

$$\mathbb{R}^4 \ni [1, 2, -1.0, 3.14] \quad \text{또는} \quad \mathbb{R}^4 \ni \begin{bmatrix} 1 \\ 2 \\ -1.0 \\ 3.14 \end{bmatrix}$$

(\mathbb{R}^n 은 실수집합)

Vector의 표현

3-dimensional column vector

$$\begin{bmatrix} 9 \\ 2 \\ 3 \end{bmatrix}$$

3-dimensional row vector

$$[9, 2, 3]$$

3-dimensional zero vector

$$[0, 0, 0]$$

Vector의 계산

n-vector들의 덧셈

- Element 위치를 대응하는 원소들의 덧셈으로 정의됨

$$[u_1, u_2, \dots, u_n] + [v_1, v_2, \dots, v_n] = [u_1 + v_1, u_2 + v_2, \dots, u_n + v_n]$$

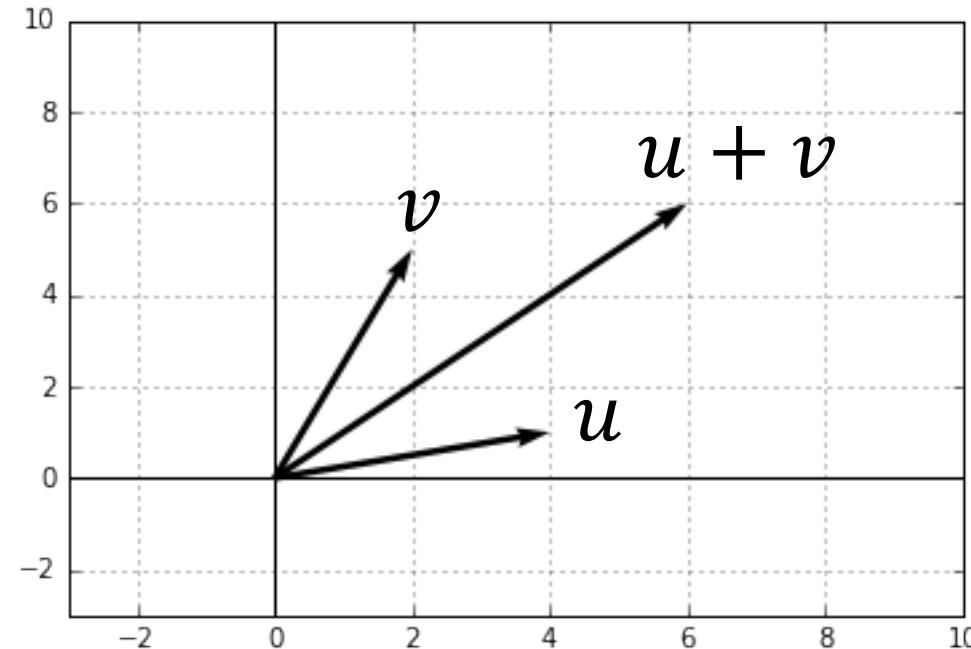
Example

$$[2, 2] + [2, 3] + [3, 5] = [7, 10]$$

Vector의 계산

$$u = \begin{bmatrix} 4 \\ 1 \end{bmatrix} \quad v = \begin{bmatrix} 2 \\ 5 \end{bmatrix}$$

$$[4, 1] + [2, 5] = [6, 6]$$



Vector의 계산

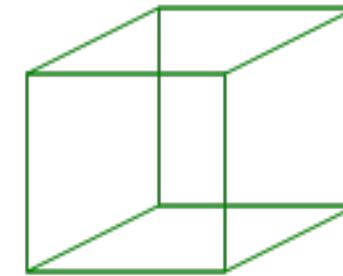
1 Dimension



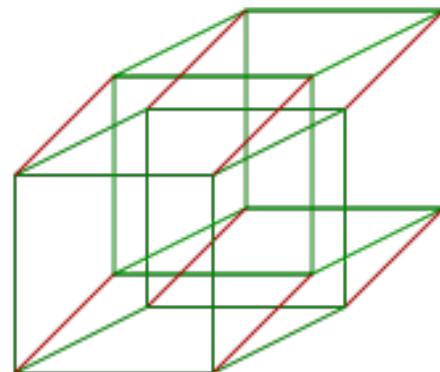
2 Dimensions



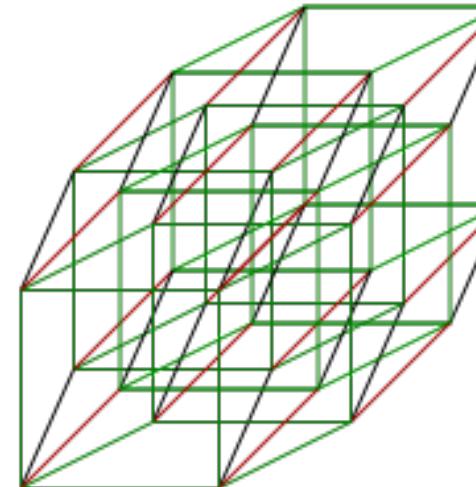
3 Dimensions



4 Dimensions



5 Dimensions



Associativity / Commutativity

임의의 벡터 u, v, w 은 아래와 같이 결합 법칙과 교환 법칙이 성립됨

결합 법칙 $(u + v) + w = u + (v + w)$

교환 법칙 $u + v = v + u$

Example

$$([2, 2] + [2, 3]) + [3, 5] = [2, 2] + ([2, 3] + [3, 5]) = [7, 10]$$

$$[2, 2] + [2, 3] = [2, 3] + [2, 2]$$

Scalar-Vector product

Proposition

$$\alpha(u + v) = \alpha u + \alpha v$$

Example

$$2([1, 2, 3] + [4, 4, 4]) = 2[4, 6, 7] = [8, 12, 14]$$

Matrix



Matrix

- 격자
- 수학에서는 사각형으로 된 수의 배열을 지칭
- 한 개 이상의 벡터(vector) = Matrix

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \begin{bmatrix} 1 \\ -2 \end{bmatrix}, [2 \quad 1]$$

일반적인 matrix의 표현방법

Matrix Representation

- Matrix는 m개의 행(row) , n개의 열(column)로 구성
- $m \times n$ 행렬이라고 함 (m by n 이라고 읽음)

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Matrix Representation

행렬 A의 i 행, j 열의 값을 A의 ij 번째 element라 함 a_{ij} 로 표시함

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad \begin{aligned} a_{11} &= 1 \\ a_{23} &= 6 \\ a_{31} &= 7 \end{aligned}$$

Matrix Equal(등치)

- 두개의 행렬 A와 B가

1) A와 B의 같은 크기이면서

2) 모든 i, j 에 대하여 같은 값을 가지면,

즉 $a_{ij} = b_{ij}$ 이면 등치라고 한다

Matrix Equal(등치)

$$A = \begin{bmatrix} 3 & 6 \\ 4 & 5 \end{bmatrix} \quad B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

이면 $A = B$ 이기 위한 조건은 아래와 같다.

$$b_{11} = 3, b_{12} = 6, b_{21} = 4, b_{22} = 5$$

Matrix Addition

두개의 행렬 A와 B가 같은 크기

→ $C = A + B$ 가 $a_{ij} + b_{ij}$ 로 구성됨

$$A = \begin{bmatrix} 3 & 6 \\ 4 & 5 \end{bmatrix} \quad B = \begin{bmatrix} 5 & 8 \\ 6 & 7 \end{bmatrix}$$

$$C = A + B = \begin{bmatrix} 3 & 6 \\ 4 & 5 \end{bmatrix} + \begin{bmatrix} 5 & 8 \\ 6 & 7 \end{bmatrix} = \begin{bmatrix} 8 & 14 \\ 10 & 12 \end{bmatrix}$$

Scalar-Matrix Product

주어진 행렬 A 에 Scalar c 를 곱하면

모든 ij 에 대해 $c \times a_{ij}$ 로 구성된 행렬이 구성됨

$$A = \begin{bmatrix} 3 & 6 \\ 4 & 5 \end{bmatrix} \quad c = 4$$

$$c \times A = 4 \times \begin{bmatrix} 3 & 6 \\ 4 & 5 \end{bmatrix} = \begin{bmatrix} 12 & 24 \\ 16 & 20 \end{bmatrix}$$

Matrix Transpose (전치 행렬)

- 주어진 $m \times n$ 의 행과 열을 바꾸어 만든 행렬
- 행렬 A 의 전치 행렬은 A^T 로 표시

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \ddots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad A^T = \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{n1} \\ a_{12} & a_{22} & \cdots & a_{n2} \\ \vdots & \ddots & & \vdots \\ a_{1m} & a_{2m} & \cdots & a_{nm} \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \quad A^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

Matrix Product (행렬 곱셈)

- 앞 행렬의 열과 뒤 행렬의 행을 dot product
- 앞 행렬 열의 수와 뒤 행렬의 행 수가 동일 해야만 계산 가능

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 56 \end{bmatrix}$$

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad N = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad M \times N = ?$$



Human knowledge belongs to the world.

A white humanoid robot with a large head and a white body. It has two large, expressive eyes and a small mouth. Its right hand is extended towards the viewer, with blue 3D-rendered binary digits (0s and 1s) floating around it. The digits are arranged in a 3D perspective, with '01100' at the top, '00110' in the middle, and '10010' at the bottom.

Linear algebra codes

Pythonic Code

최성철 교수

Director of TEAMLAB

Vector representation of python

- Vector를 파이썬으로 표시하는 다양한 방법 존재

```
vector_a = [1, 2, 10] # List로 표현했을 경우  
vector_b = (1, 2, 10) # Tuple로 표현했을 경우  
vector_c = {'x': 1, 'y': 1, 'z': 10} # dict 표현했을 경우  
  
print(vector_a, vector_b, vector_c)
```

- 최선의 방법은 없음
- 값의 변경 유무, 속성값 유무에 따라 선택할 수 있음
- 본 수업에서는 기본적으로 list로 vector 연산 실시

Vector의 계산

```
u = [2, 2]
v = [2, 3]
z = [3, 5]
result = []
for i in range(len(u)):
    result.append(u[i] + v[i] + z[i])
print(result)
```

$$[2, 2] + [2, 3] + [3, 5] = [7, 10]$$

이런 코드는 쓰면 안됨
파이썬 답지 못하고
안 아름다움

Vector handling with python

- Python은 특유의 간결성이 최대의 장점
- Vector와 같은 수학 연산을 복잡하게 표현한다면 사용이 어려움
- 최대한 파이썬만의 특징을 살려서 간단하게 연산을 표시
- Comprehension과 zip 같은 pythonic technique을 적극 활용

Vector의 계산

```
u = [2, 2]  
v = [2, 3]  
z = [3, 5]
```

$$[2, 2] + [2, 3] + [3, 5] = [7, 10]$$

```
result = [sum(t) for t in zip(u,v,z)]  
print(result)
```

Vector의 계산: Scalar-Vector product

```
u = [1, 2, 3]  2([1,2,3] + [4,4,4]) = 2[5,6,7] = [10,12,14 ]  
v = [4, 4, 4]  
alpha = 2
```

```
result = [alpha*sum(t) for t in zip(u,v)]  
print(result)
```

Matrix representation of python

- Matrix 역시 Python으로 표시하는 다양한 방법이 존재

```
matrix_a = [[3, 6], [4, 5]] # List로 표현했을 경우  
matrix_b = [(3, 6), (4, 5)] # Tuple로 표현했을 경우  
matrix_c = {(0, 0): 3, (0, 1): 6, (1, 0): 4, (1, 1): 5} # dict 표현했을 경우
```

- 특히 dict로 표현할 때는 무궁무진한 방법이 있음
- 본 수업에서는 기본적으로 two-dimensional list 형태로 표현함
- [[1번째 row], [2번째 row], [3번째 row]]

Matrix의 계산: Matrix addition

$$C = A + B = \begin{bmatrix} 3 & 6 \\ 4 & 5 \end{bmatrix} + \begin{bmatrix} 5 & 8 \\ 6 & 7 \end{bmatrix} = \begin{bmatrix} 8 & 14 \\ 10 & 12 \end{bmatrix}$$

```
matrix_a = [[3, 6], [4, 5]]  
matrix_b = [[5, 8], [6, 7]]  
result = [[sum(row) for row in zip(*t)] for t in zip(matrix_a, matrix_b)]  
  
print(result)
```

Matrix의 계산: Scalar-Matrix Product

$$\alpha \times A = 4 \times \begin{bmatrix} 3 & 6 \\ 4 & 5 \end{bmatrix} = \begin{bmatrix} 12 & 24 \\ 16 & 20 \end{bmatrix}$$

```
matrix_a = [[3, 6], [4, 5]]  
alpha = 4  
result = [[alpha * element for element in t] for t in matrix_a]  
  
print(result)
```

Matrix의 계산: Matrix Transpose

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, A^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

```
matrix_a = [[1, 2, 3], [4, 5, 6]]  
result = [ [element for element in t] for t in zip(*matrix_a) ]  
print(result)
```

Matrix의 계산: Matrix Product

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, B = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \text{ 이면 } C = A \times B = \begin{bmatrix} 1 & 1 & 2 \\ 2 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 1 & 3 \end{bmatrix} = \begin{bmatrix} 5 & 8 \\ 5 & 6 \end{bmatrix}$$

```
matrix_a = [[1, 1, 2], [2, 1, 1]]  
matrix_b = [[1, 1], [2, 1], [1, 3]]  
result = [[sum(a * b for a, b in zip(row_a, column_b))  
          for column_b in zip(*matrix_b)] for row_a in matrix_a]  
  
print(result)
```

두 개이상의
Argument가 존재할 때는?



Human knowledge belongs to the world.

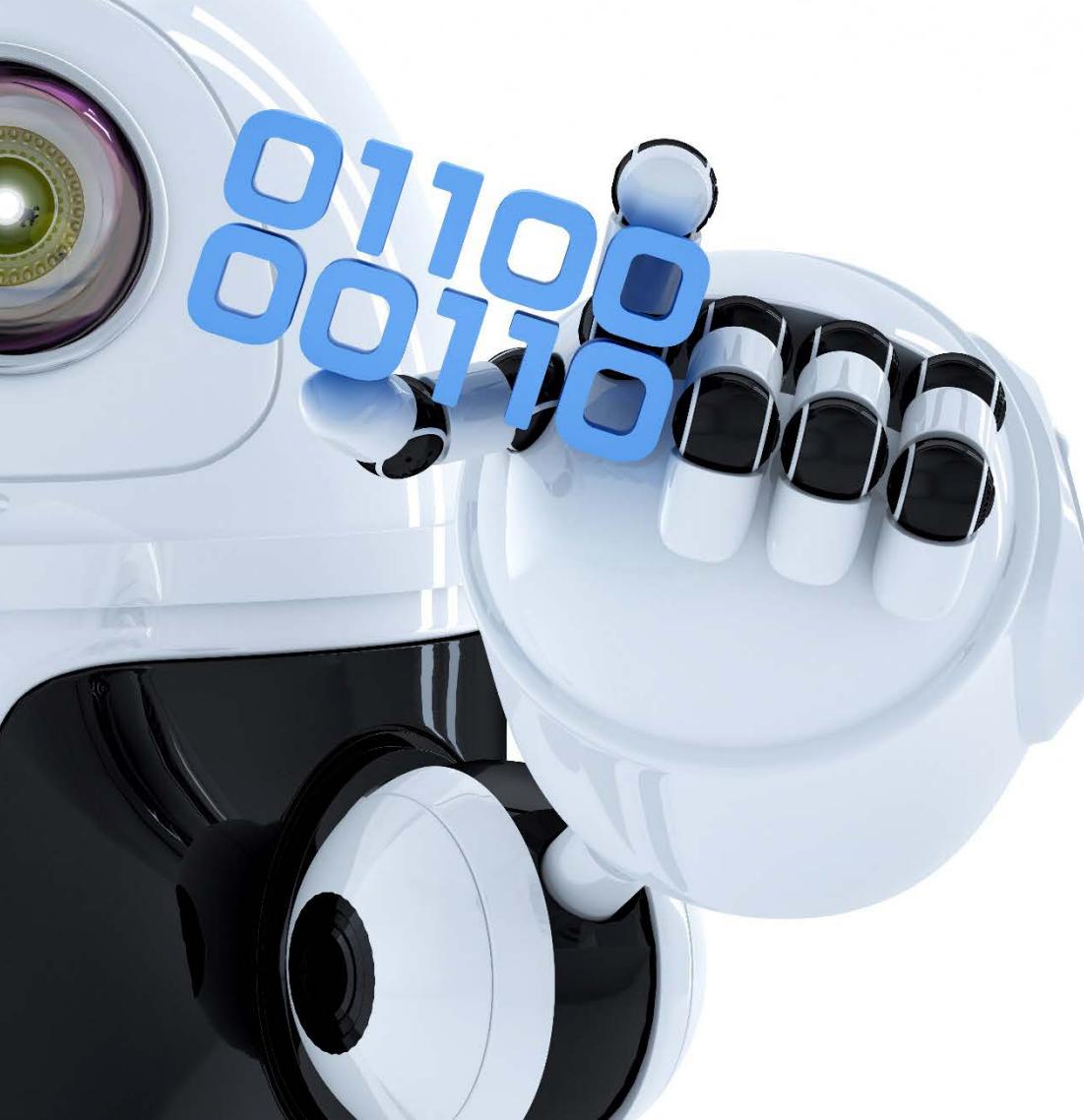
Overview

Object Oriented Programming

최성철 교수

Director of TEAMLAB

01100
00110



프로그램을 여럿이 개발할 경우

코드를 어떻게 작성해야 할까?

만들어 놓은 코드를
재사용하고 싶다!

프로그램을 기능별로 나누어

재사용하는 방법

① 함수 ② 객체 ③ 모듈

클래스와 객체

- 객체 지향 언어의 이해 -

[생각해보기]

수강신청 프로그램을 작성한다.
어떻게 해야 할까?

[생각해보기]

- ① 수강신청이 시작부터 끝까지 순서대로 작성
- ② 수강신청 관련 **주체들**(교수, 학생, 관리자)의
행동(수강신청, 과목 입력)과 **데이터**(수강과목, 강의 과목)들을
중심으로 프로그램 작성 후 연결

[생각해보기]

두 가지 모두 가능

최근엔 ②번 방식이 주류

이러한 기법을

객체 지향 프로그램 이라 함

객체 지향 프로그램

- Object-Oriented Programming, OOP
- 객체: 실생활에서 일종의 물건
속성(Attribute)와 행동(Action)을 가짐
- OOP는 이러한 객체 개념을 프로그램으로 표현
속성은 variable, 행동은 함수로 표현됨
- 파이썬 역시 객체 지향 프로그램 언어

객체 지향 프로그램

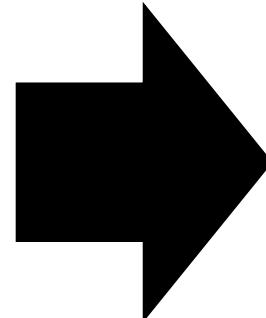
- 인공지능 축구 프로그램을 작성한다고 가정
- 객체 종류: 팀, 선수, 심판, 공
- Action : 선수 – 공을 차다, 패스하다.
심판 – 휘슬을 불다, 경고를 주다.
- Attribute : 선수 – 선수 이름, 포지션, 소속팀
팀 – 팀 이름, 팀 연고지, 팀소속 선수

객체 지향 프로그래밍

- OOP는 설계도에 해당하는 클래스(class)와 실제 구현체인 인스턴스(instance)로 나눔



붕어빵틀
(Class)



붕어빵
(인스턴스)

객체 지향 프로그래밍



고구마



말차고구마



말차팥



미니 타이야끼



블루베리 & 크림치즈



커스타드



팥 & 크림치즈



팥 & 호두

<http://rupil.egloos.com/m/5697622>

**직접 구현을 해봐야
알 수 있음**



Human knowledge belongs to the world.

Objects in Python

Object Oriented Programming

최성철 교수

Director of TEAMLAB

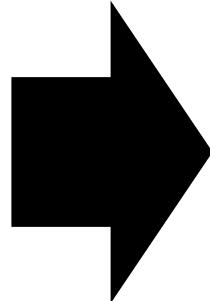
01100
00110



여러분 머릿속엔...



붕어빵틀
(Class)



붕어빵
(인스턴스)

코드로 봐야 안다!

Objects in Python

Class 구현하기 in Python

- 축구 선수 정보를 Class로 구현하기

```
class SoccerPlayer(object):
    def __init__(self, name, position, back_number):
        self.name = name
        self.position = position
        self.back_number = back_number

    def change_back_number(self, new_number):
        print("선수의 등번호를 변경합니다 : From %d to %d" % (self.back_number, new_number))
        self.back_number = new_number
```

Class 구현하기 in Python - 선언

`class` 선언은

```
class SoccerPlayer(object):
```



class 예약어 class 이름 상속받는 객체명

[알아두면 상식] Naming

- 변수와 Class명 함수명은 짓는 방식이 존재
- **snake_case** : 띄워쓰기 부분에 “_”를 추가
뱀처럼 늘여쓰기, 파이썬 함수/변수명에 사용
- **CamelCase**: 띄워쓰기 부분에 대문자
낙타의 등 모양, 파이썬 Class명에 사용

<https://goo.gl/sVb2yd>

Class 구현하기 in Python - Attribute

Attribute 추가는 `__init__`, self와 함께!

`__init__`은 객체 초기화 예약 함수

```
class SoccerPlayer(object):
    def __init__(self, name, position, back_number):
        self.name = name
        self.position = position
        self.back_number = back_number
```

[알아두면 상식] 파이썬에서 __ 의미

- __는 특수한 예약 함수나 변수에 사용됨

예) __main__, __add__, __str__

```
class SoccerPlayer(object):  
  
    def __str__(self):  
        return "Hello, My name is %s. I play in %s in center" % w  
        (self.name, self.position)  
  
jinyun = SoccerPlayer("Jinyun", "MF", 10)  
print(jinyun)
```

Class 구현하기 in Python - Function

Function(Action) 추가는 기존 함수와 같으나,
반드시 **self**를 추가해야만 class 함수로 인정됨

```
class SoccerPlayer(object):
    def change_back_number(self, new_number):
        print("선수의 등번호를 변경합니다 :
              From %d to %d" % (
            self.back_number, new_number))
        self.back_number = new_number
```

Objects(Instance) 사용하기

Object 이름 선언과 함께 초기값 입력 하기

```
jinhyun = SoccerPlayer("Jinhyun", "MF", 10)
```

객체명

Class명

init 함수 Interface, 초기값

```
def __init__(self, name, position, back_number):
```

```
class SoccerPlayer(object):
```

.....

```
jinhyun = SoccerPlayer("Jinhyun", "MF", 10)
print("현재 선수의 등번호는 :", jinhyun.back_number)
jinhyun.change_back_number(5)
print("현재 선수의 등번호는 :", jinhyun.back_number)
```

Class 구현하기 in Python

```
class SoccerPlayer(object):
    def __init__(self, name, position, back_number):
        self.name = name
        self.position = position
        self.back_number = back_number

    def change_back_number(self, new_number):
        print("선수의 등번호를 변경합니다 : From %d to %d" % (self.back_number, new_number))
        self.back_number = new_number
```

```
jinhoon = SoccerPlayer("Jinhoon", "MF", 10)
print("현재 선수의 등번호는 :", jinhoon.back_number)
jinhoon.change_back_number(5)
print("현재 선수의 등번호는 :", jinhoon.back_number)
```

이렇게 하면
뭐가 좋나요?

5명 Soccer Player 정보 저장하기

- 이차원 리스트 사용해보기

```
names = ["Jin", "Sungchul", "Ronaldo", "Hong", "Seo"]
positions = ["MF", "DF", "CF", "WF", "GK"]
numbers = [10, 15, 20, 3, 1]
```

```
players = [[name, position, number] for name, position, number in zip(names, positions, numbers)]
print(players)
print(players[0])
```

- Class로 선언하기

```
player_objects = [SoccerPlayer(name, position, number) for name, position, number in zip(names, positions, numbers)]
print(player_objects[0])
```



Human knowledge belongs to the world.



