

파이썬 익숙해지기(조금 심화 문법)

Session 5


NEXT X LIKELION 정석민

| 지난 과제 리뷰

| 목차

1. 파이썬?
2. 파이썬 자료형(list, dictionary, function)
3. 클래스

파이썬!

Feb 2024	Feb 2023	Change	Programming Language		Ratings	Change
1	1			Python	15.16%	-0.32%
2	2			C	10.97%	-4.41%
3	3			C++	10.53%	-3.40%
4	4			Java	8.88%	-4.33%
5	5			C#	7.53%	+1.15%
6	7	⬆		JavaScript	3.17%	+0.64%
7	8	⬆		SQL	1.82%	-0.30%
8	11	⬆		Go	1.73%	+0.61%
9	6	⬇		Visual Basic	1.52%	-2.62%
10	10			PHP	1.51%	+0.21%
11	24	⬆		Fortran	1.40%	+0.82%

24년 1월 기준

출처: <https://www.tiobe.com/tiobe-index/>

Worldwide, Mar 2024 :

Rank	Change	Language	Share	1-year trend
1		Python	28.59 %	+1.0 %
2		Java	15.79 %	-0.5 %
3		JavaScript	8.7 %	-0.8 %
4		C#	6.77 %	-0.0 %
5		C/C++	6.76 %	-0.0 %
6	⬆	R	4.71 %	+0.5 %
7	⬇	PHP	4.5 %	-0.7 %
8		TypeScript	2.86 %	+0.1 %
9		Swift	2.74 %	+0.5 %
10		Objective-C	2.4 %	+0.1 %
11		Rust	2.36 %	+0.3 %

24년 3월

출처: <https://pypl.github.io/PYPL.html/>

왜 파이썬? 장점?

Java

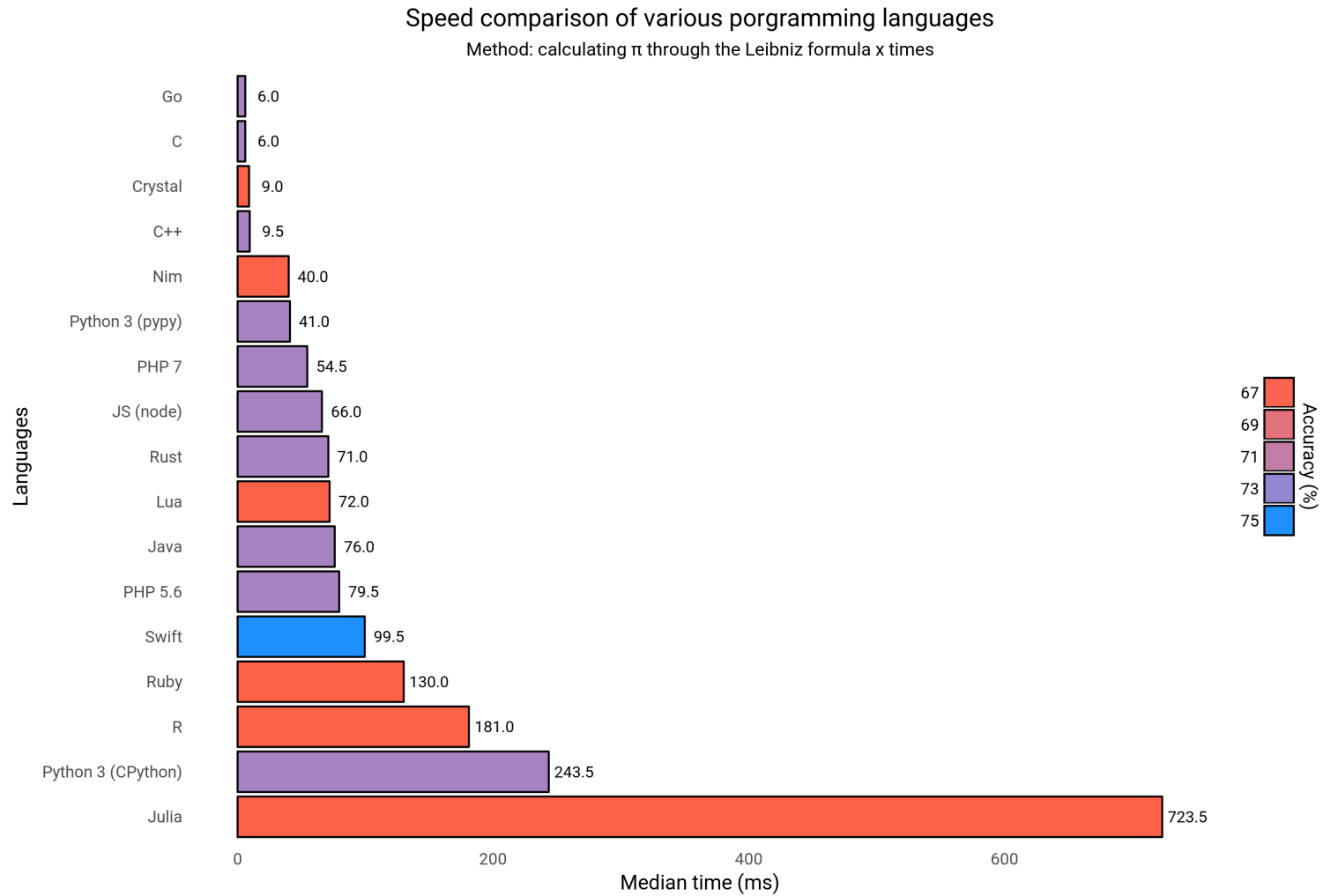
```
public class Main
{
    public static void main(String[] args) {
        int a= 10, b=20;
        int result = a+b;
        System.out.println("The result a+b = " + result);    }
}
```

Python

```
a=10
b=20

print ('The result a+b = ', a+b)
```

단점?,,속도?



| 파이썬!



