

파이썬 익숙해지기

Session 3

NEXT X LIKELION 전성운

지난 과제 리뷰

The image is a collage of screenshots from different social media platforms, likely Instagram and Qooover, showing user profiles and posts.

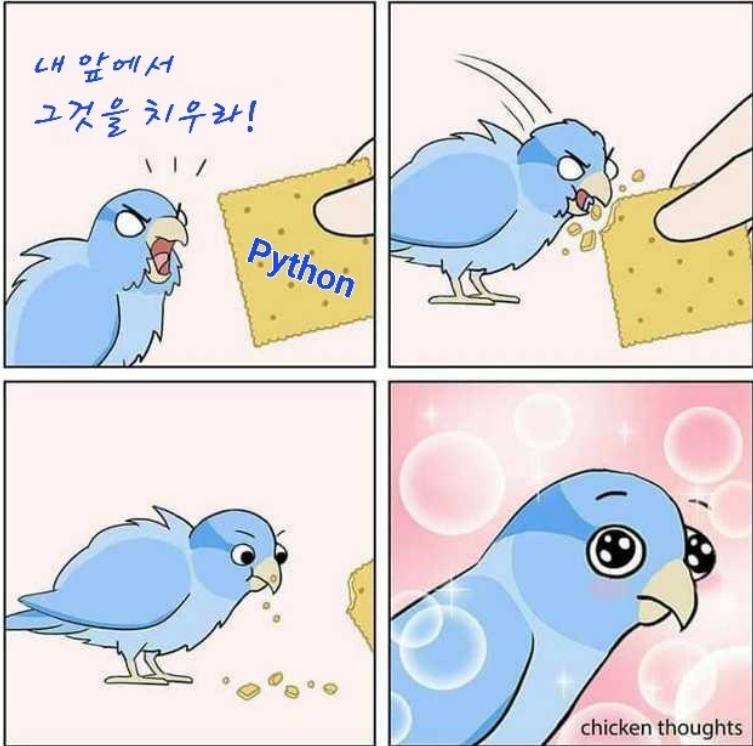
- Top Left:** A screenshot of a user profile for **iu_leejieun516**. It shows a profile picture of IU, her name, a bio mentioning she is a singer and actress, and links to her official Instagram and YouTube channels. It also shows her follower count (22.4 million) and posts (2242).
- Top Middle:** A screenshot of a user profile for **beige_seal**. It shows a profile picture of a person playing a guitar, her name, follower counts (14, 306, 292), and a bio about her being a student at Korea Univ. ELED 18, Dongsan CH. 21.
- Top Right:** A screenshot of a user profile for **qoover_copin**. It shows a profile picture of a cartoon character, her name, follower counts (347, 26K, 69), and a bio about being a doodler. It also shows a grid of her drawings and a message interface.
- Middle Left:** A screenshot of a user profile for **"LILAC"**, which is IU's 5th album. It shows a profile picture of IU, the album title, and a bio. It also shows her follower count (22.4 million) and posts (2242).
- Middle Center:** A screenshot of a user profile for **homin_joo**. It shows a profile picture of a cartoon character, her name, a bio, and a contact email. It also shows her follower count (22.4 million) and posts (2242).
- Middle Right:** A screenshot of a user profile for **wjstiddns02**. It shows a profile picture of a cat, her name, and a bio. It also shows a follower count (1935) and posts (8).
- Bottom Left:** A grid of various user posts, including photos of people, a car, and two kittens.
- Bottom Center:** A screenshot of a user profile for **culwoo**. It shows a profile picture of a cat, her name, and a bio. It also shows a follower count (1935) and posts (8).
- Bottom Right:** A screenshot of a user profile for **jeong._.min02**. It shows a profile picture of a cat, her name, and a bio. It also shows a follower count (14) and posts (8).