Lab: Notebook

Object Oriented Programming

최성철 교수
Director of TEAMLAB

구현 가능한 OOP 만들기 - 노트북

- Note를 정리하는 프로그램
- 사용자는 Note에 뭔가를 적을 수 있다.
- Note에는 Content가 있고, 내용을 제거할 수 있다.
- Note는 Notebook에 삽입된다.
- Notebook은 Note가 삽입될 때 페이지를 생성하며, 최고 300페이지까지 저장 가능하다
- 300 페이지가 넘으면 더 이상 노트를 삽입하지 못한다.

Class

method

Notebook

`add_note`
`remove_note`
`get_number_of_pages`

Note

`write_content`
`remove_all`

variable

`title`
`page_number`
`notes`

content

```
class Note(object):
    def __init__(self, content = None):
        self.content = content

    def write_content(self, content):
        self.content = content

    def remove_all(self):
        self.content = ""

    def __str__(self):
        return self.content
```

content

write_content

remove_all

```
class NoteBook(object):
    def __init__(self, title):
        self.title = title
        self.page_number = 1
        self.notes = {}

    def add_note(self, note, page = 0):
        if self.page_number < 300:
            if page == 0:
                self.notes[self.page_number] = note
                self.page_number += 1
            else:
                self.notes = {page : note}
                self.page_number += 1
        else:
            print("Page가 모두 채워졌습니다.")

    def remove_note(self, page_number):
        if page_number in self.notes.keys():
            return self.notes.pop(page_number)
        else:
            print("해당 페이지는 존재하지 않습니다.")

    def get_number_of_pages(self):
        return len(self.notes.keys())
```

add_note

remove_note

get_number_of_pages



Human knowledge belongs to the world.

Characteristics

Object Oriented Programming

최성철 교수

Director of TEAMLAB

01100
00110



객체 지향 언어의 특징

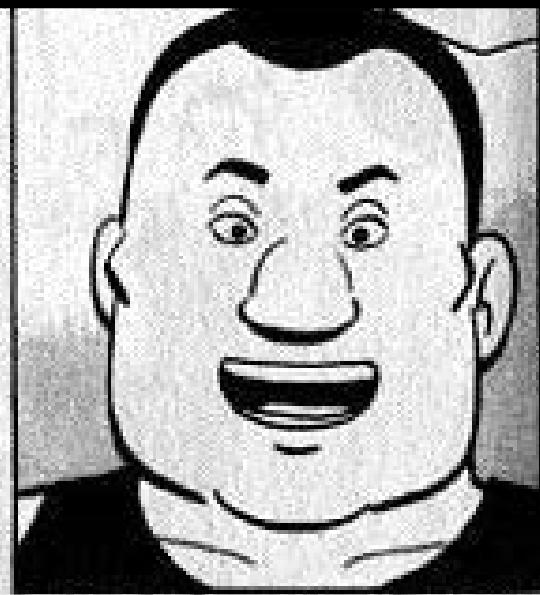
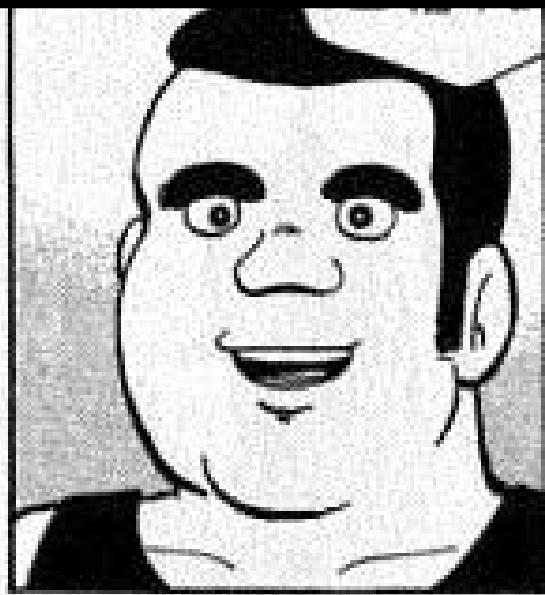
실제 세상을 모델링

이를 위한 특징들

Inheritance

Polymorphism

Visibility



<https://goo.gl/RmNwUj>

오늘 어차피 모든 걸
이해 못함

Inheritance

상속

상속 (Inheritance)

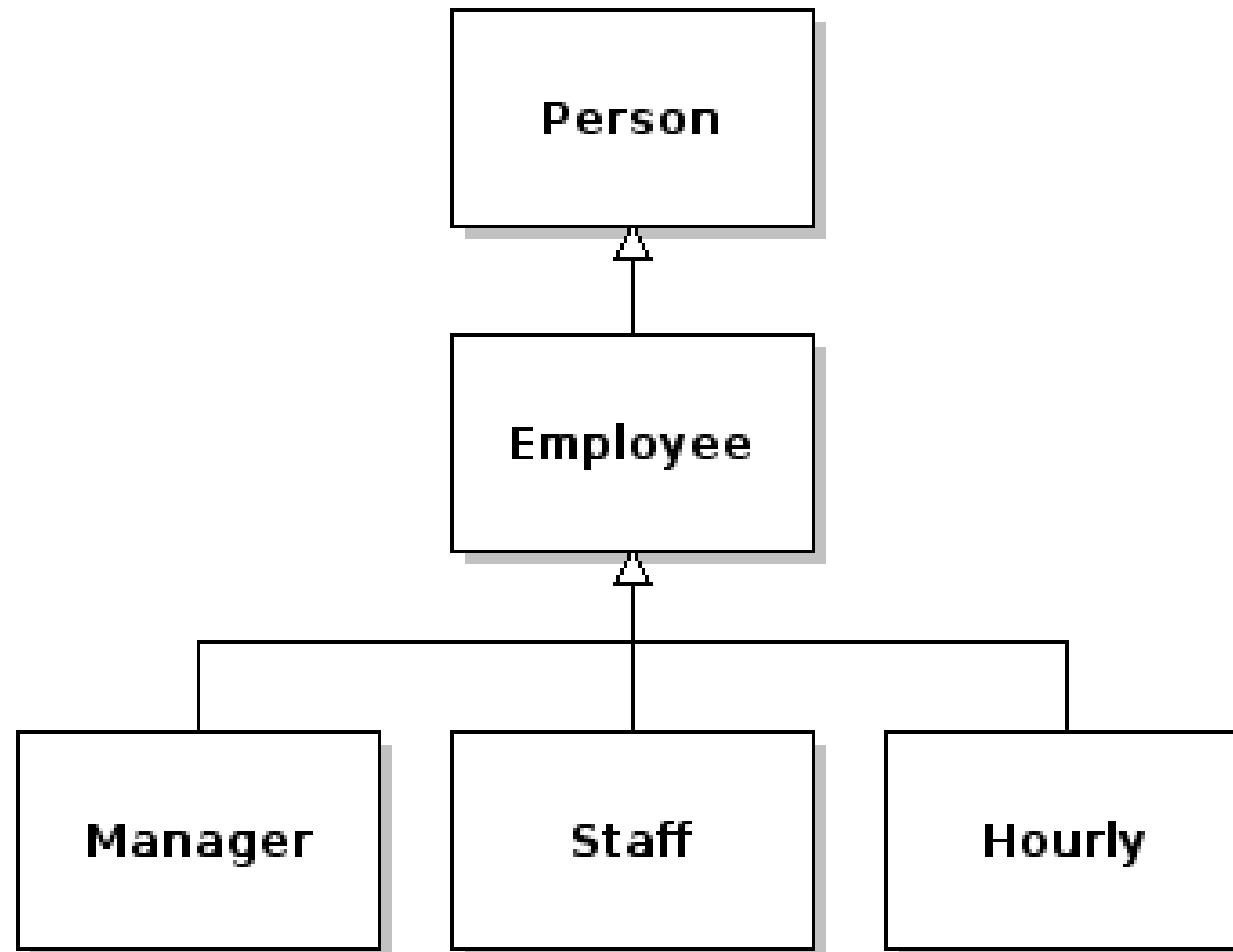
- 부모클래스로 부터 속성과 Method를 물려받은 자식 클래스를 생성 하는 것

```
class Person(object):  
    def __init__(self, name, age):  
        self.name = name  
        self.age = age
```

```
class Korean(Person):  
    pass
```

```
first_korean = Korean("Sungchul", 35)  
print(first_korean.name)
```

OOP 특징 - 상속 (Inheritance)



OOP 특징 - 상속 예제

```
class Person(object): # 부모 클래스 Person 선언
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender

    def about_me(self): # Method 선언
        print("저의 이름은 ", self.name, "이구요, 제 나이는 ", str(self.age), "살 입니다.")
```

Source: <http://blog.eairship.kr/m/post/286>

OOP 특징 - 상속 예제

```
class Employee(Person): # 부모 클래스 Person으로 부터 상속
    def __init__(self, name, age, gender, salary, hire_date):
        super().__init__(name, age, gender) # 부모객체 사용
        self.salary = salary
        self.hire_date = hire_date # 속성값 추가

    def do_work(self): # 새로운 메서드 추가
        print("열심히 일을 합니다.")

    def about_me(self): # 부모 클래스 함수 재정의
        super().about_me() # 부모 클래스 함수 사용
        print("제 급여는 ", self.salary, "원 이구요, 제 입사일은 ", self.hire_date,
              "입니다.")
```

Source: <http://blog.eairship.kr/m/post/286>

Polymorphism

다형성

다형성 (Polymorphism)

- 같은 이름 메소드의 내부 로직을 다르게 작성
- Dynamic Typing 특성으로 인해 파이썬에서는 같은 부모클래스의 상속에서 주로 발생함
- 중요한 OO의 개념 그러나 너무 깊이 알 필요X

다형성 (Polymorphism)

```
class Animal:  
    def __init__(self, name): # Constructor of the class  
        self.name = name  
  
    def talk(self): # Abstract method, defined by convention only  
        raise NotImplementedError("Subclass must implement abstract method")  
  
    class Cat(Animal):  
        def talk(self):  
            return 'Meow!'  
  
    class Dog(Animal):  
        def talk(self):  
            return 'Woof! Woof!'  
  
animals = [Cat('Missy'),  
          Cat('Mr. Mistoffelees'),  
          Dog('Lassie')]  
  
for animal in animals:  
    print(animal.name + ': ' + animal.talk())
```

<https://goo.gl/lBy9uf>

Visibility
가시성

가시성 (Visibility)

- 객체의 정보를 볼 수 있는 레벨을 조절하는 것
- 누구나 객체 안에 모든 변수를 볼 필요가 없음
 - 1) 객체를 사용하는 사용자가 임의로 정보 수정
 - 2) 필요 없는 정보에는 접근 할 필요가 없음
 - 3) 만약 제품으로 판매한다면? 소스의 보호

[알아두면 상식] Encapsulation

- 캡슐화 또는 정보 은닉 (Information Hiding)
- Class를 설계할 때, 클래스 간 간섭/정보공유의 최소화
- 심판 클래스가 축구선수 클래스 가족 정보를 알아야 하나?
- 캡슐을 던지듯, 인터페이스만 알아서 써야함

[알아두면 상식] Encapsulation



Visibility Example

- Product 객체를 Inventory 객체에 추가
- Inventory에는 오직 Product 객체만 들어감
- Inventory에 Product가 몇 개인지 확인이 필요
- Inventory에 Product items는 직접 접근이 불가

Visibility Example

```
class Product(object):
    pass

class Inventory(object):
    def __init__(self):
        self.__items = []

    def add_new_item(self, product):
        if type(product) == Product:
            self.__items.append(product)
            print("new item added")
        else:
            raise ValueError("Invalid Item")

    def get_number_of_items(self):
        return len(self.__items)
```

Visibility Example

```
my_inventory = Inventory()  
my_inventory.add_new_item(Product())  
my_inventory.add_new_item(Product())  
print(my_inventory.get_number_of_items())
```

```
print(my_inventory.__items)  
my_inventory.add_new_item(object)
```

Visibility Example

- Product 객체를 Inventory 객체에 추가
- Inventory에는 오직 Product 객체만 들어감
- Inventory에 Product가 몇 개인지 확인이 필요
- Inventory에 Product **items** 접근 허용

Visibility Example

```
class Inventory(object):
    def __init__(self):
        self.__items = []

    @property
    def items(self):
        return self.__items

my_inventory = Inventory()
my_inventory.add_new_item(Product())
my_inventory.add_new_item(Product())
print(my_inventory.get_number_of_items())

items = my_inventory.items
items.append(Product())
print(my_inventory.get_number_of_items())
```

Visibility

```
class Inventory(object):  
    def __init__(self):  
        self.__items = []
```

Private 변수로 선언 남들이 접근 못함

Property decorator 숨겨진 변수를 반환하게 해줌

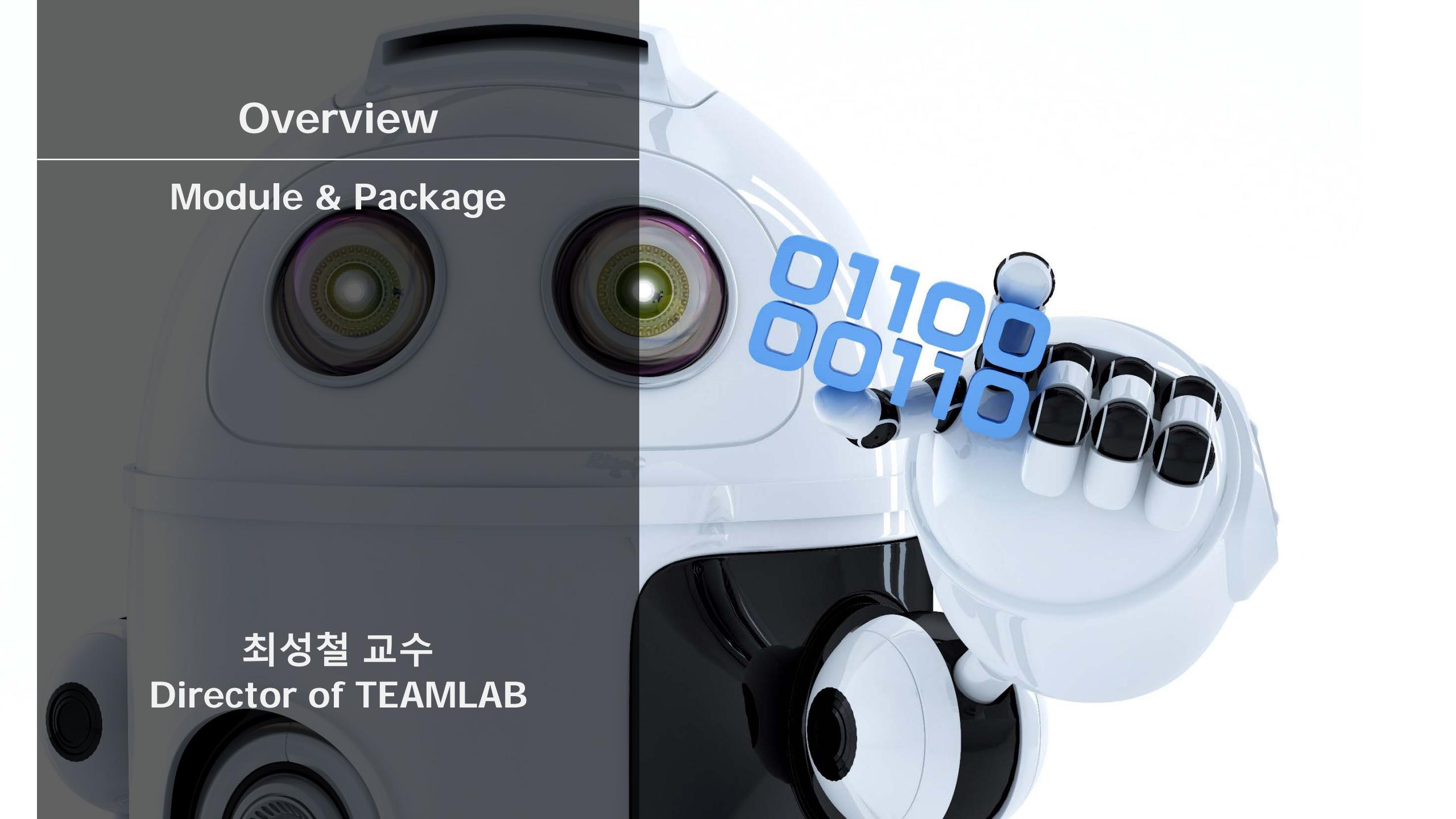
```
@property  
def items(self):  
    return self.__items
```

```
my_inventory.items
```

Property decorator로 함수를 변수처럼 호출



Human knowledge belongs to the world.

The background features a white humanoid robot with a smooth, rounded body. It has two large, circular eyes on its forehead and a small, thin mouth at the bottom. On its chest, there are large, three-dimensional blue binary digits: '01100' stacked vertically on top of '00110'. The robot's hands are visible, showing five fingers each.

Overview

Module & Package

최성철 교수

Director of TEAMLAB

PYTHON

JAVA



파이썬은 대부분의 라이브러리가 이미
다른 사용자에 의해서 구현되어 있음

**그런데 어떻게
쓰나요?**

남이 만든 프로그램 쓰는 법

- ① 함수
- ② 객체
- ③ 모듈

모듈과 패키지

모듈 (Module)

- 어떤 대상의 부분 혹은 조각

예) 레고 블록, 벽돌, 자동차 부품들



<https://goo.gl/tlv3UO>

모듈 (Module)

- 프로그램에서는 작은 프로그램 조각들,
모듈들을 모아서 하나의 큰 프로그램을 개발함
- 프로그램을 모듈화 시키면 다른 프로그램이 사용하기 쉬움
예) 카카오톡 게임을 위한 카카오톡 접속 모듈

모듈 (Module)

The screenshot shows the Kakao Developers website with the following details:

- Header:** Includes the Kakao Developers logo, navigation links for "API 소개", "개발가이드", "개발자포럼", a search bar, and a "로그인" button.
- Left Sidebar (개발가이드):** A sidebar menu with sections for iOS, Android, and various services like KakaoTalk, KakaoLink, and Push Notifications.
- Right Content Area (시작하기):**
 - Section Header:** "시작하기" with a "URL" link.
 - Text:** An introduction about the Kakaoplatform services and the need for the Kakao SDK.
 - Text:** A note that this document covers basic development environment setup and Kakaoplatform usage, not just the Kakaoplatform API.
 - Text:** A statement that the provided functions are described in detail.
 - List:** A bulleted list of steps: "개발환경 구성", "샘플앱 실행", "앱 생성", and "문제해결".
 - Text:** A note that the provided functions are described in detail.
 - List:** A bulleted list of services: "사용자 관리", "카카오스토리", "카카오톡", "카카오링크", "푸시 알림", "앱로그 분석", and "Troubleshooting".
 - Section Header:** "Android 레퍼런스".
 - Section Header:** "JavaScript 개발가이드".
 - Section Header:** "JavaScript 레퍼런스".
 - Section Header:** "REST API 개발가이드".
 - Section Header:** "소셜 플러그인".
 - Section Header:** "REST API 도구".
 - Section Header:** "Cache Tool".
 - Section Header:** "SDK 다운로드".
- Bottom Note:** A note at the bottom states: "아래 문서는 다음의 환경을 기준으로 작성되었으며, 개발자의 시스템 환경에 따라 구성이 조금씩 다를 수 있습니다."

<https://developers.kakao.com/docs>

모듈 (Module) in Python

- Built-in Module인 Random을 사용,
난수를 쉽게 생성할 수 있음

```
>>> import random  
>>> random.randint(1,1000)  
315  
>>> random.randint(1,1000)  
840  
>>> random.randint(1,1000)  
780  
>>> random.randint(1,1000)  
57  
>>>
```

패키지

- 모듈을 모아놓은 단위, 하나의 프로그램

```
Speech/          패키지의 최 상위 레벨  
    __init__.py  
SignalProcessing/ 신호 처리 하위 패키지  
    __init__.py  
    LPC.py  
    Cepstrum.py  
    FFT.py  
    FilterBank.py  
Recognition/      음성 인식 하위 패키지  
    __init__.py  
    HMM.py  
    NN.py  
    DTW.py  
Synthesis/        음성 합성 하위 패키지  
    __init__.py  
    Tagging.py  
    ProsodyControl.py  
    DBAccess.py
```

Source: 파이썬 3바이블, 이강성

직접 구현을 해봐야
알 수 있음



Human knowledge belongs to the world.



A white humanoid robot arm is shown from the waist up, holding a blue digital display. The display shows the binary sequence "01100110". The robot's hand is visible, and its arm is positioned towards the right side of the frame. The background is a dark, blurred image of the same robot.

Module

Module & Package

최성철 교수
Director of TEAMLAB

Module 만들기

- 파이썬의 Module == py 파일을 의미
- 같은 폴더에 Module에 해당하는 .py 파일과 사용하는 .py을 저장한 후
- import 문을 사용해서 Module을 호출

Module 만들기

fah_converter.py

```
def convert_c_to_f(celcius_value):
    return celcius_value * 9.0 / 5 + 32
```

module_ex.py

```
import fah_converter

print ("Enter a celsius value: "),
celsius = float(input())
fahrenheit = fah_converter.convert_c_to_f(celsius)
print ("That's ", fahrenheit, " degrees Fahrenheit")
```

Namespace

- 모듈을 호출할 때 범위 정하는 방법
- 모듈 안에는 함수와 클래스 등이 존재 가능
- 필요한 내용만 골라서 호출 할 수 있음
- `from` 과 `import` 키워드를 사용함

Namespace 사용 예시

Alias 설정하기 – 모듈명을 별칭으로 써서

```
import fah_converter as fah      fah_converter를 fah라는 이름으로  
print(fah.convert_c_to_f(41.6))    그 안에 convert_c_to_f 함수를 쓴다
```

모듈에서 특정 함수 또는 클래스만 호출하기

```
from fah_converter import convert_c_to_f  
print(convert_c_to_f(41.6))      convert_c_to_f 함수만 호출함
```

모듈에서 모든 함수 또는 클래스를 호출하기

```
from fah_converter import *  
print(convert_c_to_f(41.6))      전체 호출
```

Built-in Modules

- 파이썬이 기본 제공하는 라이브러리
- 문자처리, 웹, 수학 등 다양한 모듈이 제공됨
- 별다른 조치없이 import 문으로 활용 가능

Built-in Modules Examples

#난수

```
import random
print (random.randint (0,100)) # 0~100사이의 정수 난수를 생성
print (random.random()) # 일반적인 난수 생성
```

#시간

```
import time
print(time.localtime()) # 현재 시간 출력
```

#웹

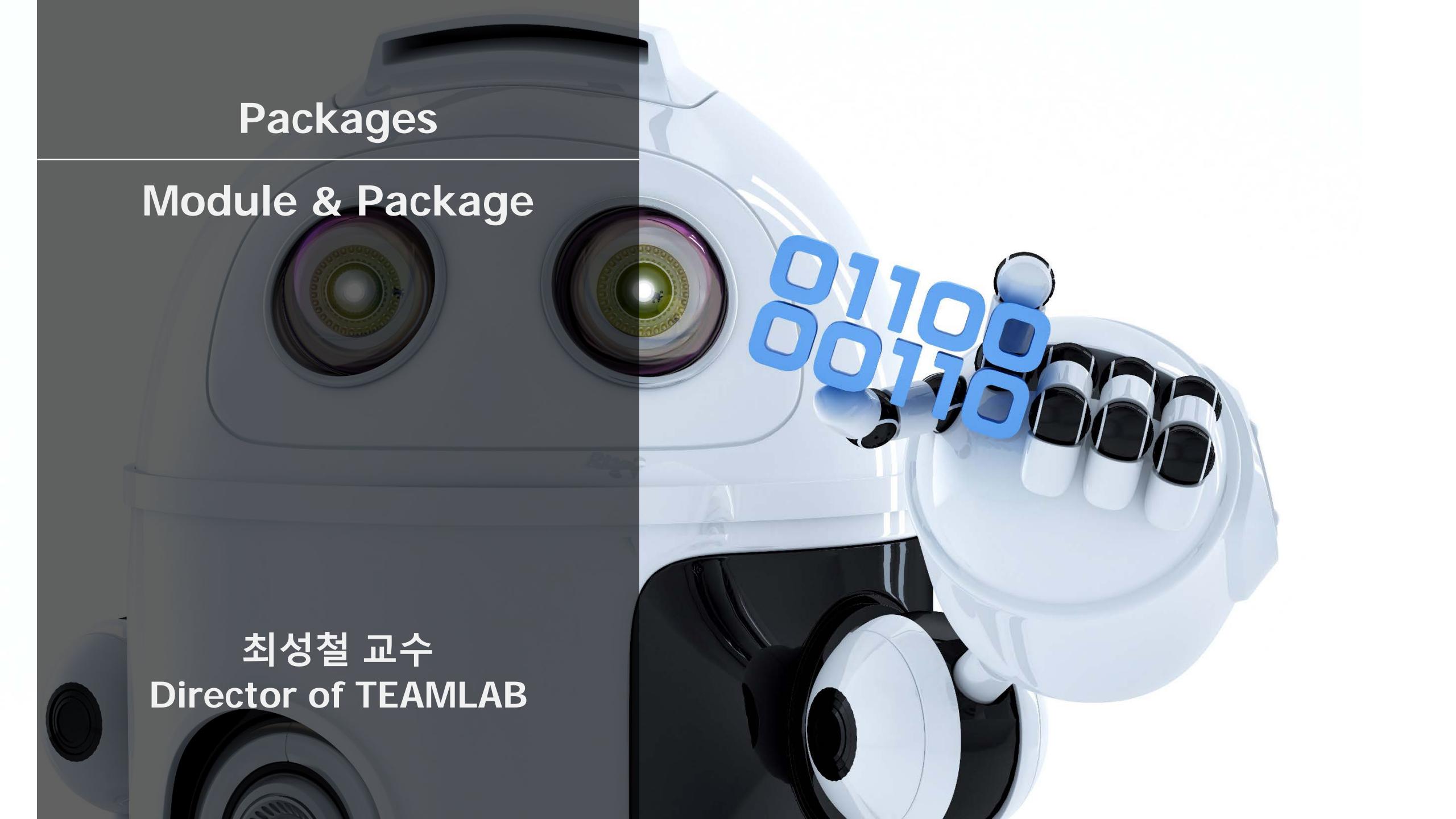
```
import urllib.request
response = urllib.request.urlopen("http://cs50.gachon.ac.kr")
print(response.read())
```

Built-in Modules

- 수 많은 파이썬 모듈은 어떻게 검색할 것인가?
 - 1) 구글신에게 물어본다
 - 2) 모듈을 import 후 구글신 검색 또는 Help 쓰기
 - 3) 공식 문서를 읽어본다 <https://docs.python.org/3/library/>
- 실습: 1부터 100까지 특정 난수를 뽑고 싶다!