어찌 됐든 우리에게는 최적의 언어!

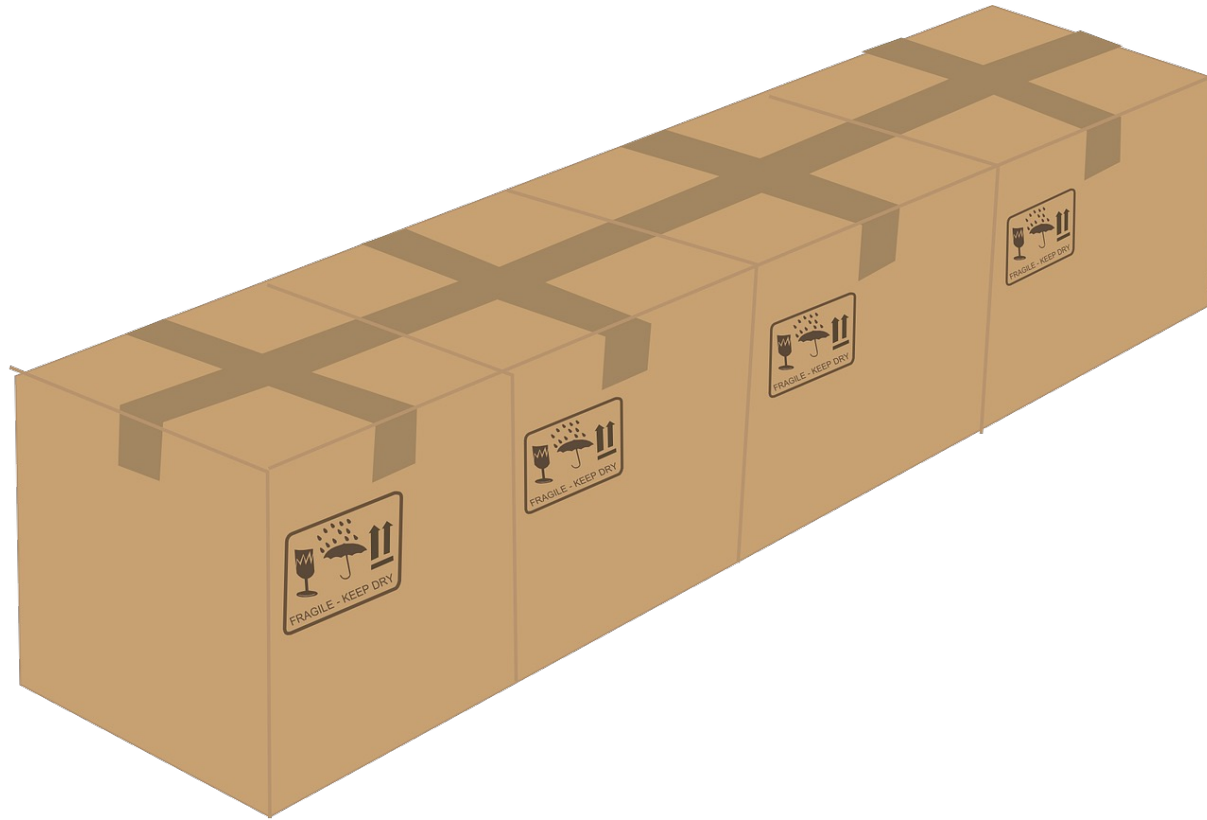
| 많이 쓸 개념과 문법들..

1. List(배열 자료구조)

2. Dictionary

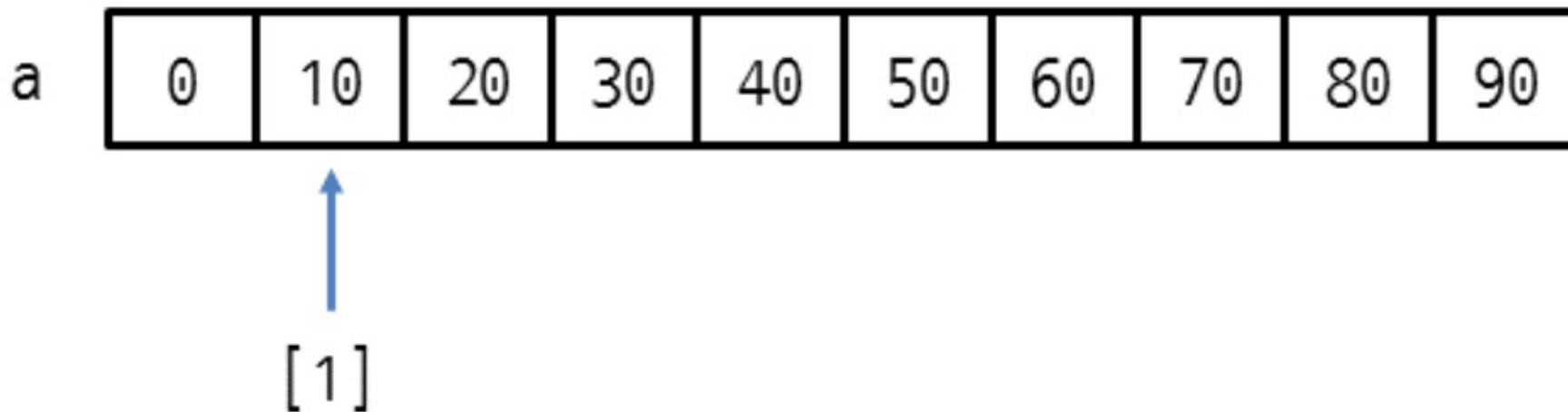
3. function

| 1. 배열



여러 데이터를 하나의 변수에 저장할 수 있게 하는 자료구조!

| 1. 배열



1. list(배열 자료구조)

```
[1] # 대괄호 ( []) 로 감싸주고 각 요소 값은 쉼표 (,) 로 구분  
odd = [1, 3, 5, 7, 9]
```

```
print(type(odd))
```

```
<class 'list'>
```

```
[2] a=[]  
    b=[1, 2, 3]  
    c = ['Life', 'is', 'too', 'short']  
    d = [1, 2, 'Life', 'is']  
    e = [1, 2, ['Life', 'is']]
```

Python에서는 List 라는 자료형을 통해 배열 자료구조를 제공한다!!