목차

1. 잡다한 파이썬 이야기..
2. 추가적인 파이썬 문법들
3. 클래스



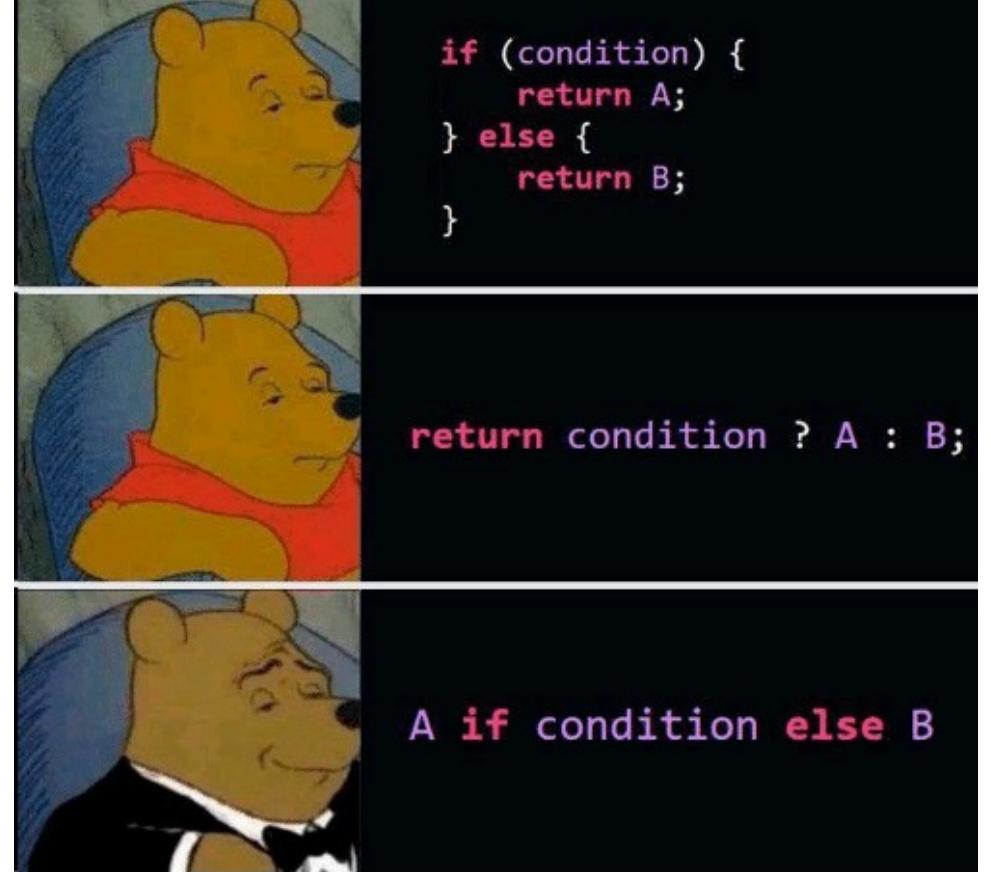
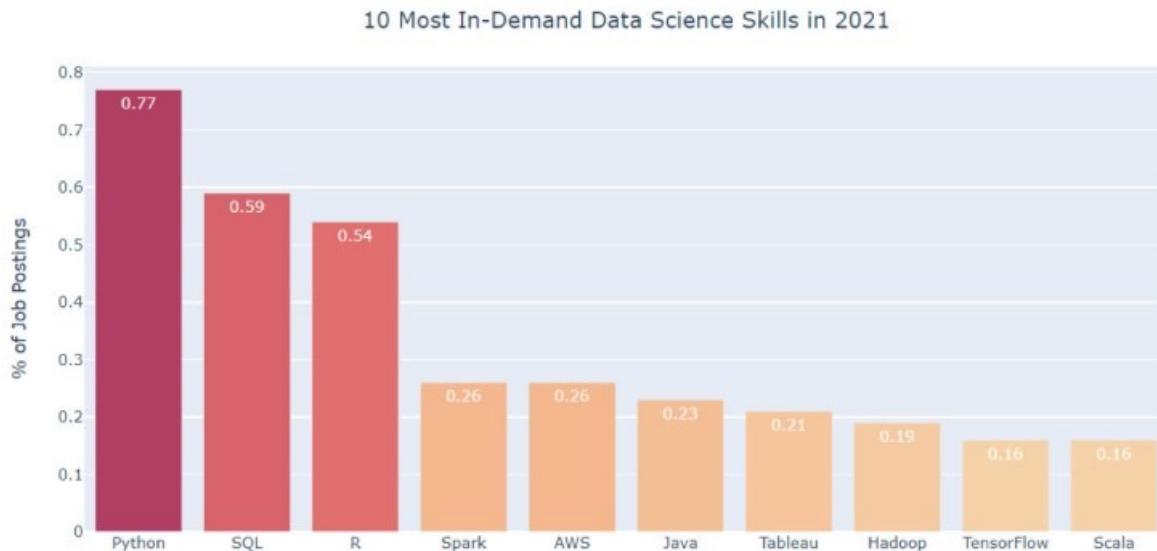
파이썬?