Human knowledge belongs to the world.

The background features a white humanoid robot with large, expressive eyes and a visible mechanical arm. The robot's hand is held up, displaying the binary code "01100110" in blue. The robot is set against a dark, blurred background.

Packages

Module & Package

최성철 교수

Director of TEAMLAB

패키지(Package)

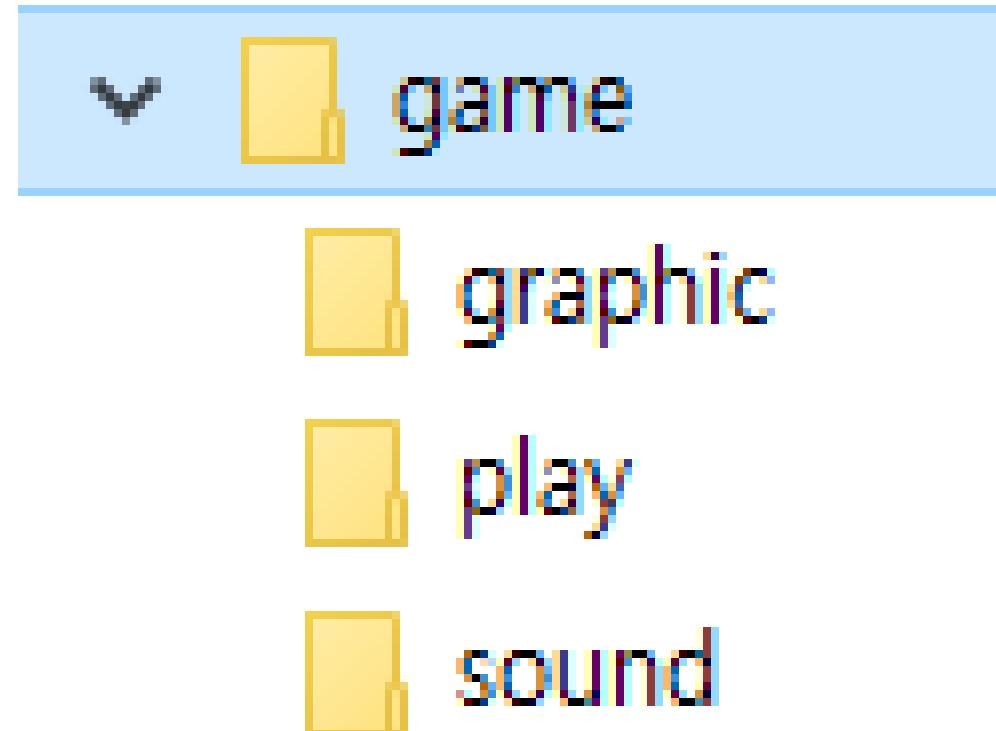
- 하나의 대형 프로젝트를 만드는 코드의 묶음
- 다양한 모듈들의 합, 폴더로 연결됨
- `__init__`, `__main__` 등 키워드 파일명이 사용됨
- 다양한 오픈 소스들이 모두 패키지로 관리됨

<https://wikidocs.net/1418>

<https://github.com/scikit-learn/scikit-learn/tree/master/sklearn>

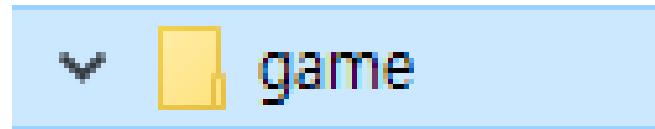
Package 만들기

1) 기능들을 세부적으로 나눠 폴더로 만듦



Package 만들기

2) 각 폴더별로 필요한 모듈을 구현함



graphic

play

sound

```
game/
    __init__.py
sound/
    __init__.py
    echo.py
    wav.py
graphic/
    __init__.py
    screen.py
    render.py
play/
    __init__.py
    run.py
    test.py
```

Package 만들기

2) 각 폴더별로 필요한 모듈을 구현함

```
# echo.py  
def echo_test():  
    print ("echo")
```

```
# render.py  
def render_test():  
    print ("render")
```

3) 1차 Test – python shell

```
>>> from game.sound.echo import echo_test  
>>> echo_test()  
echo  
>>> from game.graphic import render  
>>> from game.graphic import render as rd  
>>> rd.render_test()  
render  
>>> -
```

Package 만들기

4) 폴더별로 `_init_.py` 구성하기

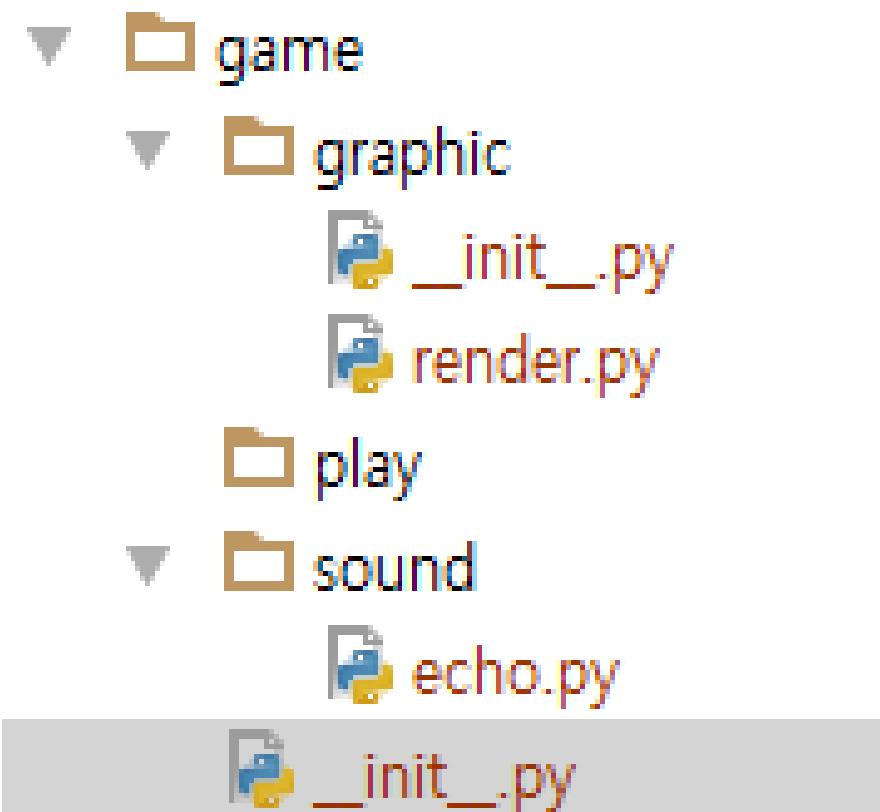
- 현재 폴더가 패키지임을 알리는 초기화 스크립트
- 없을 경우 패키지로 간주하지 않음 (3.3+ 부터는 X)
- 하위 폴더와 py 파일(모듈)을 모두 포함함
- `import`와 `_all_ keyword` 사용

Package 만들기

4) 폴더별로 __init__.py 구성하기

```
__all__=['graphic', 'play', 'sound']
```

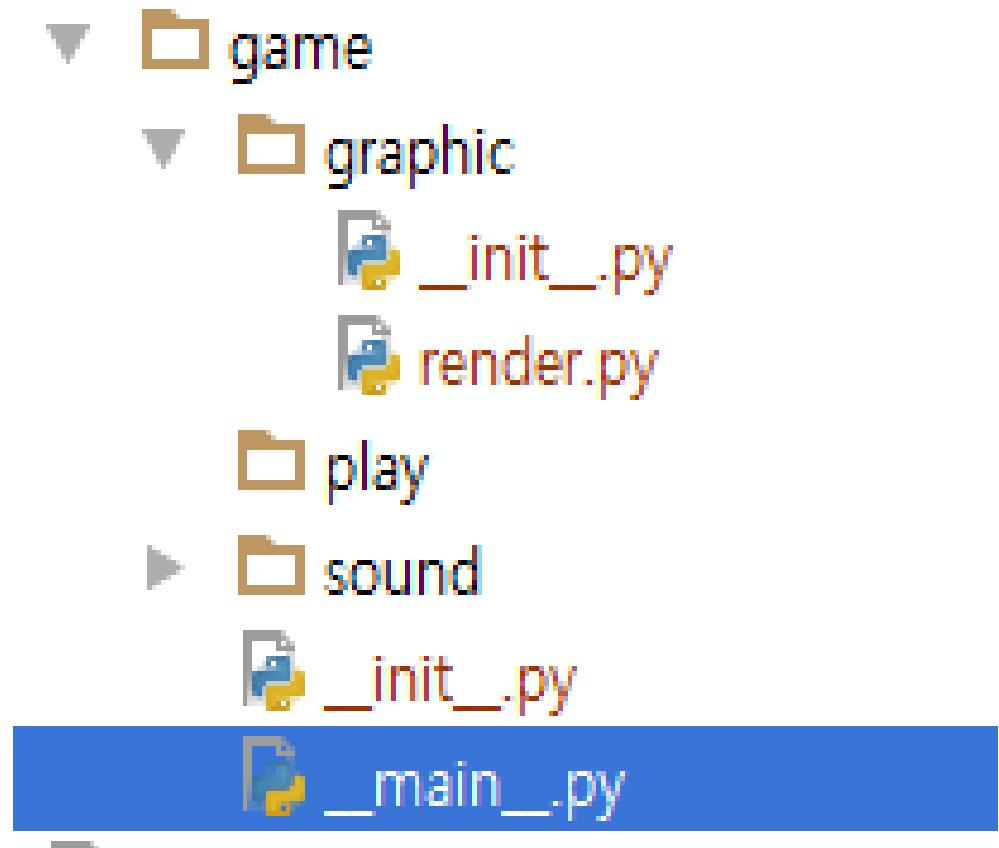
```
import graphic  
import play  
import sound
```



Package 만들기

5) __main__.py 파일 만들기

```
from graphic.render import render_test  
from sound.echo import echo_test  
  
if __name__ == '__main__':  
    render_test()  
    echo_test()
```

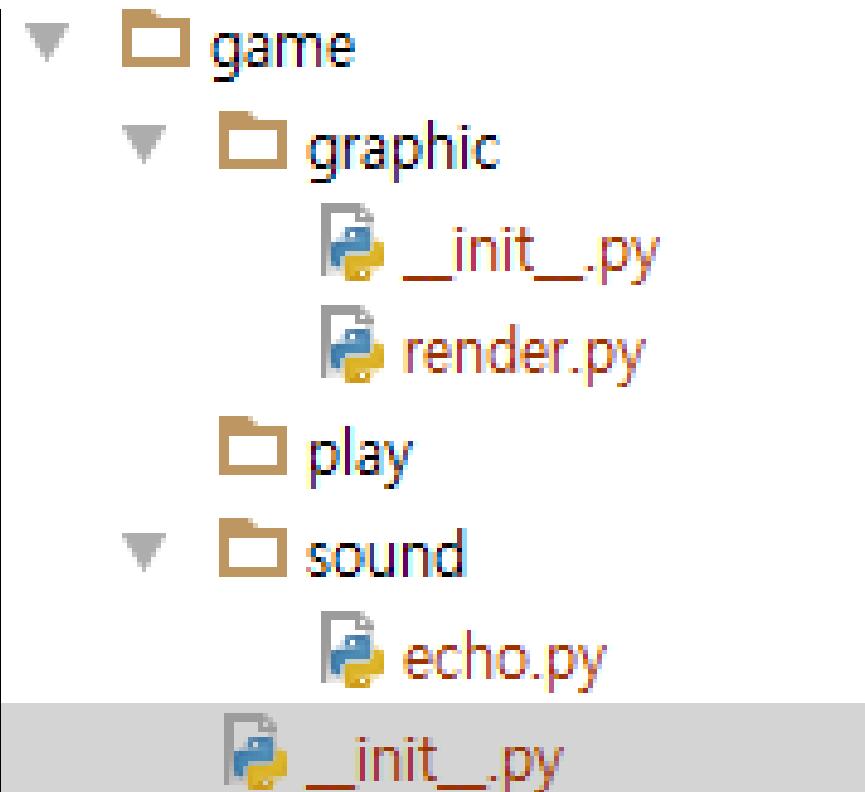


Package 만들기

5) 실행하기 – 패키지 이름만으로 호출하기

python game

```
D:\workspace\Gachon_CS50_Python_KMOOC\code\10\package>dir  
D 드라이브의 볼륨: Data  
볼륨 일련 번호: 2C24-D8FE  
  
D:\workspace\Gachon_CS50_Python_KMOOC\code\10\package 디렉터리  
  
2016-10-28 오후 12:37 <DIR> .  
2016-10-28 오후 12:37 <DIR> ..  
2016-10-28 오후 01:06 <DIR> game  
    0개 파일          0 바이트  
    3개 디렉터리  347,535,712,256 바이트 남음  
  
D:\workspace\Gachon_CS50_Python_KMOOC\code\10\package>python game  
render  
echo  
  
D:\workspace\Gachon_CS50_Python_KMOOC\code\10\package>
```



Package namespace

Package 내에서 다른 폴더의 모듈을 부를 때

상대 참조로 호출하는 방법

절대참조

```
from game.graphic.render import render_test()
```

. 현재 디렉토리 기준

```
from .render import render_test()
```

```
from ..sound.echo import echo_test()
```

.. 부모 디렉토리 기준



Human knowledge belongs to the world.



가상환경 사용하기

Module & Package

최성철 교수
Director of TEAMLAB

진짜 **프로젝트**를 한다
내 PC에 패키지를 설치한다.

두 개의 프로젝트

웹 과 데이터 분석

패키지는 둘다 설치?

가상환경 설정하기

Virtual Environment

가상환경 (Virtual Environment)

- 프로젝트 진행 시 필요한 패키지만 설치하는 환경
- 기본 인터프리터 + 프로젝트 종류별 패키지 설치
 - ex) 웹 프로젝트, 데이터 분석 프로젝트
각각 패키지 관리할 수 있는 기능
- 다양한 패키지 관리 도구를 사용함

가상환경 (Virtual Environment)

- 대표적인 도구 **virtualenv**와 **conda**가 있음

virtualenv

가장 **대표적인**
가상환경 관리 도구

레퍼런스+패키지 개수

conda

상용 가상환경도구
miniconda 기본 도구

설치의 용이성
Windows에서 장점

Conda 가상환경

```
conda create -n my_project python=3.4
```

가상환경 새로 만들기

가상환경 이름

파이썬 버전

```
Fetching package metadata: .....
Solving package specifications: .....
Package plan for installation in environment C:\Users\nhkim\Anaconda3\envs\my_project:

```

```
The following packages will be downloaded:
```

package	build
setuptools-27.2.0	py34_1

```
The following NEW packages will be INSTALLED:
```

```
pip:          8.1.2-py34_0
python:       3.4.5-0
setuptools:   27.2.0-py34_1
vs2010_runtime: 10.0.40219.1-2
wheel:        0.29.0-py34_0
```

```
Proceed ([y]/n)?
```

Conda 가상환경

가상환경 호출

```
activate my_project
```

```
#  
# To activate this environment, use:  
# > activate my_project  
#  
  
C:\Users\nhkim>activate my_project  
Deactivating environment "C:\Users\nhkim\Anaconda3" . . .  
Activating environment "C:\Users\nhkim\Anaconda3\envs\my_project" . . .
```

가상환경 해제

```
deactivate
```

패키지 설치

```
conda install <패키지명>
```

설치하고자 하는 패키지 명 입력

```
conda install matplotlib
```

```
The following packages will be downloaded:
```

package	build	
icu-57.1	vc10_0	34.3 MB
jpeg-8d	vc10_2	177 KB
openssl-1.0.2j	vc10_0	4.5 MB
numpy-1.11.2	py34_0	3.2 MB
pytz-2016.7	py34_0	170 KB
qt-5.6.0	vc10_0	51.1 MB
sip-4.18	py34_0	236 KB
pyqt-5.6.0	py34_0	4.2 MB
matplotlib-1.5.3	np111py34_1	6.2 MB
	Total:	104.0 MB

Windows에서는
conda

linux, mac에서는
pip

Windows에서는 컴파일된 C 라이브러리 설치 필요

```
The following NEW packages will be INSTALLED:
```

cycler:	0.10.0-py34_0
icu:	57.1-vc10_0
jpeg:	8d-vc10_2
libpng:	1.6.22-vc10_0
matplotlib:	1.5.3-np111py34_
mkl:	11.3.3-1
numpy:	1.11.2-py34_0
openssl:	1.0.2j-vc10_0
pyparsing:	2.1.4-py34_0
pyqt:	5.6.0-py34_0
python-datetime:	2.5.3-py34_0
pytz:	2016.7-py34_0
qt:	5.6.0-vc10_0
sip:	4.18-py34_0
six:	1.10.0-py34_0
tk:	8.5.18-vc10_0
zlib:	1.2.8-vc10_3

[vc10]
[vc10]
[vc10]

[vc10]

[vc10]

[vc10]

[vc10]

Conda 가상환경 예시

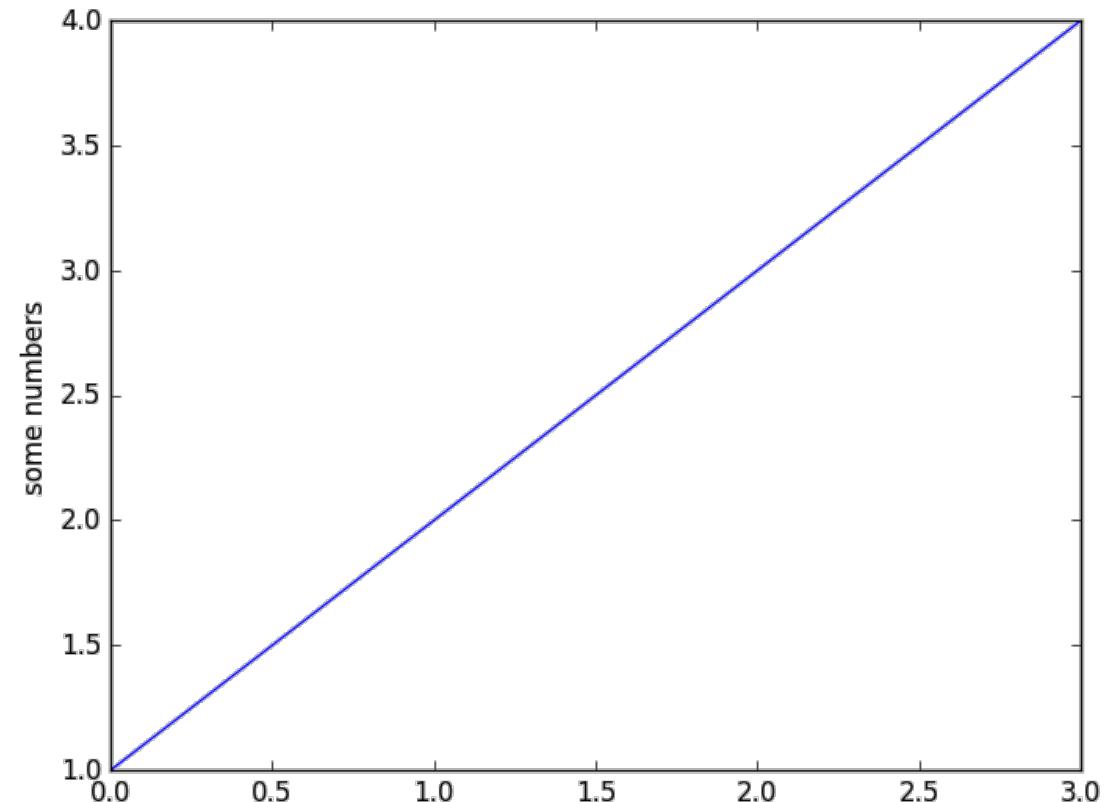
matplotlib 활용한 그래프 표시

- 대표적인 **파이썬 그래프 관리 패키지**
- 엑셀과 같은 그래프들을 화면에 표시함
- 다양한 데이터 분석 도구들과 함께 사용됨

<http://matplotlib.org/>

Conda 가상환경 예시

```
import matplotlib.pyplot as plt  
plt.plot([1,2,3,4])  
plt.ylabel('some numbers')  
plt.show()
```





Human knowledge belongs to the world.

The background of the slide features a white humanoid robot with a smooth, rounded body. It has two large, bulging eyes with yellow and green pupils, giving it a slightly surprised or curious expression. A hand is visible on the right side, holding a blue digital tablet. The screen of the tablet displays the binary code "01100110" in a large, bold, blue font.

Overview

Exception Handling

최성철 교수

Director of TEAMLAB

**프로그램 사용할 때
일어나는 흔한 일들**



<http://jino.me/1837>

[생각해보기]

프로그램 사용할 때 일어나는 오류들

- 주소를 입력하지 않고 배송 요청
- 저장도 안 했는데 컴퓨터 전원이 나감
- 게임 아이템 샀는데 게임에서 튕김

→ 예상치 못한 많은 일(예외)들이 생김

Exception

- 1) 예상 가능한 예외
- 2) 예상이 불가능한 예외

예상 가능한 예외

- 발생 여부를 사전에 인지할 수 있는 예외
- 사용자의 잘못된 입력, 파일 호출시 파일 없음
- 개발자가 반드시 명시적으로 정의 해야함

예상 불가능한 예외

- 인터프리터 과정에서 발생하는 예외, 개발자 실수
- 리스트의 범위를 넘어가는 값 호출, 정수 0으로 나눔
- 수행 불가시 인터프리터가 자동 호출

예외 처리 (Exception Handling)

- 예외가 발생할 경우 후속 조치 등 대처 필요

1) 없는 파일 호출 → 파일 없음을 알림

2) 게임 이상 종료 → 게임 정보 저장

프로그램 = 제품, 모든 잘못된상황에 대처가 필요

Exception Handling



Human knowledge belongs to the world.