다양한 type의 요소를 담을 수 있다!!

| 1. list



1. list(배열 자료구조)

```
[1] # 대괄호 ( []) 로 감싸주고 각 요소 값은 쉼표 (,) 로 구분  
odd = [1, 3, 5, 7, 9]
```

```
print(type(odd))
```

```
<class 'list'>
```

```
[2] a=[]  
    b=[1, 2, 3]  
    c = ['Life', 'is', 'too', 'short']  
    d = [1, 2, 'Life', 'is']  
    e = [1, 2, ['Life', 'is']]
```

Python에서는 List 라는 자료형을 통해 배열 자료구조를 제공한다!!

다양한 type의 요소를 담을 수 있다!!

1. list

```
[2] a=[]  
    b=[1, 2, 3]  
    c = ['Life', 'is', 'too', 'short']  
    d = [1, 2, 'Life', 'is']  
    e = [1, 2, ['Life', 'is']]
```

```
[10] print('b[1] :', b[1])  
      print('c[0:2] :', c[0:2]) # 0-1 index에 해당하는 데이터를 list 형태로 반환(슬라이싱)  
      print('d[-1] :', d[-1]) # 가장 마지막 index의 데이터를 반환=  
      print('e[2][2] :', e[2][1]) # 2번 index의 1번 index를 반환  
      print('b[0]+d[1] :', b[0] + d[1])  
      print('len(e) :', len(e)) # e list의 데이터 수, 길이를 반환
```

```
b[1] : 2  
c[0:2] : ['Life', 'is']  
d[-1] : is  
e[2][2] : is  
b[0]+d[1] : 3  
len(e) : 3
```

각 인덱스를 통해 list 내 데이터에 접근 할 수 있다!
list의 길이도 구하고, 데이터 간 연산까지!

1. list

```
[2] a=[]  
    b=[1, 2, 3]  
    c = ['Life', 'is', 'too', 'short']  
    d = [1, 2, 'Life', 'is']  
    e = [1, 2, ['Life', 'is']]
```

```
[5] print(a[1])
```

```
-----  
IndexError                                Traceback (most recent call last)  
<ipython-input-5-bb6a8a2f4689> in <cell line: 1>()  
----> 1 print(a[1])  
  
IndexError: list index out of range
```

데이터가 없는 인덱스에 접근하면 IndexError가 발생합니다!

1. list 활용 및 관련 함수

```
[12] # list 간 연산
a = [1, 2, 3]
b = [4, 5, 6]
```

```
print(a + b)
```

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

```
[15] # list 반복
a = ['나', "짱"]
```

```
print(a * 4)
```

```
['나', '짱', '나', '짱', '나', '짱', '나', '짱']
```

```
▶ # list 데이터 수정
a = ['나', '좀', '별로다']
a[2] = '멋지다'
```

```
print(a)
```

```
☞ ['나', '좀', '멋지다']
```

```
[19] # list 데이터 삭제(del 이용)
a = [1, 2, 3, 4, 0, 5]
del a[4]
```

```
print(a)
```

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

```
▶ # remove 함수 : 데이터 삭제
a = ['일', '이', '삼', '일', '이', '삼']
a.remove('삼') # 첫 번째로 나오는 '삼' 제거
```

```
print(a)
```

```
['일', '이', '일', '이', '삼']
```

```
[31] # append 함수 : 요소 추가
a = [1, 2, 3]
a.append(4) # 가장 마지막 요소로 추가
```

```
print(a)
```

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

```
▶ # insert 함수 : 요소 삽입
a = ['일', '이', '사', '오']
a.insert(2, '삼')
```

```
print(a)
```

```
☞ ['일', '이', '삼', '사', '오']
```

```
[27] # sort 함수 : 요소 정렬
a = [2, 3, 1, 4, 6, 5]
a.sort()
```

```
b = ['c', 'a', 'b']
b.sort()
```

```
print(a)
print(b)
```

```
[1, 2, 3, 4, 5, 6]
['a', 'b', 'c']
```

```
▶ # reverse 함수 : list 뒤집기
a = [1, 2, 3, 4, 5]
a.reverse()
```

```
print(a)
```

```
☞ [5, 4, 3, 2, 1]
```

```
[30] # index 함수 : 해당 데이터의 index 반환
a = ['나는', 'NEXT', '12기', '입니다']
a.index('NEXT')
```

```
1
```


1. 문자열도 배열??

```
[8] name = 'NEXT입니다'
```

```
print('type :', type(name))  
print('0번째 인덱스 :', name[0])  
print('1부터 3까지 인덱스 :', name[1:4])  
print('-2번째 인덱스 :', name[-2])
```

```
type : <class 'str'>  
0번째 인덱스 : N  
1부터 3까지 인덱스 : EXT  
-2번째 인덱스 : 니
```

문자열 역시 일종의 배열 자료구조!!

| 1. 간단 실습



1. 간단 실습1

<https://www.acmicpc.net/problem/2743>

문제

알파벳으로만 이루어진 단어를 입력받아, 그 길이를 출력하는 프로그램을 작성하시오.