**When you switch
from C++ to Python**



파이썬!

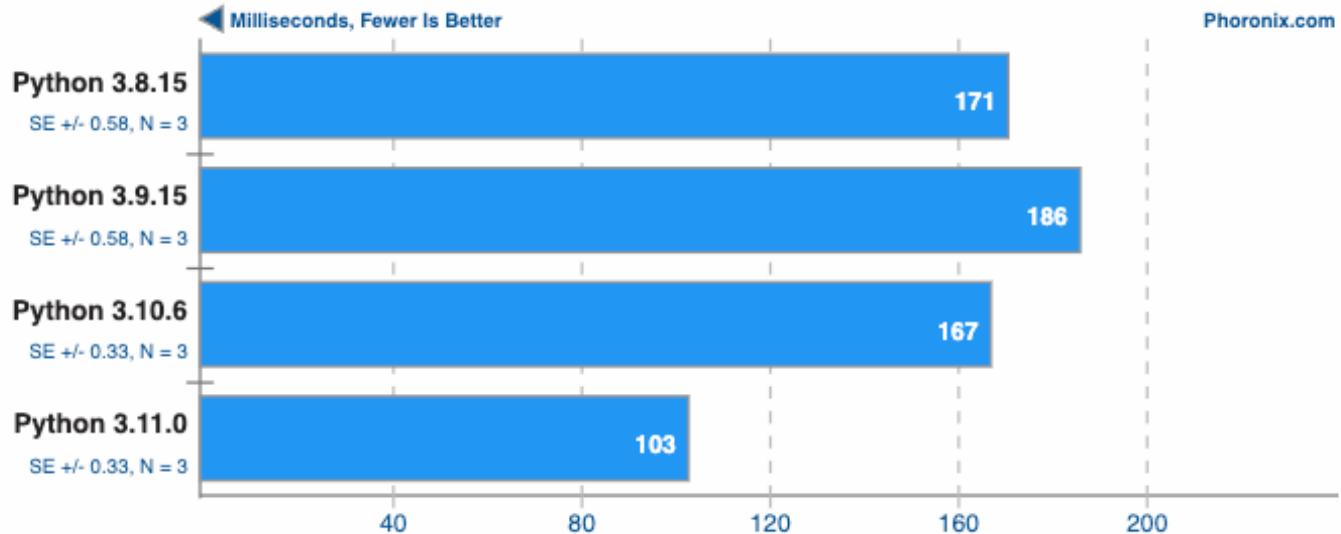


Only Python programmers will understand 😂

속도?

PyPerformance 1.0.0

Benchmark: go



| 파이썬!



| 많이 쓸 개념들..

1. 딕셔너리(Dictionary)