Implementation

Exception Handling

최성철 교수
Director of TEAMLAB

예외 처리

Exception Handling

파이썬의 예외 처리

try ~ except 문법

try:

예외 발생 가능 코드

except <Exception Type>:

예외 발생시 대응하는 코드

파이썬의 예외 처리 예제

- 0으로 숫자를 나눌 때 예외처리 하기

```
for i in range(10):
    try:
        print(10 / i)
    except ZeroDivisionError:
        print("Not divided by 0")
```

Exception의 종류

- Built-in Exception: 기본적으로 제공하는 예외

Exception 이름	내용
IndexError	List의 Index 범위를 넘어갈 때
NameError	존재하지 않은 변수를 호출 할 때
ZeroDivisionError	0으로 숫자를 나눌 때
ValueError	변환할 수 없는 문자/숫자를 변환할 때
FileNotFoundException	존재하지 않는 파일을 호출할 때

파이썬의 예외 처리 예제

- 예외 정보 표시하기

```
for i in range(10):
    try:
        print(10 / i)
    except ZeroDivisionError as e:
        print(e)
        print("Not divided by 0")
```

else 구문

try ~ except ~ else

try:

예외 발생 가능 코드

except <Exception Type>:

예외 발생시 동작하는 코드

else:

예외가 발생하지 않을 때 동작하는 코드

else 구문 예시

try ~ except ~ else

```
for i in range(10):
    try:
        result = 10 / i
    except ZeroDivisionError:
        print("Not divided by 0")
    else:
        print(10 / i)
```

finally 구문

try ~ except ~ finally

try:

예외 발생 가능 코드

except <Exception Type>:

예외 발생시 동작하는 코드

finally:

예외 발생 여부와 상관없이 실행됨

finally 구문 예시

try ~ except ~ finally

```
try:  
    for i in range(1, 10):  
        result = 10 // i  
        print(result)  
except ZeroDivisionError:  
    print("Not divided by 0")  
finally:  
    print("종료되었습니다.")
```

raise 구문

필요에 따라 강제로 Exception을 발생

```
raise <Exception Type>(예외정보)
```

```
while True:  
    value = input("변환할 정수 값을 입력해주세요")  
    for digit in value:  
        if digit not in "0123456789":  
            raise ValueError("숫자값을 입력하지 않으셨습니다")  
    print("정수값으로 변환된 숫자 -", int(value))
```

Assert 구문

특정 조건에 만족하지 않을 경우 예외 발생

assert 예외조건

```
def get_binary_nmubmer(decimal_number):
    assert isinstance(decimal_number, int)
    return bin(decimal_number)

print(get_binary_nmubmer(10))
```



Human knowledge belongs to the world.

Overview

File

최성철 교수
Director of TEAMLAB

01100
00110

**컴퓨터를 실행할 때
가장 기본이 되는 단위**

컴퓨터를 실행할 때
가장 기본이 되는 단위

파일

[생각해보기]

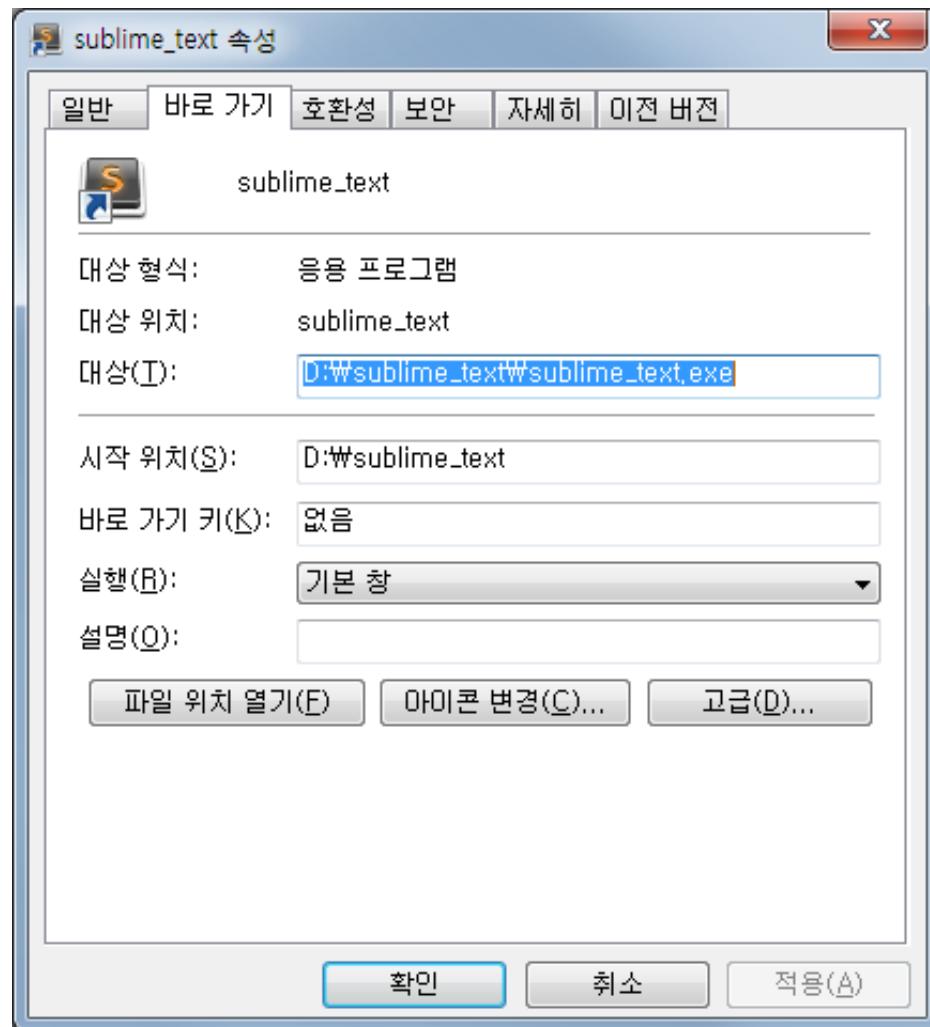
우리는 어떻게 프로그램을 시작하나?



보통은 이렇게 생긴 아이콘을 누른다!

그러나 실제로는
아이콘이 아닌 “실행 파일”을 실행시키는 것
아이콘을 클릭하고 오른쪽 마우스 클릭 “속성”을 선택해 볼 것

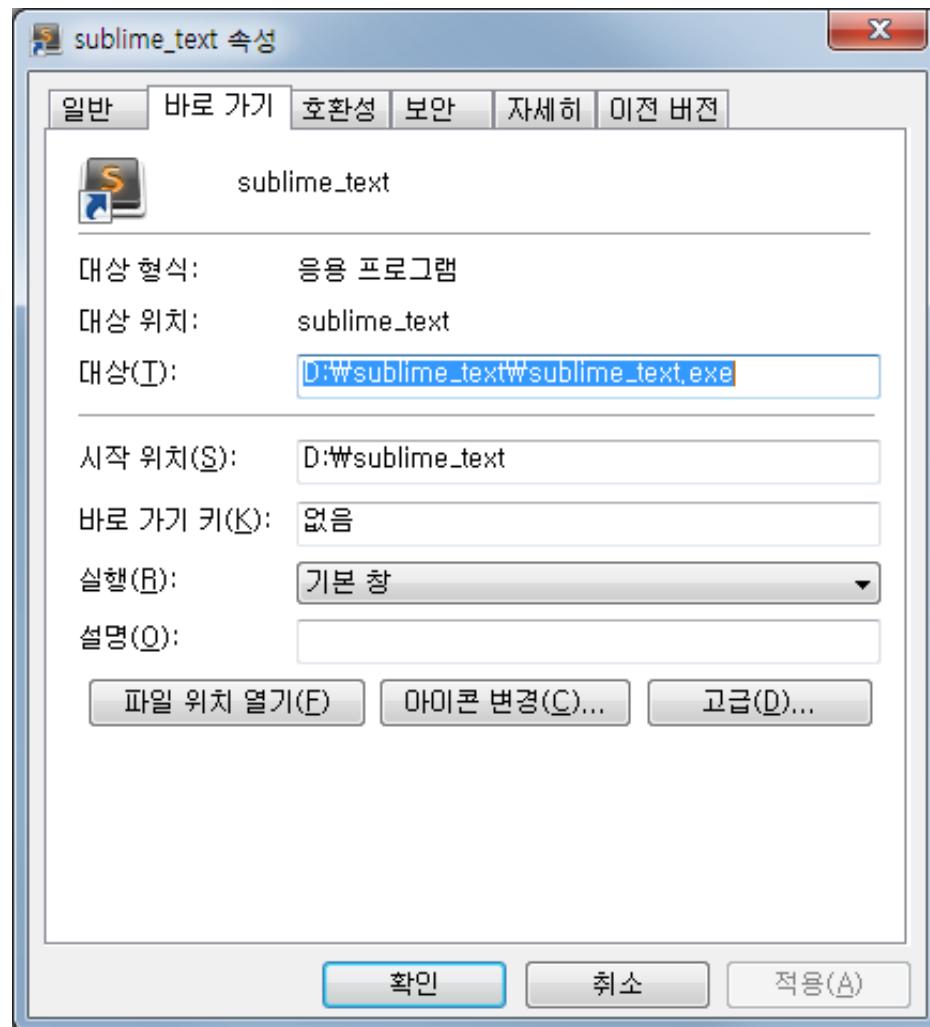
[생각해보기]



실행 시켜 보기

- 1) 대상에 있는 **text**전체를 복사
- 2) **Windows key + r**을 누르고
- 3) **cmd** 를 입력 후 엔터를 칠 것
- 4) 오른쪽 마우스를 클릭
- 5) “붙여넣기” 메뉴선택 한 후
- 6) 텍스트를 입력한 후 Enter
- 7) 해당 프로그램이 실행됨

[생각해보기]

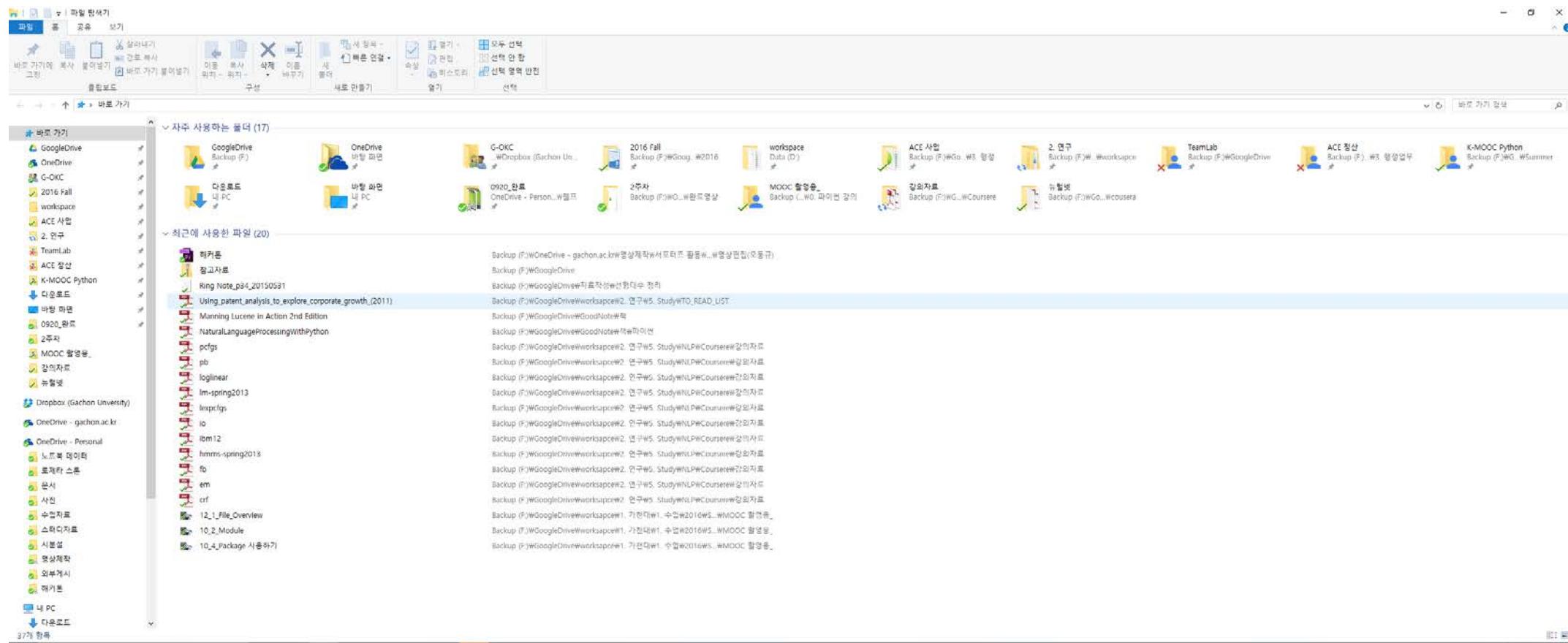


실행 시켜 보기

- 1) 대상에 있는 **text** 전체를
- 2) **Windows key** + **S**를 누르는 것은 실제로는
- 3) cmd 를 실행하는 것
- 4) 아이콘을 누르는 것은 명령하는 것
- 5) 파일의 실행을 클릭
- 6) “파일” 메뉴선택 한 후
“실행” 메뉴선택한 후
“작업 스트리밍” 키워드를 입력한 후 Enter
- 7) 해당 프로그램이 실행됨

파일의 이해

파일은 파일을 담고 있는 디렉토리와 파일로 나눌 수 있음



파일의 이해

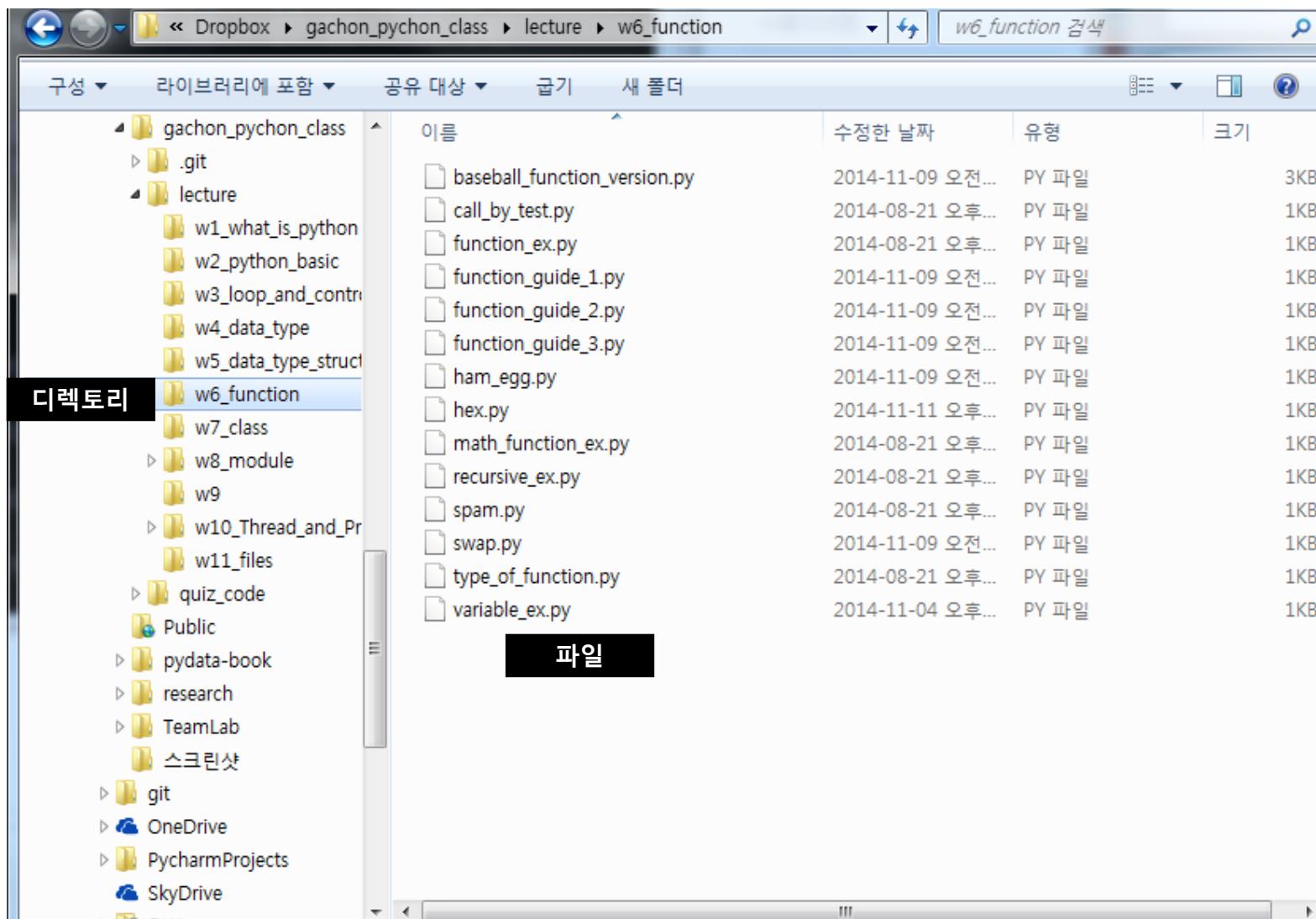
디렉토리 (Directory)

- 폴더 또는 디렉토리로 불림
- 파일과 다른 디렉토리를 포함할 수 있음

파일 (File)

- 컴퓨터에서 정보를 저장하는 논리적인 단위 ([wikipedia](#))
- 파일은 파일명과 확장자로 식별됨 (예: hello.py)
- 실행, 쓰기, 읽기 등을 할 수 있음

파일의 구조(Windows)



파일의 구조(Windows)

```
./lecture/w10_Thread_and_Process: 디렉토리
합계 52
-rw-r--r-- 1 root root 1486 2014-11-09 03:28 BubbleSort.class
-rw-r--r-- 1 root root 1313 2014-11-09 03:28 BubbleSort.java
drwxr-xr-x 2 root root 4096 2014-11-12 18:06 bubble_sort_comparison 파일
-rw-r--r-- 1 root root 642 2014-11-09 03:28 bubble_sort_gevent.py
-rw-r--r-- 1 root root 549 2014-11-09 03:28 bubble_sort_non_thread.py
-rw-r--r-- 1 root root 768 2014-11-09 03:28 bubble_sort_process.py
-rw-r--r-- 1 root root 1678 2014-11-09 03:28 bubble_sort_process.pyc
-rw-r--r-- 1 root root 636 2014-11-09 03:28 bubble_sort_thread.py
-rw-r--r-- 1 root root 319 2014-11-09 03:28 process_basic.ex.py
-rw-r--r-- 1 root root 781 2014-11-09 03:28 process_basic.ex.pyc
-rw-r--r-- 1 root root 270 2014-11-09 03:28 thread_baisc_ex.py
-rw-r--r-- 1 root root 442 2014-11-09 03:28 thread_baisc_ex_2.py
-rw-r--r-- 1 root root 130 2014-11-09 03:28 thread_basic_code.py

./lecture/w10_Thread_and_Process/bubble_sort_comparison:
합계 8
-rw-r--r-- 1 root root 585 2014-11-09 03:28 code_a.py
-rw-r--r-- 1 root root 579 2014-11-09 03:28 code_b.py

./lecture/w11_files:
합계 0
```

파일의 종류

Binary 파일	Text 파일
<ul style="list-style-type: none">- 컴퓨터만 이해할 수 있는 형태인 이진(법)형식으로 저장된 파일- 일반적으로 메모장으로 열면 내용이 깨져 보임 (메모장 해설 불가)- 엑셀파일, 워드 파일 등등	<ul style="list-style-type: none">- 인간도 이해할 수 있는 형태인 문자열 형식으로 저장된 파일- 메모장으로 열면 내용 확인 가능- 메모장에 저장된 파일, HTML 파일, 파이썬 코드 파일 등

- 컴퓨터는 Text 파일을 처리하기 위해 Binary 파일로 변환시킴 (예: pyc 파일)
- 모든 Text 파일도 실제는 Binary 파일, ASCII/Unicode 문자열 집합으로 저장되어 사람이 읽을 수 있음

파일의 종류

ASCII CODE TABLE

10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자
0	0x00	NULL	22	0x16	STN	44	0x2C	,	66	0x42	B	88	0x58	X	110	0x6E	n
1	0x01	SOH	23	0x17	ETB	45	0x2D	-	67	0x43	C	89	0x59	Y	111	0x6F	o
2	0x02	STX	24	0x18	CAN	46	0x2E	.	68	0x44	D	90	0x5A	Z	112	0x70	p
3	0x03	ETX	25	0x19	EM	47	0x2F	/	69	0x45	E	91	0x5B	[113	0x71	q
4	0x04	EOT	26	0x1A	SUB	48	0x30	0	70	0x46	F	92	0x5C	₩	114	0x72	r
5	0x05	ENQ	27	0x1B	ESC	49	0x31	1	71	0x47	G	93	0x5D]	115	0x73	s
6	0x06	ACK	28	0x1C	FS	50	0x32	2	72	0x48	H	94	0x5E	^	116	0x74	t
7	0x07	BEL	29	0x1D	GS	51	0x33	3	73	0x49	I	95	0x5F	-	117	0x75	u
8	0x08	BS	30	0x1E	RS	52	0x34	4	74	0x4A	J	96	0x60	`	118	0x76	v
9	0x09	HT	31	0x1F	US	53	0x35	5	75	0x4B	K	97	0x61	a	119	0x77	w
10	0x0A	₩n	32	0x20	SP	54	0x36	6	76	0x4C	L	98	0x62	b	120	0x78	x
11	0x0B	VT	33	0x21	!	55	0x37	7	77	0x4D	M	99	0x63	c	121	0x79	y
12	0x0C	FF	34	0x22	"	56	0x38	8	78	0x4E	N	100	0x64	d	122	0x7A	z
13	0x0D	₩r	35	0x23	#	57	0x39	9	79	0x4F	O	101	0x65	e	123	0x7B	{
14	0x0E	SO	36	0x24	\$	58	0x3A	:	80	0x50	P	102	0x66	f	124	0x7C	
15	0x0F	SI	37	0x25	%	59	0x3B	;	81	0x51	Q	103	0x67	g	125	0x7D	}
16	0x10	DLE	38	0x26	&	60	0x3C	<	82	0x52	R	104	0x68	h	126	0x7E	~
17	0x11	DC1	39	0x27	'	61	0x3D	=	83	0x53	S	105	0x69	i	127	0x7F	DEL
18	0x12	DC2	40	0x28	(62	0x3E	>	84	0x54	T	106	0x6A	j			
19	0x13	DC3	41	0x29)	63	0x3F	?	85	0x55	U	107	0x6B	k			
20	0x14	DC4	42	0x2A	*	64	0x40	@	86	0x56	V	108	0x6C	l			
21	0x15	NAK	43	0x2B	+	65	0x41	A	87	0x57	W	109	0x6D	m			

출처:
<http://sexy.pe.kr>



Human knowledge belongs to the world.