입력

첫째 줄에 영어 소문자와 대문자로만 이루어진 단어가 주어진다. 단어의 길이는 최대 100이다.

출력

첫째 줄에 입력으로 주어진 단어의 길이를 출력한다.

예제 입력 1 [복사](#)

pulljima

예제 출력 1 [복사](#)

8



```
# input : 특정 값을 입력받아 저장하는 함수  
a = input()  
  
print(a)
```

정석민 -> 입력값

정석민 -> 출력값

Input() 함수 이용하시면 됩니다!

1. 간단 실습2

```
a = ['삼다수', '백산수', '에비앙', '피지워터', '아이시스', '백두산']
```

위 list가 있을 때,
오른쪽과 같이 출력되도록 해보세요!!

0번째	리스트	값은	삼다수
1번째	리스트	값은	백산수
2번째	리스트	값은	에비앙
3번째	리스트	값은	피지워터
4번째	리스트	값은	아이시스
5번째	리스트	값은	백두산

2. Dictionary



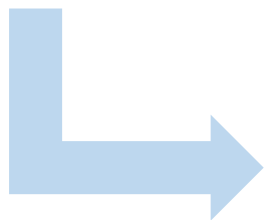
대응관계를 나타낼 수 있다!

{아메리카노:2500, 카페라테:3000, 딸기주스:3500, ...}

2. Dictionary

Key	Value
이름	정석민
나이	20
학과	생명공학부

대응관계를 나타낼 수 있다!



```
[50] jsm_profile = {'이름' : '정석민', '나이' : 20, '학과' : '생명공학부'}  
  
print(type(jsm_profile))  
  
<class 'dict'>
```

| 2. Dictionary 활용 및 관련 함수



2. Dictionary 활용 및 관련 함수

```
[57] # 딕셔너리 요소 추가
profile = {'이름' : '정석민', '나이' : 20, '학과' : '생명공학부'}
profile['전화번호'] = '010-2784-0000'

print(profile)
```

```
{'이름': '정석민', '나이': 20, '학과': '생명공학부', '전화번호': '010-2784-0000'}
```

```
[58] # 딕셔너리 요소 삭제
del profile['이름']

print(profile)
```

```
{'나이': 20, '학과': '생명공학부', '전화번호': '010-2784-0000'}
```

```
[59] # 딕셔너리 키 리스트 생성
keys = profile.keys()

print(keys)
```

```
dict_keys(['나이', '학과', '전화번호'])
```

```
[63] # 딕셔너리 값 리스트 생성
values = profile.values()

print(values)
```

```
dict_values([20, '생명공학부', '010-2784-0000'])
```

```
[63] # 딕셔너리 값 리스트 생성
values = profile.values()

print(values)
```

```
dict_values([20, '생명공학부', '010-2784-0000'])
```

```
[65] # 딕셔너리 아이템 리스트 생성
items = profile.items()

print(items)
print(list(items))
```

```
dict_items([('나이', 20), ('학과', '생명공학부'), ('전화번호', '010-2784-0000')])
[('나이', 20), ('학과', '생명공학부'), ('전화번호', '010-2784-0000')]
```

```
[68] # 딕셔너리 특정 키 존재 여부 확인
is_in_there = 'name' in profile

print(is_in_there)
```

```
False
```


2. 간단 실습 3

```
▶ # 주어진 문장  
sentence = "to be or not to be, that is the question"
```

```
# 문장을 단어로 분리, 특수 문자를 제거  
words = sentence.replace(',', ' ').split()
```

```
# 각 단어의 빈도를 세기 위한 딕셔너리를 생성  
word_count = {}
```

```
# 단어의 빈도를 계산합니다.
```



```
print(word_count)
```

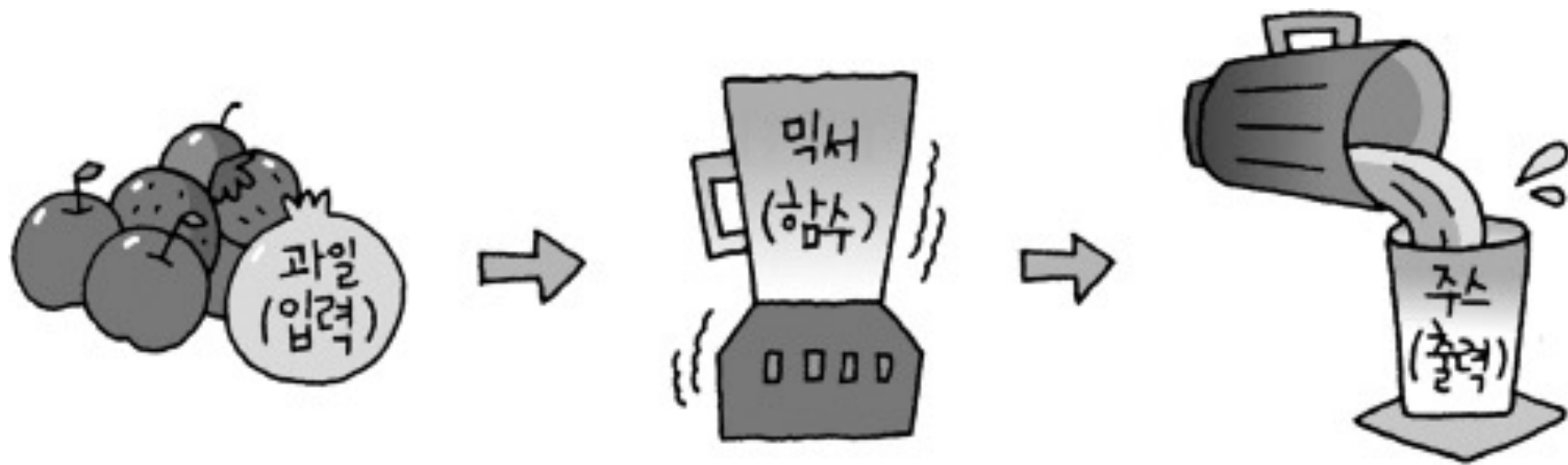
```
➞ {'to': 2, 'be': 2, 'or': 1, 'not': 1, 'that': 1, 'is': 1, 'the': 1, 'question': 1}
```

빈칸을 채워주세요!