2. 함수

3. 배열

4. 슬라이싱

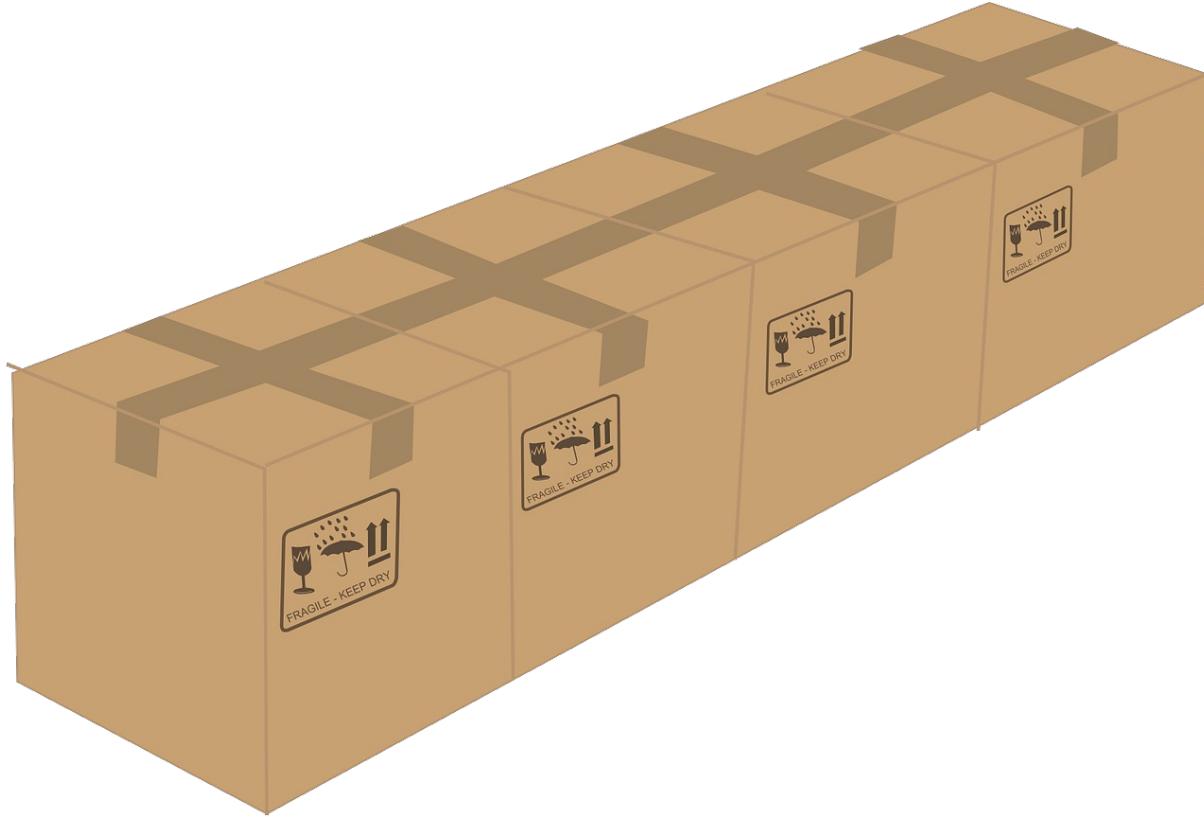
| 대표적인 OJ 사이트

BAEKJOON
ONLINE JUDGE



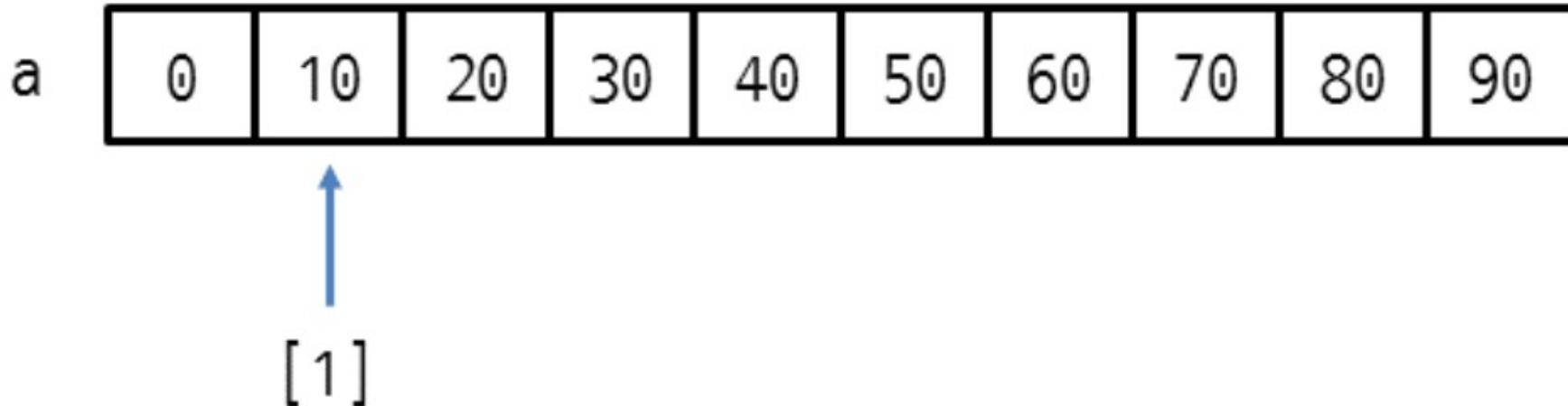
백준 회원가입 해주세요!