File Handling

File

최성철 교수
Director of TEAMLAB

01100
00110

파이썬의 File I/O

파이썬은 파일 처리를 위해 “open” 키워드를 사용함

```
f = open("<파일이름>", "접근 모드")  
f.close()
```

파일열기모드	설명
r	읽기모드 - 파일을 읽기만 할 때 사용
w	쓰기모드 - 파일에 내용을 쓸 때 사용
a	추가모드 - 파일의 마지막에 새로운 내용을 추가 시킬 때 사용

파일 읽기

파이썬의 File Read

f.read() txt 파일 안에 있는 내용을 문자열로 반환

```
f = open("i_have_a_dream.txt", "r")
contents = f.read()
print(contents)
f.close()
```

대상파일이 같은 폴더에 있을 경우

with 구문과 함께 사용하기

```
with open("i_have_a_dream.txt", "r") as my_file:
    contents = my_file.read()
    print(type(contents), contents)
```

파이썬의 File Read

한 줄씩 읽어 List Type으로 반환함

```
with open("i_have_a_dream.txt", "r") as my_file:  
    content_list = my_file.readlines() #파일 전체를 list로 반환  
    print(type(content_list)) #Type 확인  
    print(content_list) #리스트 값 출력
```

파이썬의 File Read

실행 시마다 한 줄씩 읽어오기

```
with open("i_have_a_dream.txt", "r") as my_file:  
    i = 0  
    while 1:  
        line = my_file.readline().replace("\n", "")  
        if line.strip() == "":  
            continue  
        if not line: break  
        print(str(i) + " == " + line) #한줄씩 값 출력  
        i = i + 1
```

파이썬의 File Read

단어 통계 정보 산출

```
with open("i_have_a_dream.txt", "r") as my_file:  
    contents = my_file.read()  
    word_list = contents.split(" ")  
    #빈칸 기준으로 단어를 분리 리스트  
    line_list = contents.split("\n")  
    #한줄 씩 분리하여 리스트  
  
    print("Total Number of Characters :", len(contents))  
    print("Total Number of Words:", len(word_list))  
    print("Total Number of Lines :", len(line_list))
```

파일 쓰기

파이썬의 File Write

mode는 "w", encoding="utf8"

```
f = open("count_log.txt", 'w', encoding="utf8")
for i in range(1, 11):
    data = "%d번째 줄입니다.\n" % i
    f.write(data)
f.close()
```

mode는 "a"는 추가 모드

```
with open("count_log.txt", 'a', encoding="utf8") as f:
    for i in range(1, 11):
        data = "%d번째 줄입니다.\n" % i
        f.write(data)
```

파이썬의 Directory 만들기

os 모듈을 사용하여 Directory 다루기

```
import os  
os.mkdir("log")
```

디렉토리가 있는지 확인하기

```
if not os.path.isdir("log"):  
    os.mkdir("log")
```

Log 파일 생성하기

1) 디렉토리가 있는지, 2) 파일이 있는지 확인 후

```
import os
if not os.path.isdir("log"):
    os.mkdir("log")
if not os.path.exists("log/count_log.txt"):
    f = open("log/count_log.txt", 'w', encoding="utf8")
    f.write("기록이 시작됩니다\n")
    f.close()

with open("log/count_log.txt", 'a', encoding="utf8") as f:
    import random, datetime
    for i in range(1, 11):
        stamp = str(datetime.datetime.now())
        value = random.random() * 1000000
        log_line = stamp + " " + str(value) + " 값이 생성되었습니다" + "\n"
        f.write(log_line)
```

python pickle

Pickle

- 파이썬의 객체를 영속화(persistence)하는 built-in 객체
- 데이터, object 등 실행중 정보를 저장 → 불러와서 사용
- 저장해야하는 정보, 계산 결과(모델) 등 활용이 많음

```
import pickle

f = open("list.pickle", "wb")      f = open("list.pickle", "rb")
test = [1, 2, 3, 4, 5]           test_pickle = pickle.load(f)
pickle.dump(test, f)              print(test_pickle)
f.close()                         f.close()
```

Pickle

```
import pickle

class Mutltiply(object):
    def __init__(self, multiplier):
        self.multiplier = multiplier

    def multiply(self, number):
        return number * self.multiplier

mulpily = Mutltiply(5)
mulpily.multiply(10)

f = open("multiply_object.pickle", "wb")
pickle.dump(mulpily, f)
f.close()

f = open("multiply_object.pickle", "rb")
multiply_pickle = pickle.load(f)
multiply_pickle.multiply(5)
```



Human knowledge belongs to the world.

Logging Overview

Logging

최성철 교수
Director of TEAMLAB

01100
00110

게임을 만들었는데
HACK쓰는 애들 때문에
망했어요...

어떻게 잡을 수 있을까?

일단은 기록부터!!

로그 남기기 - Logging

- 프로그램이 실행되는 동안 일어나는 정보를 기록을 남기기
- 유저의 접근, 프로그램의 Exception, 특정 함수의 사용
- Console 화면에 출력, 파일에 남기기, DB에 남기기 등등
- 기록된 로그를 분석하여 의미있는 결과를 도출 할 수 있음
- 실행시점에서 남겨야 하는 기록, 개발시점에서 남겨야하는 기록

print vs logging

- 기록을 print로 남기는 것도 가능함
- 그러나 Console 창에만 남기는 기록은 분석시 사용할 수가 없음
- 때로는 레벨별(개발, 운영)로 기록을 남길 필요도 있음
- 모듈별로 별도의 logging을 남길 필요도 있음
- 이러한 기능을 체계적으로 지원하는 모듈이 필요함

logging 모듈

- Python의 기본 Log 관리 모듈

```
import logging

logging.debug("틀렸잖아!")
logging.info("확인해")
logging.warning("조심해!")
logging.error("에러났어!!!")
logging.critical ("망했다...")
```

logging level

- 프로그램 진행 상황에 따라 다른 Level의 Log를 출력함
- 개발 시점, 운영 시점 마다 다른 Log가 남을 수 있도록 지원함
- **DEBUG > INFO > WARNING > ERROR > Critical**
- Log 관리시 가장 기본이 되는 설정 정보

logging level

<https://stackoverflow.com/questions/2031163/when-to-use-the-different-log-levels>

Level	개요	예시
debug	개발시 처리 기록을 남겨야하는 로그 정보를 남김	<ul style="list-style-type: none">- 다음 함수로 A 를 호출함- 변수 A 를 무엇으로 변경함
info	처리가 진행되는 동안의 정보를 알림	<ul style="list-style-type: none">- 서버가 시작되었음- 서버가 종료됨- 사용자 A가 프로그램에 접속함
warning	사용자가 잘못 입력한 정보나 처리는 가능하나 원래 개발시 의도치 않는 정보가 들어왔을 때 알림	<ul style="list-style-type: none">- Str입력을 기대했으나, Int가 입력됨 → Str casting으로 처리함- 함수에 argument로 이차원 리스트를 기대했으나 → 일차원 리스트가 들어옴, 이차원으로 변환후 처리
error	잘못된 처리로 인해 에러가 났으나, 프로그램은 동작 할 수 있음을 알림	<ul style="list-style-type: none">- 파일에 기록을 해야하는데 파일이 없음 --> Exception 처리후 사용자에게 알림- 외부서비스와 연결 불가
critical	잘못된 처리로 데이터 손실이나 더이상 프로그램이 동작할 수 없음을 알림	<ul style="list-style-type: none">- 잘못된 접근으로 해당 파일이 삭제됨- 사용자의 의한 강제 종료

logging level

```
import logging
```

```
logger = logging.getLogger("main")
```

Logger 선언

```
stream_hander = logging.StreamHandler()
```

Logger의 output 방법 선언

```
logger.addHandler(stream_hander)
```

Logger의 output 등록

```
logger.setLevel(logging.DEBUG)
```

```
logger.setLevel(logging.CRITICAL)
```

```
logger.debug("틀렸잖아!")
```

```
logger.debug("틀렸잖아!")
```

```
logger.info("확인해")
```

```
logger.info("확인해")
```

```
logger.warning("조심해!")
```

```
logger.warning("조심해!")
```

```
logger.error("에러났어!!!")
```

```
logger.error("에러났어!!!")
```

```
logger.critical("망했다...")
```

```
logger.critical("망했다...")
```



Human knowledge belongs to the world.

Comma Separate Value

File

가천대학교 | 산업경영공학과
최성철 교수

01100
00110



Comma Separate Value

- CSV, 필드를 쉼표(,)로 구분한 텍스트 파일
- 엑셀 양식의 데이터를 프로그램에 상관없이 쓰기 위한 데이터 형식이라고 생각하면 쉬움
- 탭(TSV), 빈칸(SSV) 등으로 구분해서 만들기도 함
- 통칭하여 character-separated values (CSV) 부름
- 엑셀에서는 “다음 이름 저장” 기능으로 사용 가능

엑셀로 CSV 파일 만들기

- 1) 파일 다운로드 - <https://goo.gl/lQvsQF>
- 2) 파일 열기
- 3) 파일 → 다른 이름으로 저장
→ CSV(쉼표로 분리) 선택 후 → 파일명 입력
- 4) 엑셀 종료 후 Notepad로 파일 열어보기

엑셀로 CSV 파일 만들기

데이터가 “,”로 나눠져 있음

조사번호	조사지역	조사일자	시간대	X좌표	Y좌표	행정구역명	날씨	남자10대	남자20대	남자30대	남자40대	남자50대	여자10대	여자20대	여자30대	여자40대	여자50대
2544	신촌네거리	서울특별시	2010-06-21	12시~13시까지	343099	417482	대전광역시 서구	울릉동,한글을,2,24,68,50,31,4,37,64,44,26									
2544	신촌네거리	서울특별시	2010-06-21	19시~20시까지	343099	417482	대전광역시 서구	울릉동,한글을,19,44,28,33,21,14,56,49,43,18									
2544	신촌네거리	서울특별시	2010-06-20	12시~13시까지	343099	417482	대전광역시 서구	울릉동,한글을,13,33,34,61,55,13,32,29,28,12									
2544	신촌네거리	서울특별시	2010-06-20	19시~20시까지	343099	417482	대전광역시 서구	울릉동,한글을,23,33,32,547,129,12,39,13,46,4									
2545	계룡로에서	신촌네거리	2010-06-21	12시~13시까지	343121	417343	대전광역시 서구	울릉동,한글을,0,9,27,21,6,5,24,20,10,6									
2545	계룡로에서	신촌네거리	2010-06-21	19시~20시까지	343121	417343	대전광역시 서구	울릉동,한글을,4,22,12,22,12,6,30,27,7,3									
2545	계룡로에서	신촌네거리	2010-06-20	12시~13시까지	343121	417343	대전광역시 서구	울릉동,한글을,3,8,5,8,11,2,8,9,4,3									
2545	계룡로에서	신촌네거리	2010-06-20	19시~20시까지	343121	417343	대전광역시 서구	울릉동,한글을,6,17,19,21,27,5,16,18,11,11									
2546	파장시장입구	주말	2010-06-04	12시~13시까지	311013	523508	경기도 수원시	정자1동,한글을,24,26,32,45,60,21,28,98,195,201									
2546	파장시장입구	주말	2010-06-04	19시~20시까지	311013	523508	경기도 수원시	정자1동,한글을,13,36,18,20시,90,48,36,36,114,126									
2546	파장시장입구	주말	2010-06-05	12시~13시까지	311013	523508	경기도 수원시	정자1동,한글을,14,25,45,36,52,9,32,42,36,34									
2546	파장시장입구	주말	2010-06-05	19시~20시까지	311013	523508	경기도 수원시	정자1동,한글을,36,42,89,78,64,32,45,95,78,42									
2547	갈마네거리	북쪽	2010-06-24	12시~13시까지	343529	417164	대전광역시 서구	갈마동,한글을,6,10,12,12,18,0,10,29,34,46									
2547	갈마네거리	북쪽	2010-06-24	19시~20시까지	343529	417164	대전광역시 서구	갈마동,한글을,30,19,11,19,26,34,39,32,36,29									
2547	갈마네거리	북쪽	2010-06-26	12시~13시까지	343529	417164	대전광역시 서구	갈마동,한글을,11,7,7,9,12,9,18,14,22,30									
2547	갈마네거리	북쪽	2010-06-26	19시~20시까지	343529	417164	대전광역시 서구	갈마동,한글을,9,27,19,26,35,21,21,22,30,15									
2549	갈마네거리	남쪽	2010-06-24	12시~13시까지	343463	417150	대전광역시 서구	갈마동,한글을,3,8,7,9,18,2,18,32,25,29									
2549	갈마네거리	남쪽	2010-06-24	19시~20시까지	343463	417150	대전광역시 서구	갈마동,한글을,23,25,18,24,50,16,49,53,41,28									
2549	갈마네거리	남쪽	2010-06-26	12시~13시까지	343463	417150	대전광역시 서구	갈마동,한글을,3,7,12,19,26,13,25,23,23,14									
2549	갈마네거리	남쪽	2010-06-26	19시~20시까지	343463	417150	대전광역시 서구	갈마동,한글을,3,7,14,27,28,9,23,20,18,6									
2550	한마음병원	미니스토어	2010-06-27	12시~13시까지	344808	416386	대전광역시 서구	탄방동,한글을,8,8,8,5,9,12,14,7,10,2									
2550	한마음병원	미니스토어	2010-06-27	19시~20시까지	344808	416386	대전광역시 서구	탄방동,한글을,8,14,18,5,7,6,33,17,12,14									
2551	개나리마파트	후문지점	주말	2010-06-27	12시~13시까지	344797	416293	대전광역시 서구	탄방동,한글을,4,5,8,6,5,0,11,7,4,9								
2551	개나리마파트	후문지점	주말	2010-06-27	19시~20시까지	344797	416293	대전광역시 서구	탄방동,한글을,1,19,4,5,4,0,9,8,4,6								
2552	시탄역	작곡	2010-06-28	12시~13시까지	348993	427864	대전광역시 대덕구	신봉동,한글을,30,17,11,3,15,23,11,5,9,35									
2552	시탄역	작곡	2010-06-28	19시~20시까지	348993	427864	대전광역시 대덕구	신봉동,한글을,11,26,13,14,27,41,16,15,6,35									
2552	시탄역	작곡	2010-06-26	12시~13시까지	348993	427864	대전광역시 대덕구	신봉동,한글을,13,23,54,12,31,7,35,28,42,59									
2552	시탄역	작곡	2010-06-26	19시~20시까지	348993	427864	대전광역시 대덕구	신봉동,한글을,38,58,29,48,33,27,42,34,12,8									
2553	시탄역	구축	2010-06-28	12시~13시까지	349044	427745	대전광역시 대덕구	신봉동,한글을,28,6,27,12,10,15,27,38,53,22									
2553	시탄역	구축	2010-06-28	19시~20시까지	349044	427745	대전광역시 대덕구	신봉동,한글을,23,48,30,31,12,16,54,18,11,47									
2553	시탄역	구축	2010-06-26	12시~13시까지	349044	427745	대전광역시 대덕구	신봉동,한글을,42,79,68,19,18,63,45,47,13,31									
2553	시탄역	구축	2010-06-26	19시~20시까지	349044	427745	대전광역시 대덕구	신봉동,한글을,11,27,38,38,14,42,11,12,27,6									
2554	송촌미세단련	주중	2010-06-29	12시~13시까지	349741	418133	대전광역시 대덕구	송촌동,한글을,3,8,34,14,12,1,6,8,5,13									
2554	송촌미세단련	주중	2010-06-29	19시~20시까지	349741	418133	대전광역시 대덕구	송촌동,한글을,53,18,8,50,35,58,18,7,51,30									
2554	송촌미세단련	주중	2010-06-27	12시~13시까지	349741	418133	대전광역시 대덕구	송촌동,한글을,8,5,16,33,11,6,14,7,20,3									
2554	송촌미세단련	주중	2010-06-27	19시~20시까지	349741	418133	대전광역시 대덕구	송촌동,한글을,49,25,35,17,8,40,33,10,9,18									
2555	하이마트	언니파	주중	2010-06-29	12시~13시까지	349720	418187	대전광역시 대덕구	송촌동,한글을,2,18,19,8,28,1,8,6,3,13								
2555	하이마트	언니파	주중	2010-06-29	19시~20시까지	349720	418187	대전광역시 대덕구	송촌동,한글을,63,25,21,15,17,60,22,11,8,10								
2555	하이마트	언니파	주말	2010-06-27	12시~13시까지	349720	418187	대전광역시 대덕구	송촌동,한글을,0,3,8,6,2,2,11,10,4,10								
2555	하이마트	언니파	주말	2010-06-27	19시~20시까지	349720	418187	대전광역시 대덕구	송촌동,한글을,23,4,3,8,32,8,12,5,35,44								

CSV 파일 다루기

CSV 파일 읽기/쓰기

- Text 파일 형태로 데이터 처리 예제
- 예제 데이터: customer.csv (<https://goo.gl/FIsMqB>)
- 일반적 **textfile**을 처리하듯 파일을 읽어온 후, 한 줄 한 줄씩 데이터를 처리함

<http://goo.gl/1JdKst>

CSV 파일 예제

customers	
customerNumber	INT(11)
customerName	VARCHAR(50)
contactLastName	VARCHAR(50)
contactFirstName	VARCHAR(50)
phone	VARCHAR(50)
addressLine1	VARCHAR(50)
addressLine2	VARCHAR(50)
city	VARCHAR(50)
state	VARCHAR(50)
postalCode	VARCHAR(15)
country	VARCHAR(50)
salesRepEmployeeNumber	INT(11)
creditLimit	DOUBLE
Indexes	

해당데이터는 고객의
FullName, 성, 이름, 전화번호, 국가 등 기록
""로 분리

customers.csv - 메모장	
파일(F)	편집(E)
서식(O)	보기(V)
도움말(H)	
customerNumber, customerName, contactLastName, contactFirstName, phone, addressLine1, addressLine2, city, state, postalCode, country, salesRepEmployeeNumber, creditLimit	
103, "Atelier graphique", Schmitt, "Carine", "40,32,2555,"54, rue Royale", NULL, Nantes, NULL, 44000, France, 1370, 21000	
112, "Signal Gift Stores", King, Jean, "7025551330, 8489 Strom St.", "NULL, Las Vegas, NV, 89303, USA, 1166, 71800	
114, "Australian Collectors, Co.", Ferguson, Peter, "03 9520 4555, "636 St Kilda Road", Level 3, Melbourne, Victoria, 3004, Australia, 1611, 117300	
129, "La Rochelle Gifts", Labrune, Janine, "40,67,6555, 67, rue des Quinze Otages", NULL, Nantes, NULL, 44000, France, 1370, 18200	
132, "Baamani Gifts", Bergman, Jonas, "07-98 6555, 47, Rue des Skates, apt. 1B", NULL, "S-113 51, Norway, 4100, 81700	
134, "Signal Gifts Distribution, Ltd.", Nelson, Susan, "4155551450, 5577 Strom St., NULL, "S-113 51, Norway, 4100, 81700	
125, "Havel & Zbyzsek Co.", Pleszczniowicz, "Zbyzsek", "(26) 642-7555", "ul. Filtrowa 68", NULL, Warsaw, NULL, 01-012, Poland, NULL, 0	
128, "Blauer See Auto, Co.", Keitel, Roland, "49 69 66 2655", "Lungenstr. 34", NULL, Frankfurt, NULL, 60529, Germany, 1504, 59700	
129, "Mini Wheels Co.", Murphy, Julie, "6005555787, 5557 North Pendale Street", NULL, "San Francisco, CA, 94217, USA, 1165, 64600	
131, "Land of Toys Inc.", Lee, Kwai, "2125557818, 897 Long Airport Avenue", NULL, NYC, NY, 10022, USA, 1323, 114900	
141, "Euro Shopping Channel", Freyre, Diego, "(91) 555 94 44", "C/ Moralzarzal, 86", NULL, Madrid, NULL, 28034, Spain, 1370, 227600	
144, "Volvo Model Replicas, Co.", Berslund, "Christina", "0921-12 3555", "Bergaupsvaagen 8", NULL, Luleå, NULL, "S-958 22, Sweden, 1504, 53100	
145, "Danish Wholesale Imports", Petersen, "Jytte", "31 12 3555", "Vibaustræet 34", NULL, Kobenhavn, NULL, 1734, Denmark, 1401, 83400	
146, "Saveler & Henriot, Co.", Saveler, "Mary", "78 32 5555, 2, rue du Commerce", NULL, Lyon, NULL, 69004, France, 1337, 123900	
148, "Dragon Souvenirs, Ltd.", Natividad, Eric, "+65 221 7555", "Bronz Sok.", "Bronz Apt. 3/6 Tesviklve", Singapore, NULL, 079903, Singapore, 1621, 103800	
151, "Muscle Machine Inc.", Young, Jeff, "2125557413, 4098 Furth Circle", "Suite 3/6", NYC, NY, 10022, USA, 1286, 138500	
152, "Diecast Classics Inc.", Leong, Kelvin, "2155551555, 7588 Pompton St.", NULL, Allentown, PA, 70267, USA, 1216, 100600	
161, "Technics Stores Inc.", Hashimoto, Juri, "6505555809, 8408 Furth Circle", NULL, Burlingame, CA, 94017, USA, 1165, 84600	
166, "Handi Gifts Co.", Victorino, Wendi, "+65 224 1555, 106 Linden Road, Saldanha", "2nd Floor", Singapore, NULL, 069045, Singapore, 1612, 97900	
167, "Herkku Gifts", Oeztan, Veysel, "+47 2267 3219, Breheimen St, 121", PR 334, Sentrum, Bergen, NULL, "N-5000, Norway, 1504, 96800	
188, "Osaka Souvenirs Inc.", Franco, Keith, "2035557845, 49 Spinaker Dr, 10th Fl", NULL, Hove, "BN3 2EP, UK, 1286, 0	
189, "Porta Import Co.", de Casanova, "Esteban", "(1) 1555-2555", "Estade da saudade, n.59", NULL, Porto, NULL, "7600, Portugal, 1370, 82900	
171, "Daedalus Designs Imports", Ponce, "Martine", "(20,16,1555)-184, chaussee de Tournai", NULL, Lille, NULL, 59000, France, 1370, 82900	
172, "La Corne D'abondance", Co., Bertrand, Marie, "(1) 42 34 2555, 265, boulevard Charonne", NULL, Paris, NULL, 75012, France, 1337, 84300	
173, "Cambridge Collectables Co.", Tseng, Jerry, "6175555555, 4658 Baden Av", NULL, Cambridge, MA, 01247, USA, 1188, 43400	
175, "Gift Depot Inc.", King, Julie, "2035552570, 2593 South Bay Ln.", NULL, Bridgewater, CT, 97562, USA, 1323, 84300	
177, "Osaka Souvenirs Co.", Kentary, Mory, "+81 06 6342 5555", "1-6-20 Dojima", NULL, Kita-ku, Osaka, "530-0003", Japan, 1621, 81200	
181, "Vitachrome Inc.", Frick, Michael, "2125551265, 2678 Kingston Rd.", "Suite 101", NYC, NY, 10022, USA, 1286, 76400	
186, "Toys of Finland, Co.", Karttunen, Matti, "90-224 8555, "Keskuskatu 45", NULL, Helsinki, NULL, 21240, Finland, 1501, 96500	
187, "AV Stores Co.", Ashworth, Rachel, "(171) 555-1555, "Fauntleroy Circus", NULL, Manchester, NULL, "EC2 5NT", UK, 1501, 136800	
189, "Clover Collections Co.", Cassidy, Don, "+358 1862 1555, 25 Maiden Lane", "Floor No. 4", Dublin, NULL, 2, Ireland, 1504, 69400	
198, "Auto Moto Classics Inc.", Taylor, Leslie, "6175558428, 16780 Pompton St.", NULL, Brickhaven, MA, 58339, USA, 1216, 23000	
201, "UK Collectables, Ltd.", Devon, Elizabeth, "(71) 555-2282, 12, Berkeley Gardens Blvd", NULL, Liverpool, "WX1 6LT", UK, 1501, 92700	
202, "Canadian Gift Exchange Network", Tamuri, "Merry", "(604) 555-3392, 1900 Oak St.", NULL, Vancouver, BC, "V3B 2K1", Canada, 1323, 90300	
204, "Online Mini Collectables", Baraja, Miguel, "6175557555, 7635 Spinaker Dr.", NULL, Brickhaven, MA, 58339, USA, 1188, 68700	
205, "Toys4Gramps.com", Young, Julie, "6255557265, 78334 Hillside Dr, #121", Pasadena, CA, 90003, USA, 1165, 90700	
209, "Asian Shoppe", Network Co., Walker, Bryan, "60 1555-24, Place Kléber", NULL, Strasbourg, NULL, 67000, France, 1370, 53800	
211, "King Kong Collectables", Co., Gao, Mike, "+86 2251 1555", "Bank of China Tower", "1 Garden Road", Central Hong Kong", NULL, NULL, "Hong Kong, 1621, 58600	
216, "Enaco Distributors", Saavedra, Eduardo, "(93) 203 4555, "Rambla de Catalunya, 23", NULL, Barcelona, NULL, 08022, Spain, 1702, 60300	
219, "Boards & Toys Co.", Young, Mary, "3105552373, 4097 Douglas Av", NULL, Glendale, CA, 92561, USA, 1165, 11000	
223, "Naturalis Autos", Kloss, "Horst", "0372-555188, "Taucherstrasse 10", NULL, Cunerade, NULL, 01307, Germany, NULL, 0	
227, "Heintze Collectables", Ibsen, Palle, "86 21 3555", "Seasloget 45", NULL, Aarhus, NULL, 8200, Denmark, 1401, 120800	
233, "Québec Home Shopping Network", Fresnière, "Jean", "(514) 555-8054", "43 rue St. Laurent", NULL, Montréal, Québec, "H1J 1C9", Canada, 1286, 48700	
237, "ANG Resellers", Camino, "Alejandra", "(91) 745 6555, "Gran Via, 1", NULL, Madrid, NULL, 28001, Spain, NULL, 0	
239, "Collectable Mini Designs Co.", Thompson, Valarie, "760558146, 361 Furth Circle", NULL, "San Diego, CA, 92127, USA, 1166, 105000	
240, "giftsbyemail.co.uk", Bennett, "Heidi", "(198) 555-8888, "Garden House", "Crowther Way 23", Cowes, "Isle of Wight", "PO31 7PJ", UK, 1501, 93900	
242, "Alpha Cognac", Roulet, "Annette", "61 77 6555, "1 rue Alsace-Lorraine", NULL, Toulouse, NULL, 31000, France, 1370, 61100	
247, "Messenger Shopping Network", Messenger, "Renate", "(069) 0555984, "Magasinweg 7", NULL, Frankfurt, NULL, 60528, Germany, NULL, 0	
249, "Amico Models & Co.", Accorti, "Paolo", "011-4988855, "Via Monte Bianco 34", NULL, Torino, NULL, 10100, Italy, 1401, 113000	
250, "Lyon Souvenirs", Co., Da Silva, Daniel, "+33 1 38 7555, "27 rue du Colonel Pierre Avia", NULL, Paris, NULL, 75008, France, 1337, 68100	
256, "Auto Associates Co.", Tonini, "Daniel", "03 59 8555, 67, avenue de l'Europe", NULL, Versailles, NULL, 78000, France, 1370, 77900	
260, "The Special Editions Ltd.", Pfalzheim, "Hans-Joachim", "350, 0221-5554222", "Mehringdamm 350", NULL, Berlin, NULL, 10733, Germany, 1504, 120400	
260, "Royal Canadian Collectables, Ltd.", Lincoln, "Elizabeth", "(604) 555-4555, "23 Tsaussman Blvd.", NULL, Vancouver, BC, "V1C 8M4", Canada, 1323, 89600	
273, "Franken Gifts, Co.", Franken, "Peter", "089-0677555, "Berliner Platz 43", NULL, München, NULL, 80005, Germany, NULL, 0	
276, "Anna's Decorations, Ltd.", O'Hara, Anna, "02 9936 8555, "201 Miller Street", "Level 15", "North Sydney", NSW, 2060, Australia, 1611, 107800	