쉬는시간



| 함수



| 함수

```
[18] n1, n2, n3, n4 = 3, 5, 9, 54
```

```
    a = (n1 + n2) * (n1 - n2)
```

```
    b = (n3 + n4) * (n3 - n4)
```

```
    print(a)
```

```
    print(b)
```

```
-16
```

```
-2835
```

| Function



| Function

```
[19] n1, n2, n3, n4 = 3, 5, 9, 54
```

```
def hap_cha_formula(p1, p2):  
    result = (p1 + p2) * (p1 - p2)  
    return(result)
```

```
[20] a = hap_cha_formula(n1, n2)  
     b = hap_cha_formula(n3, n4)
```

```
print(a)  
print(b)
```

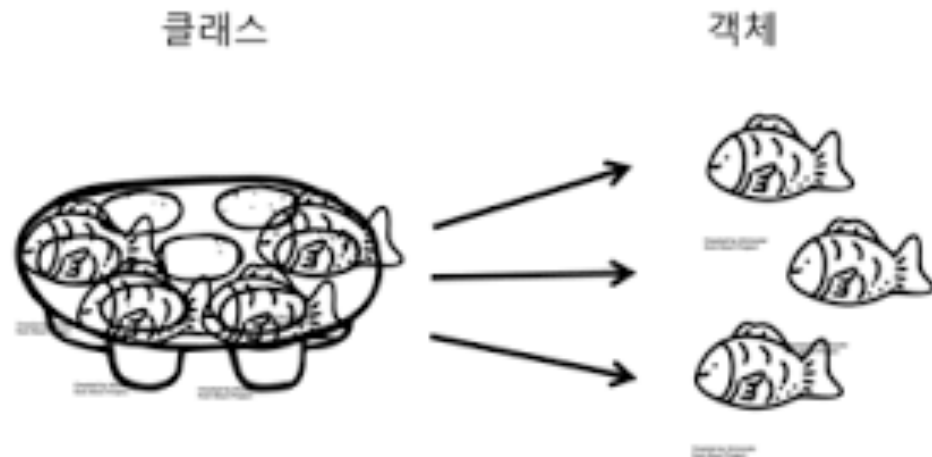
```
-16  
-2835
```

| 함수 간단실습

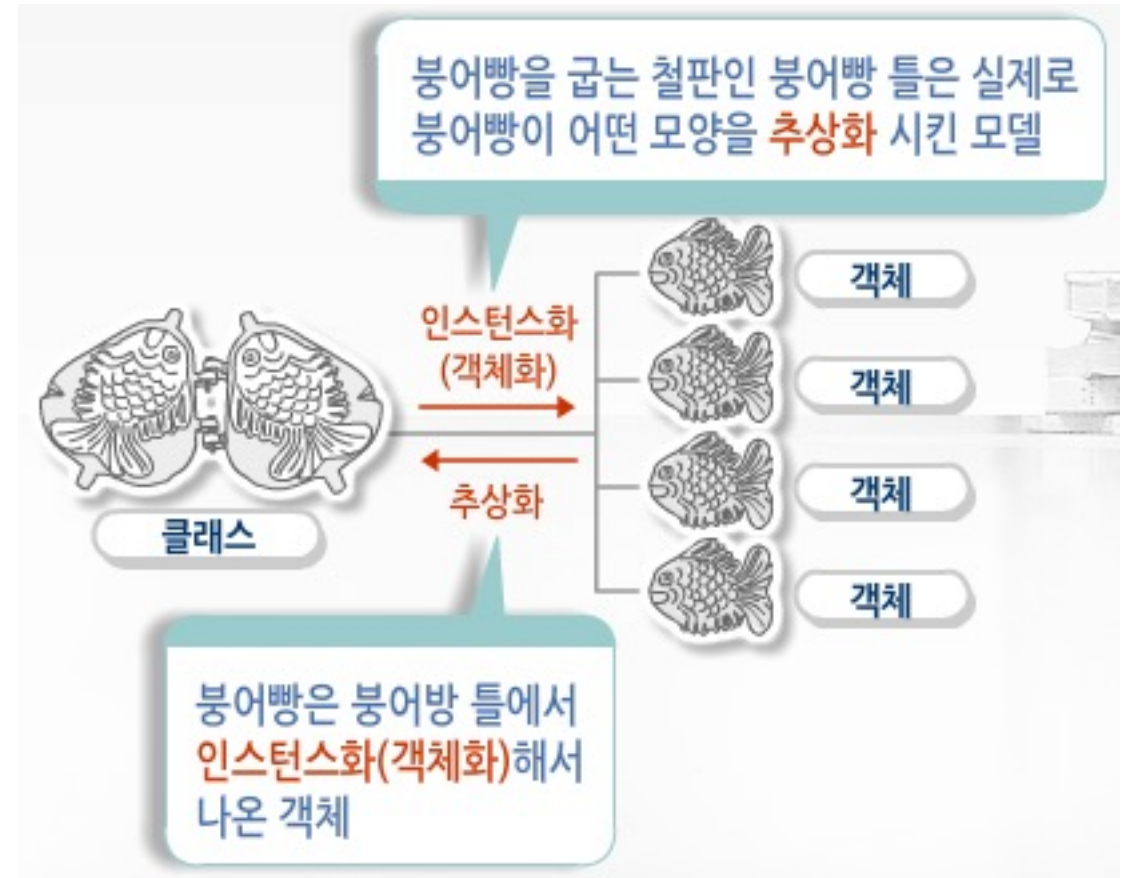
Q1. 홀수 짝수 판별하기

주어진 자연수가 홀수인지 짝수인지 판별해 주는 함수 (`is_odd`) 를 작성해 보자.

클래스?



Created by Olga
from Noun Project



클래스?

```
1 fishbread1 = {
2     "flavor" : "팥",
3     "price" : 1000,
4     "amount": 1,
5 }
6 fishbread2 = {
7     "flavor" : "슈크림",
8     "price" : 1200,
9     "amount": 1,
10 }
11 fishbread3 = {
12     "flavor" : "초코",
13     "price" : 1500,
14     "amount": 1,
15 }
16
17 def one_more_fishbread1():
18     fishbread1["amount"] += 1
19     fishbread1["price"] += 1000
20
21 def one_more_fishbread2():
22     fishbread2["amount"] += 1
23     fishbread2["price"] += 1200
```



```
1 class Fishbread:
2     def __init__(self, flavor, price, amount):
3         self.flavor = flavor
4         self.price = price
5         self.amount = amount
6
7     def one_more(self):
8         self.price += self.price
9         self.amount += 1
10
11 fishbread1 = Fishbread("팥", 1000, 1)
12 fishbread2 = Fishbread("슈크림", 1200, 1)
13 fishbread3 = Fishbread("초코", 1500, 1)
14
15 fishbread1.one_more()
16 fishbread2.one_more()
```

반복적인 코드를 줄일 수 있음

같은 틀을 통해 쉽게 관리할 수 있음

코드의 재사용성을 극대화

클래스?

class 안에 구현된 함수: **method** <-

```
1 class Fishbread:
2     def __init__(self, flavor, price, amount):
3         self.flavor = flavor
4         self.price = price
5         self.amount = amount
6
7     def one_more(self):
8         self.price += self.price
9         self.amount += 1
10
11 fishbread1 = Fishbread("팥", 1000, 1)
12 fishbread2 = Fishbread("슈크림", 1200, 1)
13 fishbread3 = Fishbread("초코", 1500, 1)
14
15 fishbread1.one_more()
16 fishbread2.one_more()
```

-> 매개변수 **self**는 생성되는 객체를 말함
나머지는 매개변수는
실제 전달되는 값에 해당

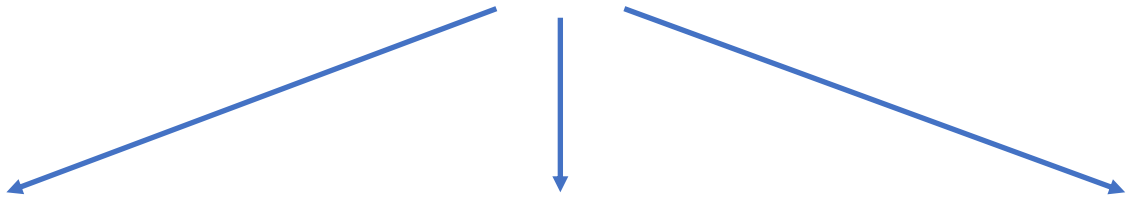
클래스명은 대문자로 시작하기를 권장!

생성자 `__init__()`: 객체가 생성될 때 자동으로 호출되는 메서드

메서드를 정의할 때는 첫 매개변수에 `self`가 포함되어 있음

클래스?