CSV 파일 읽기 예제

```
line_counter = 0      #파일의 총 줄수를 세는 변수
data_header = []      #data의 필드값을 저장하는 list
customer_list = []    #customer 개별 List를 저장하는 List

with open ("customers.csv") as customer_data: #customer.csv 파일을 customer_data 객체에 저장

    while 1:
        data = customer_data.readline() #customer.csv에 한줄씩 data 변수에 저장
        if not data: break #데이터가 없을 때, Loop 종료
        if line_counter==0:      #첫번째 데이터는 데이터의 필드
            data_header = data.split(",") #데이터의 필드는 data_header List에 저장, 데이터 저장시 ","로 분리
        else:
            customer_list.append(data.split(",")) #일반 데이터는 customer_list 객체에 저장, 데이터 저장시 ","로 분리
        line_counter += 1

    print("Header : ", data_header) #데이터 필드 값 출력
    for i in range(0,10):          #데이터 출력 (샘플 10개만)
        print ("Data",i,": ",customer_list[i])
    print (len(customer_list))     #전체 데이터 크기 출력
```

CSV 파일 쓰기 예제

```
line_counter = 0
data_header = []
employee = []
customer_USA_only_list = []
customer = None

with open ("customers.csv", "r") as customer_data:
    while 1:
        data = customer_data.readline()
        if not data:
            break
        if line_counter==0:
            data_header = data.split(",")
        else:
            customer = data.split(",")
            if customer[10].upper() == "USA": #customer 데이터의 offset 10번째 값
                customer_USA_only_list.append(customer) #즉 country 필드가 "USA" 것만
            line_counter+=1 #customer_USA_only_list에 저장

print ("Header : ", data_header)
for i in range(0,10):
    print ("Data : ", customer_USA_only_list[i])
print (len(customer_USA_only_list))

with open ("customers_USA_only.csv", "w") as customer_USA_only_csv:
    for customer in customer_USA_only_list:
        customer_USA_only_csv.write(",".join(customer).strip(' \n') + "\n")
#customer_USA_only_list 각자에 있는 데이터를 customers_USA_only.csv 파일에 쓰기
```

CSV 객체 사용하기

CSV 객체로 CSV 처리

- Text 파일 형태로 데이터 처리시 문장 내에 들어가 있는 “,” 등에 대해
전처리 과정이 필요
- 파이썬에서는 간단히 CSV파일을 처리하기 위해 **csv 객체**를 제공함
- 예제 데이터: `korea_floating_population_data.csv` (from <http://www.data.go.kr>)
- 예제 데이터는 국내 주요 상권의 유동인구 현황 정보
- 한글로 되어 있어 한글 처리가 필요

CSV 객체로 CSV 처리

- 시간대/조사일자/행정구역/날씨 등을 기준으로
연령별/성별 유동인가 해당 지역에 몇 명인가 표시

<https://goo.gl/ujHuXQ>

조사번호, 조사지역, 주구분, 조사일자, 시간대, X좌표, Y좌표, 행정구역명, 날씨, 남자10대, 남자20대, 남자30대, 남자40대, 남자50대, 여자10대, 여자20대, 여자30대, 여자40대, 여자50대
2544, 신촌네거리 계통로에 서울특별시, 현대아웃렛, 주중, 2010-06-21, 12시~13시 까지, 343099, 417482, 대전광역시 서구 월평동, 학률, 2, 24, 69, 50, 31, 4, 37, 64, 44, 26
2544, 신촌네거리 계통로에 서울특별시, 현대아웃렛, 주중, 2010-06-21, 19시~20시 까지, 343099, 417482, 대전광역시 서구 월평동, 학률, 19, 44, 28, 33, 21, 14, 56, 49, 43, 18
2544, 신촌네거리 계통로에 서울특별시, 현대아웃렛, 주중, 2010-06-20, 12시~13시 까지, 343099, 417482, 대전광역시 서구 월평동, 학률, 13, 33, 34, 61, 55, 13, 32, 29, 28, 12
2544, 신촌네거리 계통로에 서울특별시, 현대아웃렛, 주중, 2010-06-20, 19시~20시 까지, 343099, 417482, 대전광역시 서구 월평동, 학률, 23, 33, 32, 547, 129, 12, 39, 13, 46, 4
2545, 계통로에 신촌네거리 진영우축 보쌈한파13,000원화, 주중, 2010-06-21, 12시~13시 까지, 343121, 417343, 대전광역시 서구 월평동, 학률, 0, 9, 27, 21, 6, 5, 24, 20, 10, 6
2545, 계통로에 신촌네거리 진영우축 보쌈한파13,000원화, 주중, 2010-06-21, 19시~20시 까지, 343121, 417343, 대전광역시 서구 월평동, 학률, 4, 22, 12, 22, 12, 6, 30, 27, 7, 3
2545, 계통로에 신촌네거리 진영우축 보쌈한파13,000원화, 주중, 2010-06-20, 12시~13시 까지, 343121, 417343, 대전광역시 서구 월평동, 학률, 3, 8, 5, 8, 11, 2, 8, 9, 4, 3
2545, 계통로에 신촌네거리 진영우축 보쌈한파13,000원화, 주중, 2010-06-20, 19시~20시 까지, 343121, 417343, 대전광역시 서구 월평동, 학률, 6, 17, 19, 21, 27, 5, 16, 18, 11, 11
2546, 표간지장입구 주중, 2010-06-04, 12시~13시 까지, 311013, 523508, 경기도 수원시 청자1동, 맘을, 24, 26, 32, 45, 60, 21, 28, 98, 195, 201
2546, 표간지장입구 주중, 2010-06-04, 19시~20시 까지, 311013, 523508, 경기도 수원시 청자1동, 맘을, 13, 36, 18, 18, 90, 48, 36, 36, 114, 126
2546, 표간지장입구 주중, 2010-06-05, 12시~13시 까지, 311013, 523508, 경기도 수원시 청자1동, 맘을, 14, 25, 45, 36, 52, 9, 32, 42, 36, 34
2546, 표간지장입구 주중, 2010-06-05, 19시~20시 까지, 311013, 523508, 경기도 수원시 청자1동, 맘을, 36, 42, 89, 78, 64, 32, 45, 95, 78, 42
2547, 강마네거리 북쪽 흥대포 매장, 주중, 2010-06-24, 12시~13시 까지, 343529, 417164, 대전광역시 서구 강마동, 학률, 6, 10, 12, 12, 18, 0, 10, 29, 34, 46
2547, 강마네거리 북쪽 흥대포 매장, 주중, 2010-06-24, 19시~20시 까지, 343529, 417164, 대전광역시 서구 강마동, 학률, 30, 19, 11, 19, 26, 34, 39, 32, 36, 29
2547, 강마네거리 북쪽 흥대포 매장, 주중, 2010-06-26, 12시~13시 까지, 343529, 417164, 대전광역시 서구 강마동, 학률, 11, 7, 7, 9, 12, 9, 18, 14, 22, 30
2547, 강마네거리 북쪽 흥대포 매장, 주중, 2010-06-26, 19시~20시 까지, 343529, 417164, 대전광역시 서구 강마동, 학률, 9, 27, 19, 26, 35, 21, 21, 22, 30, 15
2549, 강마네거리 남쪽 LG전자 맘, 주중, 2010-06-24, 12시~13시 까지, 343463, 417150, 대전광역시 서구 강마동, 맘을, 3, 8, 7, 9, 18, 2, 18, 32, 25, 29
2549, 강마네거리 남쪽 LG전자 맘, 주중, 2010-06-24, 19시~20시 까지, 343463, 417150, 대전광역시 서구 강마동, 맘을, 23, 25, 18, 24, 50, 16, 49, 53, 41, 28
2549, 강마네거리 남쪽 LG전자 맘, 주중, 2010-06-26, 12시~13시 까지, 343463, 417150, 대전광역시 서구 강마동, 맘을, 3, 7, 12, 19, 26, 13, 25, 23, 23, 14
2549, 강마네거리 남쪽 LG전자 맘, 주중, 2010-06-26, 19시~20시 까지, 343463, 417150, 대전광역시 서구 강마동, 맘을, 15, 14, 26, 20, 9, 23, 20, 18, 6
2550, 화마을행복마을 예쁜마이스토리 앤조약, 주중, 2010-06-27, 12시~13시 까지, 344808, 416386, 대전광역시 서구 탄방동, 흐림, 8, 8, 8, 5, 9, 12, 14, 7, 10, 2
2550, 화마을행복마을 예쁜마이스토리 앤조약, 주중, 2010-06-27, 19시~20시 까지, 344808, 416386, 대전광역시 서구 탄방동, 흐림, 8, 14, 18, 5, 7, 6, 33, 17, 12, 14
2551, 개나리마아파트 흥무지점, 주중, 2010-06-27, 12시~13시 까지, 344797, 416293, 대전광역시 서구 탄방동, 흐림, 4, 5, 8, 6, 5, 0, 11, 7, 4, 9
2551, 개나리마아파트 흥무지점, 주중, 2010-06-27, 19시~20시 까지, 344797, 416293, 대전광역시 서구 탄방동, 흐림, 1, 19, 4, 5, 4, 0, 9, 8, 4, 6
2552, 신탄역 좌측 4거리 방한동, 주중, 2010-06-28, 12시~13시 까지, 348993, 427864, 대전광역시 대덕구 선봉동, 학률, 30, 17, 11, 3, 15, 23, 11, 5, 9, 35
2552, 신탄역 좌측 4거리 방한동, 주중, 2010-06-28, 19시~20시 까지, 348993, 427864, 대전광역시 대덕구 선봉동, 학률, 11, 26, 13, 14, 27, 41, 16, 15, 6, 35
2552, 신탄역 좌측 4거리 방한동, 주중, 2010-06-26, 12시~13시 까지, 348993, 427864, 대전광역시 대덕구 선봉동, 학률, 13, 23, 54, 12, 31, 7, 35, 28, 42, 59
2552, 신탄역 좌측 4거리 방한동, 주중, 2010-06-26, 19시~20시 까지, 348993, 427864, 대전광역시 대덕구 선봉동, 학률, 38, 58, 29, 48, 33, 27, 42, 34, 12, 8
2553, 신탄역 우측 거리난로, 주중, 2010-06-28, 12시~13시 까지, 349044, 427745, 대전광역시 대덕구 신탄지동, 학률, 28, 6, 27, 12, 10, 15, 27, 38, 53, 22
2553, 신탄역 우측 거리난로, 주중, 2010-06-28, 19시~20시 까지, 349044, 427745, 대전광역시 대덕구 신탄지동, 학률, 23, 48, 30, 31, 12, 16, 54, 18, 11, 47
2553, 신탄역 우측 거리난로, 주중, 2010-06-26, 12시~13시 까지, 349044, 427745, 대전광역시 대덕구 신탄지동, 학률, 42, 79, 68, 19, 18, 63, 45, 47, 13, 31
2553, 신탄역 우측 거리난로, 주중, 2010-06-26, 19시~20시 까지, 349044, 427745, 대전광역시 대덕구 신탄지동, 학률, 11, 27, 38, 38, 14, 42, 11, 12, 27, 6
2554, 송촌주민센터 흥미로, 주중, 2010-06-29, 12시~13시 까지, 349741, 418313, 대전광역시 대덕구 송촌동, 흐림, 3, 8, 34, 14, 12, 1, 6, 8, 5, 13
2554, 송촌주민센터 흥미로, 주중, 2010-06-29, 19시~20시 까지, 349741, 418313, 대전광역시 대덕구 송촌동, 흐림, 53, 18, 8, 50, 35, 58, 18, 7, 51, 30
2554, 송촌주민센터 흥미로, 주중, 2010-06-27, 12시~13시 까지, 349741, 418313, 대전광역시 대덕구 송촌동, 흐림, 8, 5, 16, 33, 11, 6, 14, 7, 20, 3
2554, 송촌주민센터 흥미로, 주중, 2010-06-27, 19시~20시 까지, 349741, 418313, 대전광역시 대덕구 송촌동, 흐림, 49, 25, 35, 17, 8, 40, 33, 10, 9, 18
2555, 차기나마트 거리, 주중, 2010-06-29, 12시~13시 까지, 349720, 418187, 대전광역시 대덕구 송촌동, 흐림, 2, 18, 19, 8, 28, 1, 8, 6, 3, 13

CSV 객체 활용

```
import csv  
reader = csv.reader(f,  
                     delimiter=',', quotechar='''',  
                     quoting=csv.QUOTE_ALL)
```

Attribute	Default	Meaning
delimiter	,	글자를 나누는 기준
lineterminator	\r\n	줄 바꿈 기준
quotechar	"	문자열을 둘러싸는 신호 문자
quoting	QUOTE_MINIMAL	데이터 나누는 기준이 quotechar에 의해 둘러싸인 레벨

<https://pymotw.com/2/csv/>

CSV 객체 활용

유동 인구 데이터 중 성남 데이터만 수집

```
import csv
seoung_nam_data = []
header = []
rownum = 0

with open("korea_floating_population_data.csv", "r", encoding="cp949") as p_file:
    csv_data = csv.reader(p_file) # CSV 객체를 이용해서 csv_data 읽기
    for row in csv_data:        # 읽어온 데이터를 한 줄씩 처리
        if rownum == 0:
            header = row # 첫 번째 줄은 데이터 필드로 따로 저장
            location = row[7]
            # "행정구역" 필드 데이터 추출, 한글 처리로 유니코드 데이터를 cp949로 변환
            if location.find(u"성남시") != -1:
                seoung_nam_data.append(row)
            # "행정구역" 데이터에 성남시가 들어가 있으면 seoung_nam_data List에 추가
        rownum += 1

with open("seoung_nam_floating_population_data.csv", "w", encoding="utf8") as s_p_file:
    writer = csv.writer(s_p_file, delimiter='|', quotechar='"', quoting=csv.QUOTE_ALL)
    # csv.writer를 사용해서 csv 파일 만들기 delimiter 필드 구분자
    # quotechar는 필드 각 데이터는 둑는 문자, quoting는 둑는 범위
    writer.writerow(header)          # 제목 필드 파일에 쓰기
    for row in seoung_nam_data:
        writer.writerow(row)         # seoung_nam_data[] 있는 정보 list에 쓰기
```

END

감사합니다.

The background of the slide features a white humanoid robot arm from the waist up. The robot's hand is holding a blue digital display that shows the binary sequence "01100110". The robot has a white, articulated arm and hand, and its body is visible in the background. The overall aesthetic is clean and modern.

Overview

Web Handling

가천대학교 | 산업경영공학과
최성철 교수

**하루 중 가장 많이
시간을 보내는 곳**

**하루 중 가장 많이
시간을 보내는 곳**

인터넷 = 웹

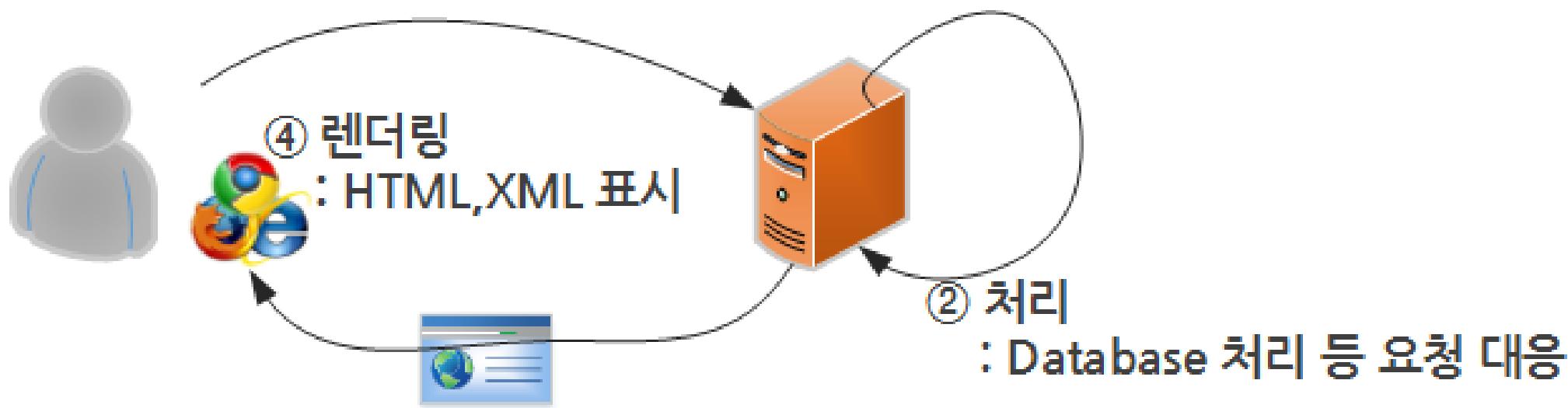
Web – 우리가 늘 쓰는 그 것

- World Wide Web(WWW), 줄여서 웹이라고 부름
- 우리가 늘 쓰는 인터넷 공간의 정식 명칭
- 팀 버너스리에 의해 1989년 처음 제안되었으며,
원래는 물리학자들간 정보 교환을 위해 사용됨
- 데이터 송수신을 위한 HTTP 프로토콜 사용,
데이터를 표시하기 위해 HTML 형식을 사용

Web은 어떻게 동작하는가?

① 요청

: 웹주소, Form, Header 등



③ 응답

: HTML, XML 등 으로 결과 반환

HTML(Hyper Text Markup Language)

- 웹 상의 정보를 구조적으로 표현하기 위한 언어
- 제목, 단락, 링크 등 요소 표시를 위해 Tag를 사용
- 모든 요소들은 꺼쇠 괄호 안에 둘러 쌓여 있음
`<title> Hello, World </title>` # 제목 요소, 값은 Hello, World
- 모든 HTML은 트리 모양의 포함관계를 가짐
- 일반적으로 웹 페이지의 HTML 소스파일은

컴퓨터가 다운로드 받은 후 웹 브라우저가 해석/표시

HTML(Hyper Text Markup Language)

```
<!doctype html>
<html>
  <head>
    <title>Hello HTML</title>
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

HTML 구조

<html> – <head> – <title>
– <body> – <p>

Element, Attribute Value 이루어짐

<tag attribute1= " att_value1" attribute2="att_value1 ">
보이는 내용(Value)
</tag>

Source: <http://ko.wikipedia.org/wiki/HTML>

왜 웹을 알아야 하는가?

- 정보의 보고, 많은 데이터들이 웹을 통해 공유됨

환율정보: <http://goo.gl/95q3mz>

상장기업 매출정보: <http://goo.gl/nwi8WE>

미국 특허정보: <http://goo.gl/wrmhWK>

- HTML도 일종의 프로그램, 페이지 생성 규칙이 있음
 - : 규칙을 분석하여 데이터의 추출이 가능
- 추출된 데이터를 바탕으로 하여 다양한 분석이 가능

[참고] Excel로 웹 데이터 추출하기

- 엑셀 실행 후 [데이터] → [웹] 메뉴 클릭
- 아래 Dialog에서 주소 입력 후 추출 대상 데이터 선택 (테이블 중심)



END

감사합니다.



HTML Parsing

Web Handling

가천대학교 | 산업경영공학과
최성철 교수

Web으로 할 수 있는 것들

Web 데이터를 다운로드 필요한 정보 저장하기

웹에서 데이터 다운로드

- 웹 상에 있는 파일을 로컬폴더에 저장함
- Built-in 모듈인 `urllib`의 `urlretrieve()` 함수 사용

```
import urllib.request #urllib 모듈 호출

url = "http://storage.googleapis.com/patents/grant_full_text/2014/ipg140107.zip"
# 다운로드 URL 주소

print ("Start Download")
fname, header = urllib.request.urlretrieve(url, 'ipg140107.zip')
#urlretrieve 함수 호출 (url 주소, 다운로드 될 파일명), 결과값으로 다운로드된 파일명과 Header 정보를 언패킹

print ("End Download")
```

어떨 때 쓸 수 있을까?

강의 자료 자동 다운로드하기

<http://web.eecs.umich.edu/~radev/coursera-slides/>

Draft slides for the online course

These slides are released on an as-is basis. The official versions will be posted w

- [01.01.pdf](#)
- [01.02.pdf](#)
- [01.03.pdf](#)
- [01.04.pdf](#)
- [01.05.pdf](#)
- [01.06.pdf](#)
- [01.07.pdf](#)
- [02.01.pdf](#)
- [02.02.pdf](#)
- [02.03.pdf](#)
- [02.04.pdf](#)
- [02.05.pdf](#)
- [02.06.pdf](#)
- [02.07.pdf](#)
- [03.01.pdf](#)
- [03.02.pdf](#)
- [03.03.pdf](#)
- [03.04.pdf](#)
- [03.05.pdf](#)
- [03.06.pdf](#)
- [03.07.pdf](#)
- [04.01.pdf](#)
- [04.02.pdf](#)
- [04.03.pdf](#)
- [04.04.pdf](#)
- [04.05.pdf](#)
- [05.01.pdf](#)
- [05.02.pdf](#)
- [05.03.pdf](#)
- [05.04.pdf](#)
- [05.05.pdf](#)
- [05.06.pdf](#)
- [05.07.pdf](#)
- [05.08.pdf](#)
- [06.01.pdf](#)
- [06.02.pdf](#)
- [06.03.pdf](#)
- [06.04.pdf](#)
- [06.05.pdf](#)
- [06.06.pdf](#)
- [06.07.pdf](#)