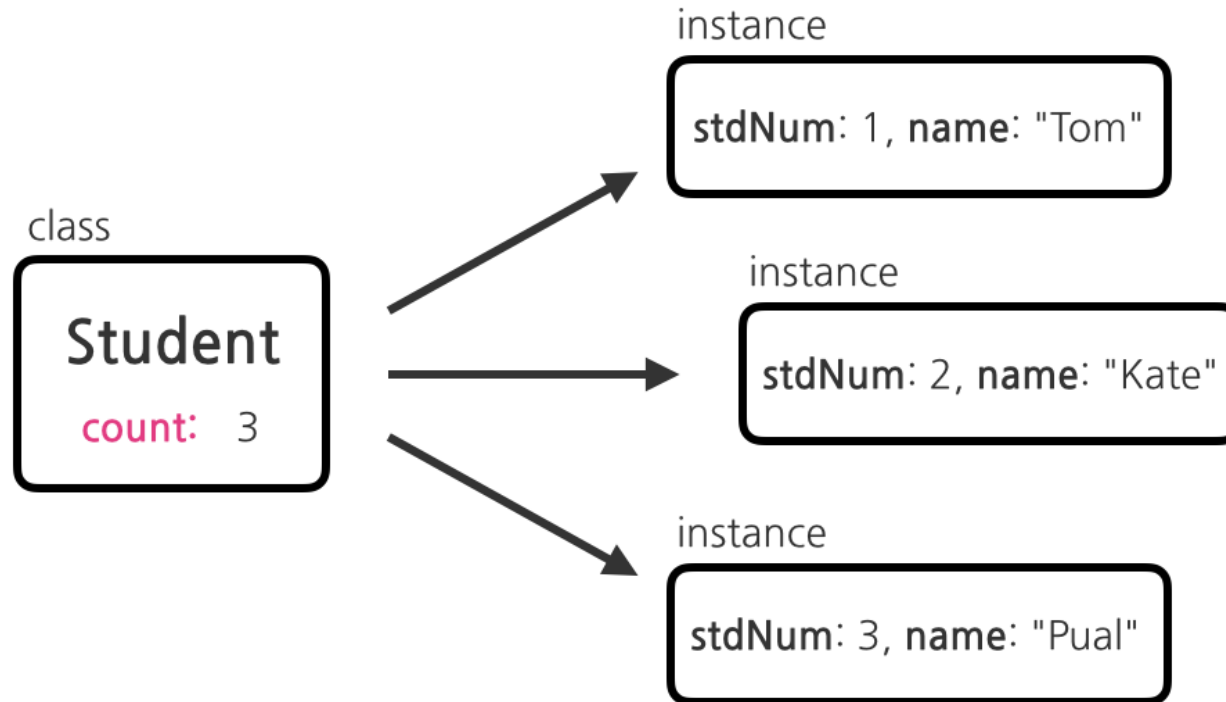
Class 라는 설계도



fishbread1		fishbread2		fishbread3	
Flavor	팥	Flavor	슈크림	Flavor	초코
price	1000	price	1200	price	1500
Amount	1	Amount	1	Amount	1

각각을
객체(인스턴스)
라고 함

클래스 변수와 인스턴스 변수



클래스 변수와 인스턴스 변수

```
[74] class Student:
    count = 0
    def __init__(self, std_num, name, age):
        self.std_num = std_num
        self.name = name
        self.age = age
        Student.count += 1

    def yell(self):
        print(f"나는 {self.std_num}번 {self.age}살 {self.name}이다!")

print(Student.count)
std1 = Student(1, "정석민", 20)
print(Student.count)
std2 = Student(2, "박경빈", 21)
print(Student.count)
std3 = Student(3, "강동혁", 23)
print(Student.count)
print()

std1.yell()
```

```
0
1
2
3
```

나는 1번 20살 정석민이다!

클래스 변수

- 특정 클래스의 모든 인스턴스 간 공유되는 변수

인스턴스 변수

- 각 인스턴스에 고유한 변수

클래스끼리 겹치는 비효율



```
1 class Korea:
2     def __init__(self, name, lang, food):
3         self.name = name
4         self.lang = lang
5         self.food = food
6
7     def yell(self):
8         print(f"우리 {self.name}의 기술력은 세계 최강이라고!")
9
10    def say(self):
11        print(f"{self.name}입니다.")
12
13 class Japan:
14     def __init__(self, name, lang, mount):
15         self.name = name
16         self.lang = lang
17         self.mount = mount
18
19     def yell(self):
20         print(f"우리 {self.name}의 기술력은 세계 최강이라고!")
21
22     def say(self):
23         print(f"{self.name}입니다.")
```

클래스의 상속

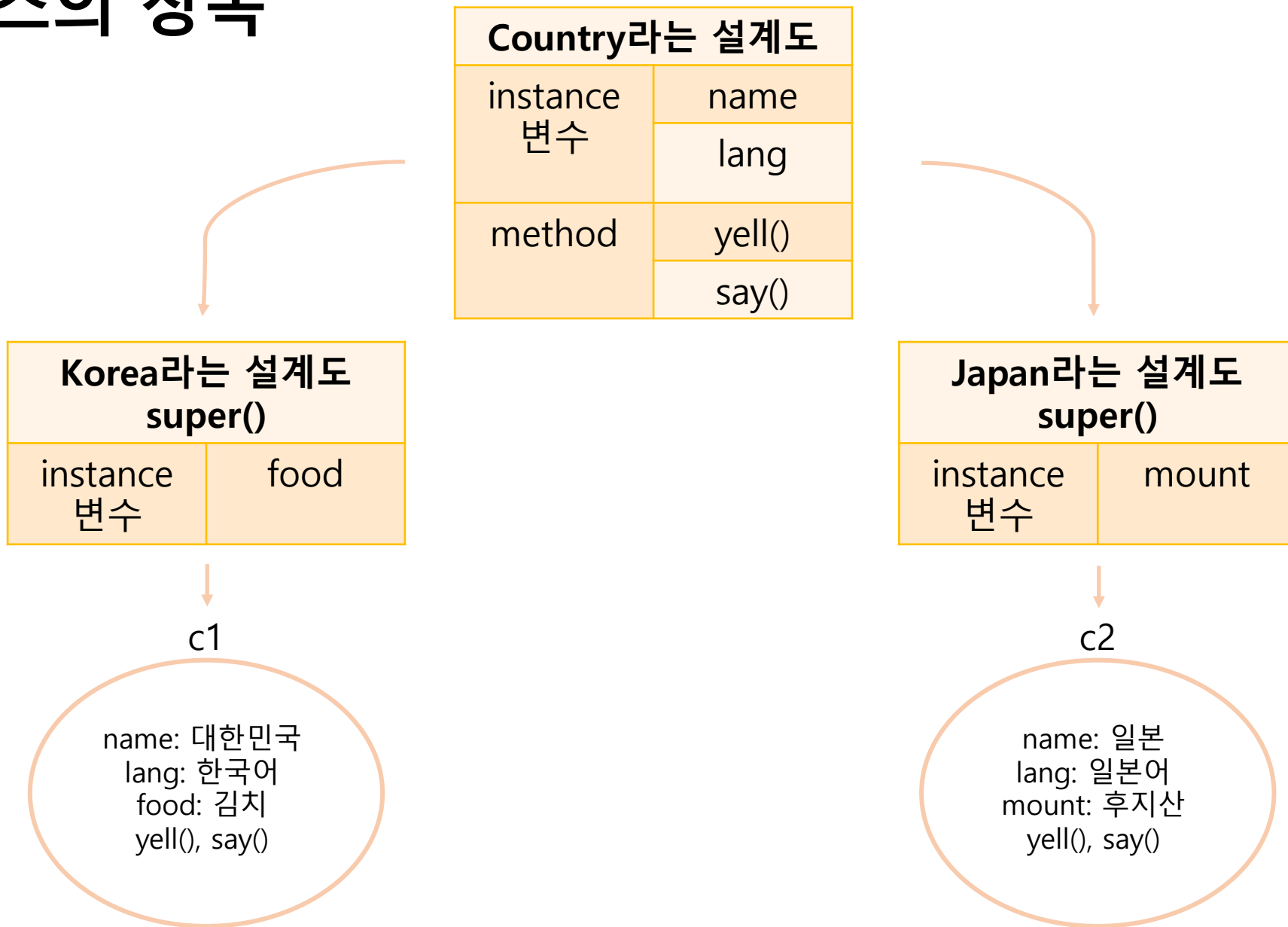
```
1 class Country:
2     def __init__(self, name, lang):
3         self.name = name
4         self.lang = lang
5
6     def yell(self):
7         print(f"우리 {self.name}의 기술력은 세계 최강이라고!")
8
9     def say(self):
10        print(f"{self.name}입니다.")
11
12 class Korea(Country):
13     def __init__(self, name, lang, food):
14         super().__init__(name, lang)
15         self.food = food
16
17 class Japan(Country):
18     def __init__(self, name, lang, mount):
19         super().__init__(name, lang)
20         self.mount = mount
21
22 c1 = Korea("대한민국", "한국어", "김치")
23 c2 = Japan("일본", "일본어", "후지산")
24
25 c1.yell()
26 c2.yell()
```