```
<UL>
<LI><A HREF="/lp/intro_co3_01.01_DR_Edit.pdf">01.01.pdf</A>
<LI><A HREF="/lp/intro_co3_01.02_DR_Edit.pdf">01.02.pdf</A>
<LI><A HREF="/lp/intro_co3_01.03_DR_Edit.pdf">01.03.pdf</A>
<LI><A HREF="/lp/intro_co3_01.04_DR_Edit.pdf">01.04.pdf</A>
<LI><A HREF="/lp/intro_co3_01.05_DR_Edit.pdf">01.05.pdf</A>
<LI><A HREF="/lp/intro_co3_01.06_DR_Edit.pdf">01.06.pdf</A>
<LI><A HREF="/lp/intro_co3_01.07_DR_Edit.pdf">01.07.pdf</A>
<LI><A HREF="/lp/intro_co3_02.01_DR_Edit.pdf">02.01.pdf</A>
<LI><A HREF="/lp/intro_co3_02.02_DR_Edit.pdf">02.02.pdf</A>
<LI><A HREF="/lp/intro_co2_02.03_DR_Edit.pdf">02.03.pdf</A>
<LI><A HREF="/lp/intro_co2_02.04_DR_Edit.pdf">02.04.pdf</A>
<LI><A HREF="/lp/intro_co2_02.05_DR_Edit.pdf">02.05.pdf</A>
<LI><A HREF="/lp/intro_co3_02.06_DR_Edit.pdf">02.06.pdf</A>
<LI><A HREF="/lp/intro_co3_02.07_DR_Edit.pdf">02.07.pdf</A>
<LI><A HREF="/lp/intro_co2_03.01_DR_Edit.pdf">03.01.pdf</A>
<LI><A HREF="/lp/intro_co2_03.02_DR_Edit.pdf">03.02.pdf</A>
<LI><A HREF="/lp/intro_co2_03.03_DR_Edit.pdf">03.03.pdf</A>
<LI><A HREF="/lp/intro_co2_03.04_DR_Edit.pdf">03.04.pdf</A>
<LI><A HREF="/lp/intro_co3_03.05_DR_Edit.pdf">03.05.pdf</A>
<LI><A HREF="/lp/intro_co3_03.06_DR_Edit.pdf">03.06.pdf</A>
<LI><A HREF="/lp/intro_co3_03.07_DR_Edit.pdf">03.07.pdf</A>
<LI><A HREF="/lp/intro_co4_04.01_DR_Edit.pdf">04.01.pdf</A>
<LI><A HREF="/lp/intro_co5_04.02_DR_Edit.pdf">04.02.pdf</A>
<LI><A HREF="/lp/intro_co5_04.03_DR_Edit.pdf">04.03.pdf</A>
<LI><A HREF="/lp/intro_co5_04.04_DR_Edit.pdf">04.04.pdf</A>
<LI><A HREF="/lp/intro_co5_04.05_DR_Edit.pdf">04.05.pdf</A>
<LI><A HREF="/lp/intro_co1_05.01_DR_Edit.pdf">05.01.pdf</A>
<LI><A HREF="/lp/intro_co1_05.02_DR_Edit.pdf">05.02.pdf</A>
<LI><A HREF="/lp/intro_co1_05.03_DR_Edit.pdf">05.03.pdf</A>
<LI><A HREF="/lp/intro_co1_05.04_DR_Edit.pdf">05.04.pdf</A>
<LI><A HREF="/lp/intro_co5_05.05_DR_Edit.pdf">05.05.pdf</A>
<LI><A HREF="/lp/intro_co5_06_DR_Edit.pdf">05.06.pdf</A>
<LI><A HREF="/lp/intro_co5_06.07_DR_Edit.pdf">05.07.pdf</A>
<LI><A HREF="/lp/intro_co5_06.08_DR_Edit.pdf">05.08.pdf</A>
<LI><A HREF="/lp/intro_co6_06.01_DR_Edit.pdf">06.01.pdf</A>
<LI><A HREF="/lp/intro_co6_06.02_DR_Edit.pdf">06.02.pdf</A>
```

자동화 하기 위해선

HTML Parsing

HTML Parsing

- 웹으로 부터 데이터를 추출해 내는 행위
- 대부분의 웹은 사용자 요구에 따라 동적으로 생성됨

페이지를 생성하는 파일 (프로그램)

http://finance.naver.com/item/main.nhn?code=005930

본 프로그램에서는 상장코드에 따라 다른 정보를 생성함
(GET 방식)

해당 프로그램의 변수
code: 변수명, 005930: 값

- HTML Parsing을 위해서는 **HTML 생성 규칙 파악**
- HTML은 Tree 구조 → **구조 파악 필요**

HTML 규칙 파악하기 (1/2)

<http://finance.naver.com/item/main.nhn?code=005930>

삼성전자 005930 코스피	 2016.11.04 기준(장마감) 결제시간 기업개요 ▾
1,627,000 전일대비 ▲11,000 +0.68%	전일 1,616,000 고가 1,634,000 (상한가 2,100,000) 거래량 141,995
	시가 1,605,000 저가 1,605,000 (하한가 1,132,000) 거래대금 230,766 백만

- HTML 열어서 분석 하기
(브라우저에서 오른쪽 마우스 클릭 후 소스보기 클릭)
- HTML 파일에서 유일하게 위 데이터를 나타낼 수 있는 패턴을 찾아야 함

HTML 규칙 파악하기 (2/2)

```
<dl class="blind">
  <dt>종목 시세 정보</dt>
  <dd>2016년 11월 04일 16시 10분 기준 잠마감</dd>
  <dd>종목명 삼성전자</dd>
  <dd>종목코드 005930 코스피</dd>
  <dd>현재가 1,627,000 전일대비 상승 11,000 플러스 0.68 퍼센트</dd>
  <dd>전일가 1,616,000</dd>
  <dd>시가 1,605,000</dd>
  <dd>고가 1,634,000</dd>
  <dd>상한가 2,100,000</dd>
  <dd>저가 1,605,000</dd>
  <dd>하한가 1,132,000</dd>
  <dd>거래량 141,995</dd>
  <dd>거래대금 230,766백만</dd>
</dl>
```

<dl class="blind"> ~ </dl>
사이 데이터 존재

각 데이터는 <dd> ~ </dd>로 나타나며
데이터 생성순서는 일시, 종목 명 ~ 거래대금 순

HTML Parsing을 위한 두가지 방법

-
- 1) 정규식 이용하기
 - 2) 모듈 활용하기

END

감사합니다.



Regular Expression

Web Handling

가천대학교 | 산업경영공학과
최성철 교수

정규식 - Regular Expression

- 정규 표현식, regexp 또는 regex 등으로 불림
- 복잡한 문자열 패턴을 정의하는 문자 표현 공식
- 특정한 규칙을 가진 문자열의 집합을 추출

010-0000-0000 `^\d{3}\-\d{4}\-\d{4}$`

203.252.101.40 `^\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}$`

From wiki, <https://goo.gl/HLntbd>

정규식 for HTML Parsing

- 주민등록 번호, 전화번호, 도서 ISBN 등 형식이 있는 문자열을 원본 문자열로부터 추출함
- HTML 역시 tag를 사용한 일정한 형식이 존재하여 정규식으로 추출이 용이함
- 관련자료: <http://www.nextree.co.kr/p4327/>

정규식 for HTML Parsing

- 문법 자체는 매우 방대, **스스로 찾아서 하는 공부 필요**
- 필요한 것들은 인터넷 검색을 통해 찾을 수 있음
- 기본적인 것을 공부 한 후 넓게 적용하는 것이 중요

이메일 : ^[a-zA-Z0-9]+@[a-zA-Z0-9]+\\$ or

<https://goo.gl/FNTwIO>

^[_0-9a-zA-Z-]+@[0-9a-zA-Z-]+(.[_0-9a-zA-Z-]+)*\\$

휴대폰 : ^01(?:0|1|[6-9]) - (?:(?:\d{3})|\d{4}) - \d{4}\\$

일반전화 : ^\d{2,3} - \d{3,4} - \d{4}\\$

주민등록번호 : \d{6} \d- [1-4]\d{6}

IP 주소 : ([0-9]{1,3}) \. ([0-9]{1,3}) \. ([0-9]{1,3}) \. ([0-9]{1,3})

해쉬태그: #([A-Za-z0-9가-힣]+)

<https://goo.gl/FNTwIO>

정규식 기본 문법 #1

문자 클래스 []: [와] 사이의 문자들과 매치라는 의미

예) [abc] ← 해당 글자가 a,b,c중 하나가 있다.
“a”, “before”, “deep” , “dud”, “sunset”

“-”를 사용 범위를 지정할 수 있음

예) [a-zA-Z] – 알파벳 전체, [0-9] – 숫자 전체

<https://wikidocs.net/4308>

?

정규식 연습장 활용하기

- 1) 정규식 연습장(<http://www.regexr.com/>) 으로 이동
- 2) 테스트하고 싶은 문서를 Text 란에 삽입
- 3) 정규식을 사용해서 찾아보기

다시 리턴

정규식 기본 문법 - 메타 문자

정규식 표현을 위해 원래 의미 X, 다른 용도로 사용되는 문자

. ^ \$ * + ? { } [] \ | ()

. - 줄바꿈 문자인 `\n`를 제외한 모든 문자와 매치 a[.]b

* - 앞에 있는 글자를 반복해서 나올 수 있음

tomor*ow tomorrow tomoow tomorrrrow

+ - 앞에 있는 글자를 최소 1회 이상 반복

<https://wikidocs.net/4308>

정규식 기본 문법 - 메타 문자

정규식 표현을 위해 원래 의미 X, 다른 용도로 사용되는 문자

. ^ \$ * + ? { } [] \ | ()

{m.n} - 반복 횟수를 지정 {1,} , {0,} {1,3}

203.252.101.40 [0-9]{1,3} \d{1,3}

? - 반복 횟수가 1회 01[01]?-[0-9]{4}-[0-9]{4}

| - or (0|1){3} ^ - not

<https://wikidocs.net/4308>

정규식 추출 연습

- ① 정규식 연습장(<http://www.regexr.com/>) 으로 이동
- ② 구글 USPTO Bulk Download 데이터페이지 소스 보기 클릭
- ③ 소스 전체 복사후 정규식 연습장 페이지에 붙여넣기
- ④ 상단 Expression 부분을 수정해가며 “Zip”로 끝나는 파일명만 추출
- ⑤ Expression에 `(http)(.+)(zip)` 를 입력

<http://www.google.com/googlebooks/uspto-patents-grants-text.html>

END

감사합니다.

Lab - Regular Expression

Web Handling

가천대학교 | 산업경영공학과
최성철 교수

01100
00110

정규식 in 파이썬

- re 모듈을 import 하여 사용 : `import re`
- 함수: `search` – 한 개만 찾기, `findall` – 전체 찾기
- 추출된 패턴은 tuple로 반환됨
- 연습 - 특정 페이지에서 ID만 추출하기 <http://goo.gl/U7mSQL>
- ID 패턴: [영문대소문자|숫자] 여러 개, 별표로 끝남
"([A-Za-z0-9]+W*W*W*)" 정규식

Code #1

```
import re
import urllib.request

url = "http://goo.gl/U7mSQI"
html = urllib.request.urlopen(url)
html_contents = str(html.read())
id_results = re.findall(r"([A-Za-z0-9]+[W*W*W*])", html_contents)
#findall 전체 찾기, 패턴대로 데이터 찾기

for result in id_results:
    print(result)
```

Code #2

```
import urllib.request # urllib 모듈 호출
import re

url = "http://www.google.com/googlebooks/uspto-patents-grants-text.html"
#url 값 입력
html = urllib.request.urlopen(url) # url 열기
html_contents = str(html.read().decode("utf8"))
# html 파일 읽고, 문자열로 변환

url_list = re.findall(r"(http)(.+)(zip)", html_contents)
for url in url_list:
    print("".join(url)) # 출력된 Tuple 형태 데이터 str으로 join
```

Code #3

```
import urllib.request # urllib 모듈 호출
import re

base_url = "http://web.eecs.umich.edu/~radev/coursera-slides/"
#url 값 입력
html = urllib.request.urlopen(base_url)
html_contents = str(html.read().decode("utf8"))

url_list = re.findall(r"\nlp[0-9a-zA-Z_\.]*\.\pdf", html_contents)
for url in url_list:
    file_name = "".join(url)
    full_url = base_url + file_name
    print(full_url)
    fname, header = urllib.request.urlretrieve(full_url, file_name)
    print ("End Download")
```

<http://web.eecs.umich.edu/~radev/coursera-slides/>

Draft slides for the online course

These slides are released on an as-is basis. The official versions will be posted weekly on the coursera site.

- 01.01.pdf
- 01.02.pdf
- 01.03.pdf
- 01.04.pdf
- 01.05.pdf
- 01.06.pdf
- 01.07.pdf
- 02.01.pdf
- 02.02.pdf
- 02.03.pdf
- 02.04.pdf
- 02.05.pdf
- 02.06.pdf
- 02.07.pdf
- 03.01.pdf
- 03.02.pdf
- 03.03.pdf
- 03.04.pdf
- 03.05.pdf
- 03.06.pdf
- 03.07.pdf
- 04.01.pdf
- 04.02.pdf
- 04.03.pdf
- 04.04.pdf
- 04.05.pdf
- 05.01.pdf
- 05.02.pdf
- 05.03.pdf
- 05.04.pdf
- 05.05.pdf
- 05.06.pdf
- 05.07.pdf
- 05.08.pdf
- 06.01.pdf
- 06.02.pdf
- 06.03.pdf
- 06.04.pdf
- 06.05.pdf
- 06.06.pdf
- 06.07.pdf

정규식 in 파이썬 for html

```
<dl class="blind">
    <dt>종목 시세 정보</dt>
    <dd>2016년 11월 04일 16시 10분 기준 장마감</dd>
    <dd>종목명 삼성전자</dd>
    <dd>종목코드 005930 코스피</dd>
    <dd>현재가 1,627,000 전일대비 상승 11,000 플러스 0.68 퍼센트</dd>
    <dd>전일가 1,616,000</dd>
    <dd>시가 1,605,000</dd>
    <dd>고가 1,634,000</dd>
    <dd>상한가 2,100,000</dd>
    <dd>저가 1,605,000</dd>
    <dd>하한가 1,132,000</dd>
    <dd>거래량 141,995</dd>
    <dd>거래대금 230,766백만</dd>
</dl>
```

이 데이터는 어떻게 뽑을까?

정규식 in 파이썬 for html

```
<dl class="blind">
    <dt>종목 시세 정보</dt>
    <dd>2016년 11월 04일 16시 10분 기준 장마감</dd>
    <dd>종목명 삼성전자</dd>
    <dd>종목코드 005930 코스피</dd>
    <dd>현재가 1,627,000 전일대비 상승 11,000 플러스 0.68 퍼센트</dd>
    <dd>전일가 1,616,000</dd>
    <dd>시가 1,605,000</dd>
    <dd>고가 1,634,000</dd>
    <dd>상한가 2,100,000</dd>
    <dd>저가 1,605,000</dd>
    <dd>하한가 1,132,000</dd>
    <dd>거래량 141,995</dd>
    <dd>거래대금 230,766백만</dd>
</dl>
```

① <dl class="blind"> ~~~~ </dl> 에 있는
② <dd> ~~~~ </dd> 정보를 추출하면 됨

정규식 in 파이썬 for html

① <dl class="blind"> ~~~~ </dl>

($\backslash<dl\ class=\backslash"blind\backslash">$)([$\backslash{s}\backslash{S}$]+?)($\backslash<\backslash/dl\backslash>$)

<dl class에서 시작해서 / 사이에 아무 글자나 있고 / </dl> 로 끝내기

② <dd> ~~~~ </dd> 정보를 추출하면 됨

($\backslash<dd\backslash>$)([$\backslash{s}\backslash{S}$]+?)($\backslash<\backslash/dd\backslash>$)

<dd>에서 시작해서 / 사이에 아무 글자나 있고 / </dl>로 끝내기

① 를 먼저 찾고 ① 안에 ②를 차례대로 찾으면 됨

정규식 in 파이썬 for html

```
import urllib.request
import re

url = "http://finance.naver.com/item/main.nhn?code=005930"
html = urllib.request.urlopen(url)
html_contents = str(html.read().decode("ms949"))

stock_results = re.findall("(₩<dl class='blind'>)([₩₩₩]+?)(₩</dl>)", html_contents)
samsung_stock = stock_results[0] # 두 개 tuple 값 중 첫번째 패턴
samsung_index = samsung_stock[1] # 세 개의 tuple 값 중 두 번째 값
# 하나의 괄호가 tuple index가 됨
index_list= re.findall("(₩<dd>)([₩₩₩]+?)(₩</dd>)", samsung_index)

for index in index_list:
    print(index[1]) # 세 개의 tuple 값 중 두 번째 값
```

END

감사합니다.



XML Overview

XML and JSON

가천대학교 | 산업경영공학과
최성철 교수

우리가 처리하는 데이터 저장 방식들

CSV, HTML

XML, JSON

eXtensible Markup Language

XML이란

- 데이터의 구조와 의미를 설명하는 TAG(MarkUp)를 사용하여 표시하는 언어
- TAG와 TAG사이에 값이 표시되고, 구조적인 정보를 표현할 수 있음
- HTML과 문법이 비슷, 대표적인 데이터 저장 방식

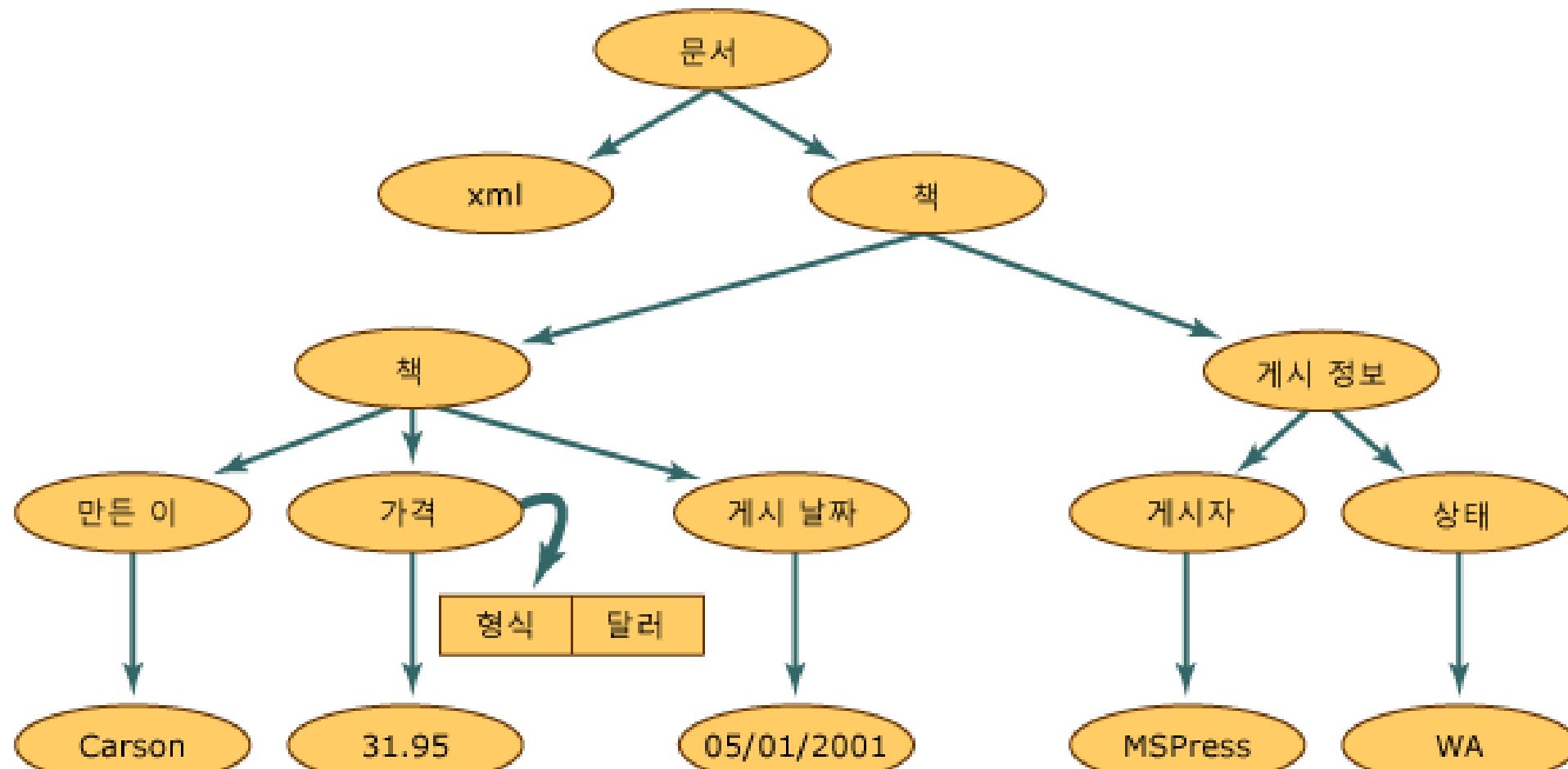
XML이란

- 정보의 구조에 대한 정보인 스키마와 DTD 등으로 정보에 대한 정보(메타정보)가 표현되며, 용도에 따라 다양한 형태로 변경가능
- XML은 컴퓨터(예: PC ↔ 스마트폰)간에 정보를 주고받기 매우 유용한 저장 방식으로 쓰이고 있음

XML 예제

```
<?xml version="1.0"?>
<고양이>
  <이름>나비</이름>
  <품종>샴</품종>
  <나이>6</나이>
  <중성화>예</중성화>
  <발톱 제거>아니요</발톱 제거>
  <등록 번호>lzz138bod</등록 번호>
  <소유자>이강주</소유자>
</고양이>
```

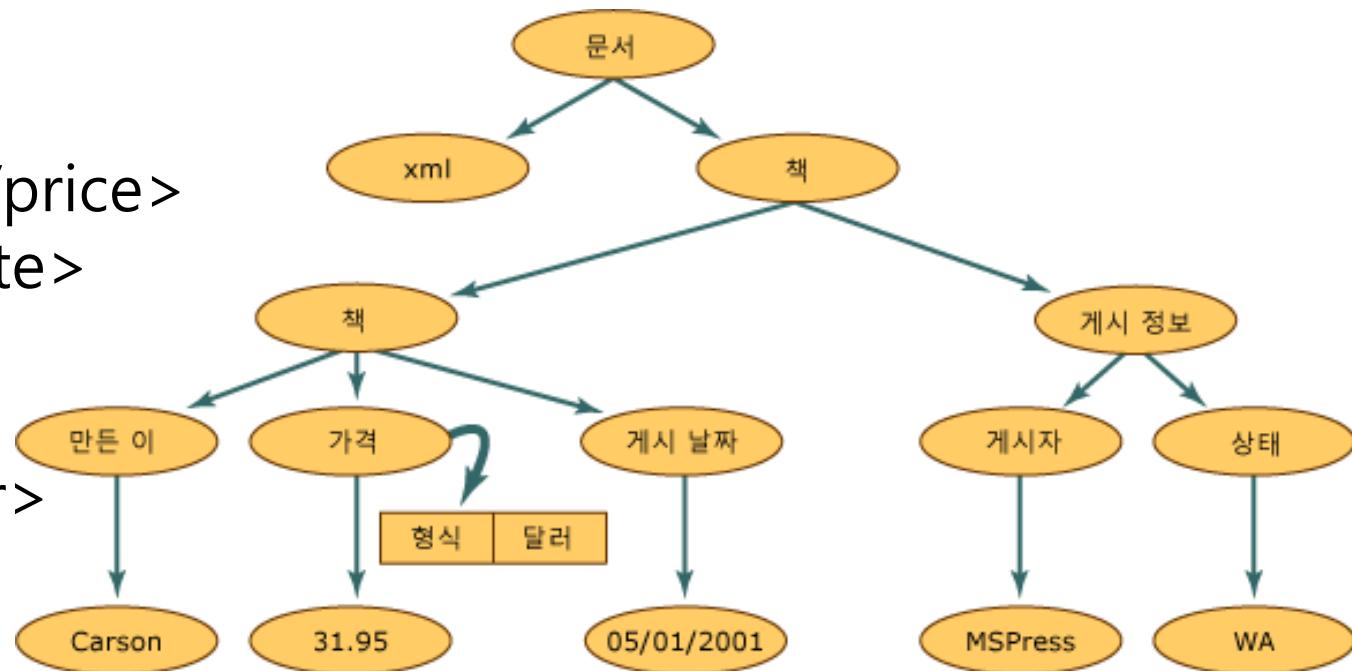
XML 형태로 만들어 보기



<http://goo.gl/7mO15w>

XML 형태로 만들어 보기

```
<?xml version="1.0"?>
<books>
  <book>
    <author>Carson</author>
    <price format="dollar">31.95</price>
    <pubdate>05/01/2001</pubdate>
  </book>
  <pubinfo>
    <publisher>MSPress</publisher>
    <state>WA</state>
  </pubinfo>
</books>
```



<http://goo.gl/7mO15w>

XML Parsing in Python

- XML도 HTML과 같이 구조적 MarkUp 언어
- 정규표현식으로 Parsing이 가능함
- 그러나 좀 더 손쉬운 도구들이 개발되어 있음
- 가장 많이 쓰이는 parser인 **beautifulsoup**으로 파싱

END

감사합니다.



Lab – XML Parsing

Web Handling

가천대학교 | 산업경영공학과
최성철 교수

BeautifulSoup

- HTML, XML등 Markup 언어 Scraping을 위한 대표적인 도구
- <https://www.crummy.com/software/BeautifulSoup/>
- lxml 과 html5lib 과 같은 Parser를 사용함
- 속도는 상대적으로 느리나 간편히 사용할 수 있음

<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

BeautifulSoup

Parser	Python 2.7		Python 3.2	
	Speed (KB/s)	Success rate	Speed (KB/s)	Success rate
Beautiful Soup 3.2 (SGMLParser)	211	100%	-	-
html5lib (BS3 treebuilder)	253	99%	-	-
Beautiful Soup 4.0 + lxml	255	100%	2140	96%
html5lib (lxml treebuilder)	270	99%	-	-
Beautiful Soup 4.0 + html5lib	271	98%	-	-
Beautiful Soup 4.0 + HTMLParser	299	59%	1705	57%
html5lib (simpletree treebuilder)	332	100%	-	-
HTMLParser	5194	52%	3918	57%
lxml	17925	100%	14258	96%

[https://www.crummy.com/2012/01/22/0](https://www.crummy.com/2012/01/22/)

beautifulsoup 설치

- conda 가상 환경으로 lxml과 beautifulsoup 설치

```
activate python_kmooc
```

```
conda install lxml
```

```
conda install -c anaconda beautifulsoup4=4.5.1
```

beautifulsoup 모듈 사용

- 모듈 호출

```
from bs4 import BeautifulSoup
```

- 객체 생성

```
soup = BeautifulSoup(books_xml, "lxml")
```

- Tag 찾는 함수 `find_all` 생성

```
soup.find_all("author")
```

beautifulsoup 모듈 사용

- `find_all`: 정규식과 마찬가지로 해당 패턴을 모두 반환
- `find('invention-title')`
Tag 네임 = title
- `get_text()`: 반환된 패턴의 값 반환 (태그와 태그 사이)

```
<invention-title id="d2e43">  
Adjustable shoulder device for hard upper torso suit  
</invention-title>
```

<http://goo.gl/aeKMGS>, <http://goo.gl/lKhFzh> 참고

beautifulsoup Example

- 데이터 다운로드 받기

<https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/books.xml>

```
from bs4 import BeautifulSoup

with open("books.xml", "r", encoding="utf8") as books_file:
    books_xml = books_file.read() # File을 String으로 읽어오기

soup = BeautifulSoup(books_xml, "lxml") # lxml Parser를 사용해서 데이터 분석

# author가 들어간 모든 element 추출
for book_info in soup.find_all("author"):
    print(book_info)
    print(book_info.get_text())
```

beautifulsoup 예제 데이터

- 미국 특허청 (USPTO) 특허 데이터는 XML로 제공됨
- 해당 데이터중 등록번호 “08621662” 인

“Adjustable shoulder device for hard upper torso suit” 분석

참고: <http://www.google.com/patents/US20120260387>

<https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/US08621662-20140107.XML>

- XML 데이터를 Beautiful Soup을 통해 데이터 추출

beautifulsoup 예제 데이터

- 데이터 다운로드 받기

<https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/US08621662-20140107.XML>

```
import urllib.request
from bs4 import BeautifulSoup

with open("US08621662-20140107.XML", "r", encoding="utf8") as patent_xml:
    xml = patent_xml.read() # File을 String으로 읽어오기

soup = BeautifulSoup(xml, "lxml") # lxml parser 호출

#invention-title tag 찾기
invention_title_tag = soup.find("invention-title")
print(invention_title_tag.get_text())
```

beautifulsoup 예제 데이터 응용

- 특허의 출원번호, 출원일, 등록번호, 등록일, 상태, 특허명을 추출

```
<publication-reference>      등록 관련 정보  
<document-id>  
<country>US</country>  
<doc-number>08621662</doc-number> 등록번호  
<kind>B2</kind> 상태  
<date>20140107</date>      등록일자  
</document-id>  
</publication-reference>
```

```
<application-reference appl-type="utility"> 출원 관련 정보  
<document-id>  
<country>US</country>  
<doc-number>13175987</doc-number> 출원 번호  
<date>20110705</date>      출원일  
</document-id>  
</application-reference>
```

beautifulsoup 예제 데이터 응용

```
publication_reference_tag = soup.find("publication-reference")
p_document_id_tag = publication_reference_tag.find("document-id")
p_country = p_document_id_tag.find("country").get_text()
p_doc_number = p_document_id_tag.find("doc-number").get_text()
p_kind = p_document_id_tag.find("kind").get_text()
p_date = p_document_id_tag.find("date").get_text()
```

```
application_reference_tag = soup.find("application-reference")
a_document_id_tag = application_reference_tag.find("document-id")
a_country = p_document_id_tag.find("country").get_text()
a_doc_number = p_document_id_tag.find("doc-number").get_text()
a_date = p_document_id_tag.find("date").get_text()
```

```
<publication-reference>
<document-id>
<country>US</country>
<doc-number>08621662</doc-number>
<kind>B2</kind>
<date>20140107</date>
</document-id>
</publication-reference>

<application-reference appl-type="utility">
<document-id>
<country>US</country>
<doc-number>13175987</doc-number>
<date>20110705</date>
</document-id>
</application-reference>
```

[연습] ipg140107.xml 분석

- ipa110106.xml 파일은 11년 첫째주에 나온 출원 특허를 모은 파일

<https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/ipa110106.XML>

- 개별 특허들을 나눠서 CSV 형태로 저장 하는 문제
- 개별 특허 시작은 <?xml version="1.0" 시작함
- 분할된 특허 문서로 부터 특허의 등록번호, 등록일자, 출원 번호, 출원 일자, 상태, 특허 제목을 추출하여 CSV로 만들 것

END

감사합니다.

JSON Overview

XML and JSON

가천대학교 | 산업경영공학과
최성철 교수

01100
00110



JavaScript Object Notation

JSON

- JavaScript Object Notation
- 원래 웹 언어인 **Java Script**의 데이터 객체 표현 방식
- 간결성으로 기계/인간이 모두 이해하기 편함
- 데이터 용량이 적고, **Code**로의 전환이 쉬움
- 이로 인해 **XML**의 대체제로 많이 활용되고 있음

JSON 예시

```
{  
  "users": [  
    {  
      "name": "John",  
      "age": 25  
    },  
    {  
      "name": "Mark",  
      "age": 29  
    },  
    {  
      "name": "Sarah",  
      "age": 22  
    }  
  "dataTitle": "JSON Tutorial!",  
  "swiftVersion": 2.1  
}
```

Python의 Dict Type과 유사,
Key:Value 쌍으로 데이터 표시

<https://goo.gl/gVy0Ms>

JSON vs XML

JSON

```
{  
  "siblings": [  
    {"firstName": "Anna", "lastName": "Clayton"},  
    {"lastName": "Alex", "lastName": "Clayton"}  
  ]  
}
```

<http://www.pcmag.com/encyclopedia/term/56790/json>

XML

```
<siblings>  
  < sibling >  
    < firstName >Anna</firstName>  
    < lastName >Clayton</lastName>  
  </ sibling >  
  < sibling >  
    < firstName >Alex</firstName>  
    < lastName >Clayton</lastName>  
  </ sibling >  
</ siblings >
```

JSON in Python

- json 모듈을 사용하여 손 쉽게 파싱 및 저장 가능
- 데이터 저장 및 읽기는 dict type과 상호 호환 가능
- 웹에서 제공하는 API는 대부분 정보 교환 시 JSON 활용
- 페이스북, 트위터, Github 등 거의 모든 사이트
- 각 사이트마다 Developer API의 활용법을 찾아 사용

END

감사합니다.



Lab – JSON Handling

Web Handling

가천대학교 | 산업경영공학과
최성철 교수

JSON Read

- JSON 파일의 구조를 확인 → 읽어온 후 → Dict Type처럼 처리

```
{"employees": [  
    {"firstName": "John", "lastName": "Doe"},  
    {"firstName": "Anna", "lastName": "Smith"},  
    {"firstName": "Peter", "lastName": "Jones"}  
]}
```

```
import json  
  
with open("json_example.json", "r", encoding="utf8") as f:  
    contents = f.read()  
    json_data = json.loads(contents)  
    print(json_data["employees"])
```

JSON Data

https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/json_example.json

JSON Write

- Dict Type으로 데이터 저장 → json모듈로 Write

```
import json

dict_data = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

with open("data.json", "w") as f:
    json.dump(dict_data, f)
```

XML Read → JSON Write

- CSV로 저장된 ipa110106.xml 파일을 JSON으로 변환하기

<https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/ipa110106.XML>

- 분할된 특허 문서로 부터 특허의 등록번호, 등록일자, 출원 번호, 출원 일자, 상태, 특허 제목을 추출하여 JSON로 만들 것
- 각 문서의 Key 값으로 Application ID를 활용함

Twitter 데이터 가져오기

<http://jinse.datastats.info/1>

- Twitter에서 제공하는 Developer API를 사용하여 트위터 데이터 수집
- 수집되는 데이터 형태는 JSON 형태로 제공함
- <https://dev.twitter.com/> Oauth 인증으로 데이터를 주고 받을 수 있음
- 다양한 기능을 이해하기 위해 API 문서의 공부가 필요
<https://dev.twitter.com/overview/api>

Twitter 데이터 가져오기

<http://jinse.datastats.info/1>

- 트위터 가입후 Twitter App 생성 <https://apps.twitter.com/>

The screenshot shows the Twitter Application Management interface. At the top, there's a navigation bar with a user profile icon and a dropdown menu. Below it, a blue header bar contains the text "Twitter Apps". On the left, there are two app entries:

- gachon_cs50**: A blue gear icon with a Twitter bird inside. Below it is the text "Test for teamlab".
- python_kmooc**: A blue gear icon with a Twitter bird inside. Below it is the text "Test application for Python K-MOOC".

In the bottom right corner of the main area, there is a white button with a grey border and the text "Create New App". This button is highlighted with a red rectangular box.

Twitter 데이터 가져오기

<http://jinse.datastats.info/1>

- 트위터 App 정보 입력

Create an application

Application Details

Name *

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL

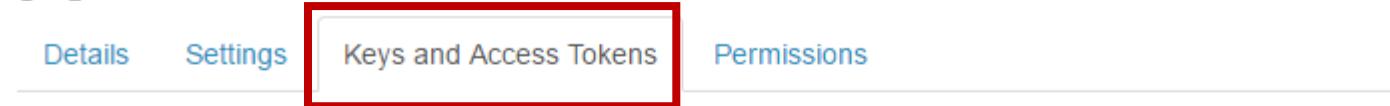
Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

Twitter 데이터 가져오기

<http://jinse.datastats.info/1>

- Keys와 Access Tokens로 가서 API Key 값 확인

python_kmooc



Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	QT... miS9
Consumer Secret (API Secret)	Ux4EYBYI... jRd7I

Access Level Read and write (modify app permissions)

Owner SungchulChoi82

Owner ID 63969436

Twitter 데이터 가져오기

- Keys와 Access Tokens로 가서 API Key 값 확인

Consumer Key (API Key) QTwAI 57JzamIS9

Consumer Secret (API Secret) Ux4EY8Yt /HJX3Om9JKCP3TjRd7I

Access Level Read and write (modify app permissions)

Owner SungchulCho182

Owner ID 63969436

API 사용을 위한 모듈 설치

- conda 가상 환경으로 requests 와 oauthlib 설치

```
Acitvate python_kmooc
```

```
conda install requests
```

```
pip install requests-oauthlib
```

Code

http://jinse.datastats.info/1

- oauth 접속 권한 받기

```
import requests
from requests_oauthlib import OAuth1

consumer_key = '확인한 consumer_key'
consumer_secret = '확인한 consumer_secret'
access_token = '확인한 access_token'
access_token_secret = '확인한 access_token_secret'

oauth = OAuth1(client_key=consumer_key, client_secret=consumer_secret,
               resource_owner_key=access_token, resource_owner_secret=access_token_secret)
```

Code <http://jinse.datastats.info/1>

- 특정 계정의 타임라인 데이터 가져오기

```
# Twitter REST api // screen_name 은 트위터 계정명
url = 'https://api.twitter.com/1.1/statuses/user_timeline.json?screen_name={0}'.format('naver_d2')
r = requests.get(url=url, auth=oauth)
statuses = r.json()

for status in statuses:
    print(status['text'], status['created_at'])
```

END

감사합니다.



Wrap-Up Python!

What's NEXT?

01100
00110

가천대학교 | 산업경영공학과
최성철 교수

이제 기본은
다 배웠음

이제 뭘 해야 할까?

입문자 to 숙련자

파이썬 스터디의 주요 분야

- Data 분석: 머신러닝, 통계, Visualization
- 웹 프로그래밍: 웹 프레임워크
- 파이썬 성능 향상: 동시성, 프로파일링
- 코드 작성 & 협업: 파이썬 문서화, Github

Data 분석

Scikit-Learn – 머신러닝 라이브러리 <http://scikit-learn.org/>

matplotlib – 데이터 시각화 라이브러리 <http://matplotlib.org/>

numpy – 과학 연산을 위한 라이브러리 <http://www.numpy.org/>

pandas – 데이터 Hadling을 위한 라이브러리 <http://pandas.pydata.org/>

Tensorflow – 딥러닝/머신러닝 라이브러리 <https://www.tensorflow.org/>

Data 분석



Andrew Ng



- 스탠포드 Andrew Ng 교수님 강의
 - Coursera, 머신러닝 분야의 정석
 - Matlab to Python 도전 권장
- <https://www.coursera.org/learn/machine-learning>

- 밑바닥부터 시작하는 데이터 과학
- 파이썬의 데이터 분석 기초 과정
- 학부 수준 통계, 확률, 선형대수 이해 필수

Data 분석



- Coding the Matrix (Coursera)
- 브라운 대학의 Kevin
- 선형대수학을 Python으로 배우는 과정
<http://codingthematrix.com/>

- 홍콩 과기대 김성훈 교수님
- 모두를 위한 머신러닝/딥러닝 강의
- 한국어로 된 최고의 딥러닝 입문 과정
<https://hunkim.github.io/ml/>

웹 프로그래밍

Django – 가장 넓게 쓰이는 파이썬 웹 프레임워크

<https://www.djangoproject.com/>

flask – 경량 파이썬 웹 프레임워크, Easy & Simple

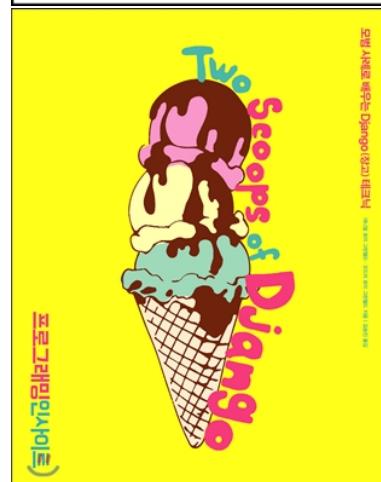
<http://flask.pocoo.org/>

웹 프로그래밍



- django Girls Tutorial
- <https://tutorial.djangogirls.org/ko/>
- 누구나 쉽게 웹 프로그래밍 시작하기
- ask Django
- <https://www.facebook.com/groups/askdjango/>
- Django의 다양한 개발 사례를 공유
- 오프라인 세미나 개최

웹 프로그래밍



- 파이썬 웹 프로그래밍 실전편
- 기초부터 실전까지 입문서로 추천

- Two Scoops of Django
- Django 중급 이상을 위한 좋은 가이드북

파이썬 성능향상

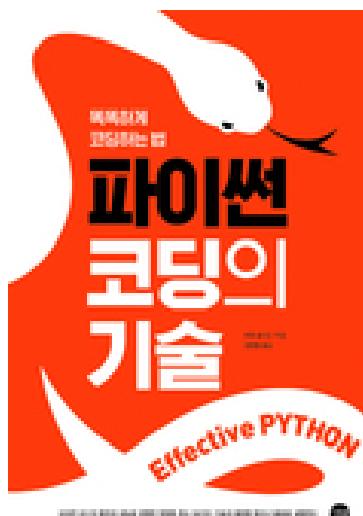
동시성 - 한번에 한 가지 이상의 Task를 실행시키기!

profile - 메모리/연산이 많은 지점 확인 하기

파이썬 성능향상

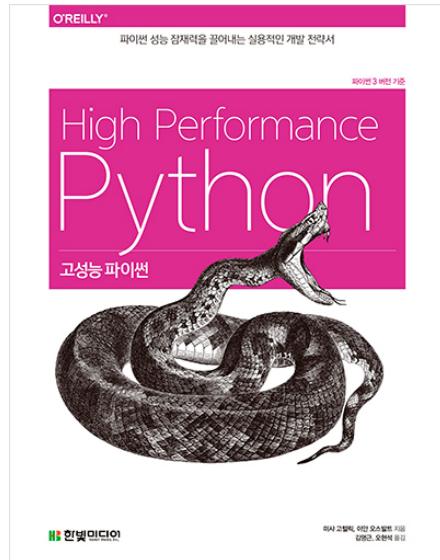


Fluent Python – 전문가를 위한 파이썬
(한빛미디어, 2016)

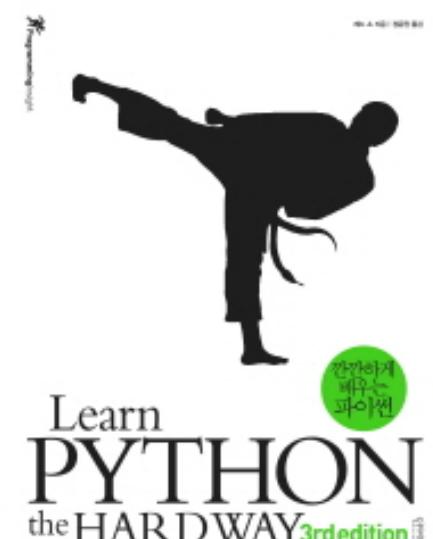


Effective Python - 파이썬 코딩의 기술
(길벗, 2016)

파이썬 성능향상



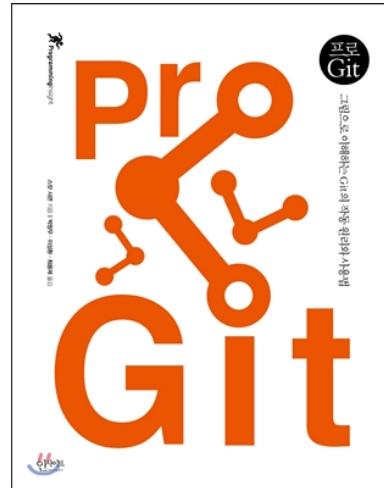
High Performance Python - 고성능 파이썬
(한빛미디어, 2016)



깐깐하게 배우는 파이썬
(인사이트, 2016)

프로그래밍 잘하기

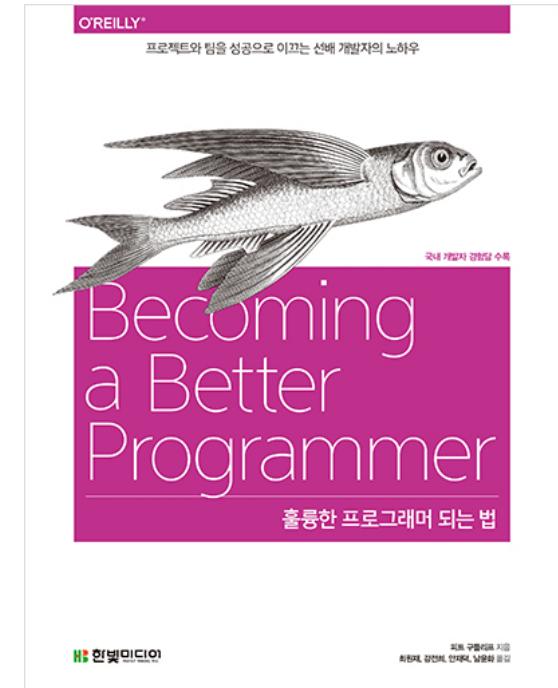
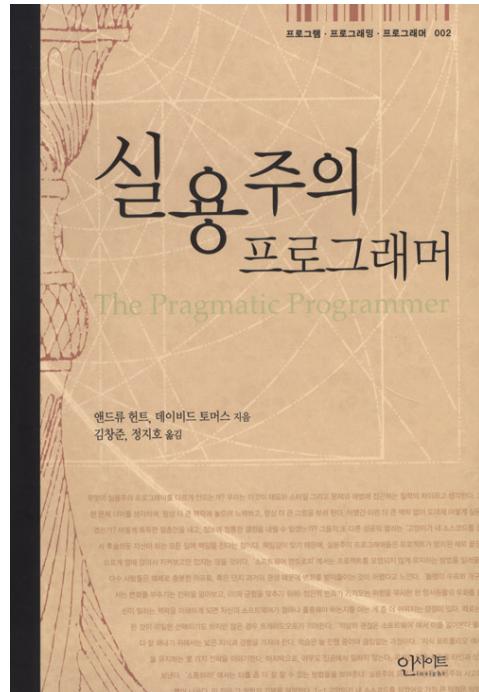
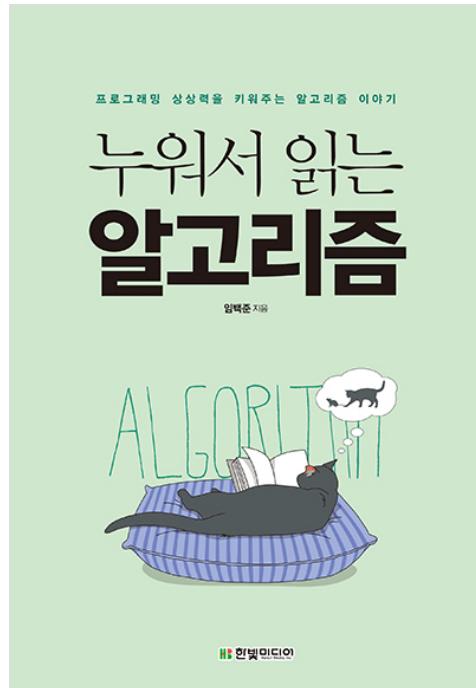
Git, Github study – 같이 개발하는 방법 배우기



- 가천대학교 CS50 강좌 - <https://goo.gl/FTrK90>
- 홍콩과기대 김성훈 교수 github flow
<https://goo.gl/t9K8gn>, <https://goo.gl/Ek35Zi>

프로그래밍 잘하기

프로그래밍 자체에 대한 공부



END

감사합니다.