우리 대한민국의 기술력은 세계 최강이라고!
우리 일본의 기술력은 세계 최강이라고!

→ `super().__init__()` 을 통해서 다른 클래스의 속성과 메소드를 불러온다!!

→ 기존 클래스의 인스턴스 변수 이외에 새로운 인스턴스 변수를 선언할 수 있다!!

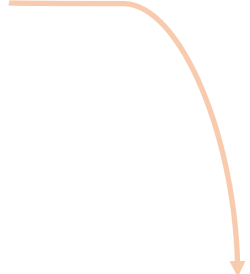
클래스의 상속



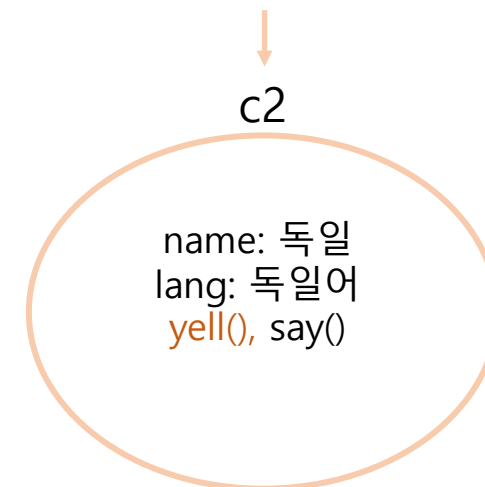
오버라이딩

```
1 class Germany(Country):
2     def __init__(self, name, lang):
3         super().__init__(name, lang)
4
5     def yell(self):
6         print(f"우리 {self.name}의 기술력은 우주 최강이라고!")
7
8 c3 = Germany("독일", "독일어")
9 c3.yell()
```

우리 독일의 기술력은 우주 최강이라고!



Germany라는 설계도 super()	
method	yell()



오버라이딩: 부모 클래스에 있는 메서드를 동일한 이름으로 다시 만드는 것!

Class 실습!

다음 조건에 맞게 실습을 진행해 주세요.

1. Person이라는 클래스가 있습니다. 이 클래스는 name, age, height 라는 인스턴스 변수를 가지고 있고, introduce(), yell()이라는 메서드를 가지고 있습니다. introduce()라는 메서드를 호출하면 자신의 이름과 나이를 말합니다. yell()이라는 메서드를 실행하면 '아?' 라고 말합니다.
2. Developer이라는 클래스가 있습니다. 이 클래스는 Person클래스를 상속합니다. Developer클래스는 yell()을 실행하면 '어?'라고 말합니다. 이 클래스는 keyboard = '기계식' 이라는 클래스 변수를 가지고 있습니다.
3. Designer이라는 클래스가 있습니다. 이 클래스는 Person클래스를 상속합니다. Designer클래스는 disease라는 인스턴스 변수를 추가적으로 가지고 있습니다.
4. ProductManager이라는 클래스가 있습니다. 이 클래스는 Person클래스를 상속합니다. ProductManager클래스는 yell()을 실행시키면 '개발자님 여기 오류있어요'라고 말합니다.
5. 클래스를 다 설계한 후 d1 이라는 Developer인스턴스, d2 라는 Designer인스턴스, p1 이라는 ProductManager 인스턴스를 생성한 후, 각 인스턴스의 introduce()와 yell()을 실행시켜보세요.

인스턴스 변수의 값은 아무거나 해도 상관없습니다. 원하는 값으로 넣어주세요

| 대표적인 OJ 사이트

BAEKJOON>
ONLINE JUDGE

 **programmers**

| +선택사항. 이미 잘하시는 분들을 위한 문제

구현 : 로봇 청소기 (골드 5)

<https://www.acmicpc.net/problem/14503>

백트래킹: N-Queen (골드 4)

<https://www.acmicpc.net/problem/9663>

그리디: 멀티탭 스케줄링 (골드 1)

<https://www.acmicpc.net/problem/1700>

구현: 치킨 배달 (골드 5)

<https://www.acmicpc.net/problem/15686>

<https://battered-pineapple-b0d.notion.site/2024-3-18-e7fa44ccb2404b4bac13f98de3e0aad5?pvs=4>

과제 (3월 25일 전까지)

여러분,, 힘들지만 지금 계속 해야 안 잊어버립니다...