| 3. 배열(+문자열)



여러 데이터를 한 번에 관리하고 싶다..!

| 3. 배열(+문자열)



| 3. 배열(+문자열)



3. 배열(+문자열)

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

| 3. 배열(+문자열)

<https://www.acmicpc.net/problem/1152>

<https://www.acmicpc.net/problem/1259>

3. 배열(+문자열)

단어의 개수

성공



2 브론즈 //

| 시간 제한 | 메모리 제한 | 제출 | 정답 | 맞힌 사람 | 정답 비율 |
|-------|--------|--------|-------|-------|---------|
| 2 초 | 128 MB | 296812 | 93778 | 75044 | 32.361% |

문제

영어 대소문자와 공백으로 이루어진 문자열이 주어진다. 이 문자열에는 몇 개의 단어가 있을까? 이를 구하는 프로그램을 작성하시오. 단, 한 단어가 여러 번 등장하면 등장한 횟수만큼 모두 세어야 한다.

입력

첫 줄에 영어 대소문자와 공백으로 이루어진 문자열이 주어진다. 이 문자열의 길이는 1,000,000을 넘지 않는다. 단어는 공백 한 개로 구분되며, 공백이 연속해서 나오는 경우는 없다. 또 한 문자열은 공백으로 시작하거나 끝날 수 있다.

출력

첫째 줄에 단어의 개수를 출력한다.

3. 배열(+문자열)

팰린드롬수

성공

다국어



한국어

1 브론즈 |

| 시간 제한 | 메모리 제한 | 제출 | 정답 | 맞힌 사람 | 정답 비율 |
|-------|--------|-------|-------|-------|---------|
| 1 초 | 128 MB | 41404 | 23787 | 21003 | 57.880% |

문제

어떤 단어를 뒤에서부터 읽어도 똑같다면 그 단어를 팰린드롬이라고 한다. 'radar', 'sees'는 팰린드롬이다.

수도 팰린드롬으로 취급할 수 있다. 수의 숫자들을 뒤에서부터 읽어도 같다면 그 수는 팰린드롬수다. 121, 12421 등은 팰린드롬수다. 123, 1231은 뒤에서부터 읽으면 다르므로 팰린드롬수가 아니다. 또한 10도 팰린드롬수가 아닌데, 앞에 무의미한 0이 올 수 있다면 010이 되어 팰린드롬수로 취급할 수도 있지만, 특별히 이번 문제에서는 무의미한 0이 앞에 올 수 없다고 하자.

입력

입력은 여러 개의 테스트 케이스로 이루어져 있으며, 각 줄마다 1 이상 99999 이하의 정수가 주어진다. 입력의 마지막 줄에는 0이 주어지며, 이 줄은 문제에 포함되지 않는다.

출력

각 줄마다 주어진 수가 팰린드롬수면 'yes', 아니면 'no'를 출력한다.

| 4. 슬라이싱



| 4. 슬라이싱

<https://www.acmicpc.net/problem/9093>

4. 슬라이싱

단어 뒤집기 성공 다국어

☆ 한국어 ▾

① 브론즈 |

| 시간 제한 | 메모리 제한 | 제출 | 정답 | 맞힌 사람 | 정답 비율 |
|-------|--------|-------|-------|-------|---------|
| 1 초 | 128 MB | 29598 | 15496 | 11617 | 53.172% |

문제

문장이 주어졌을 때, 단어를 모두 뒤집어서 출력하는 프로그램을 작성하시오. 단, 단어의 순서는 바꿀 수 없다. 단어는 영어 알파벳으로만 이루어져 있다.

입력

첫째 줄에 테스트 케이스의 개수 T 가 주어진다. 각 테스트 케이스는 한 줄로 이루어져 있으며, 문장이 하나 주어진다. 단어의 길이는 최대 20, 문장의 길이는 최대 1000이다. 단어와 단어 사이에는 공백이 하나 있다.

출력

각 테스트 케이스에 대해서, 입력으로 주어진 문장의 단어를 모두 뒤집어 출력한다.

1. 딕셔너리(Dictionary)



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

| 1. 딕셔너리(Dictionary)



1. 딕셔너리(Dictionary)

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

| 쉬는시간



함수

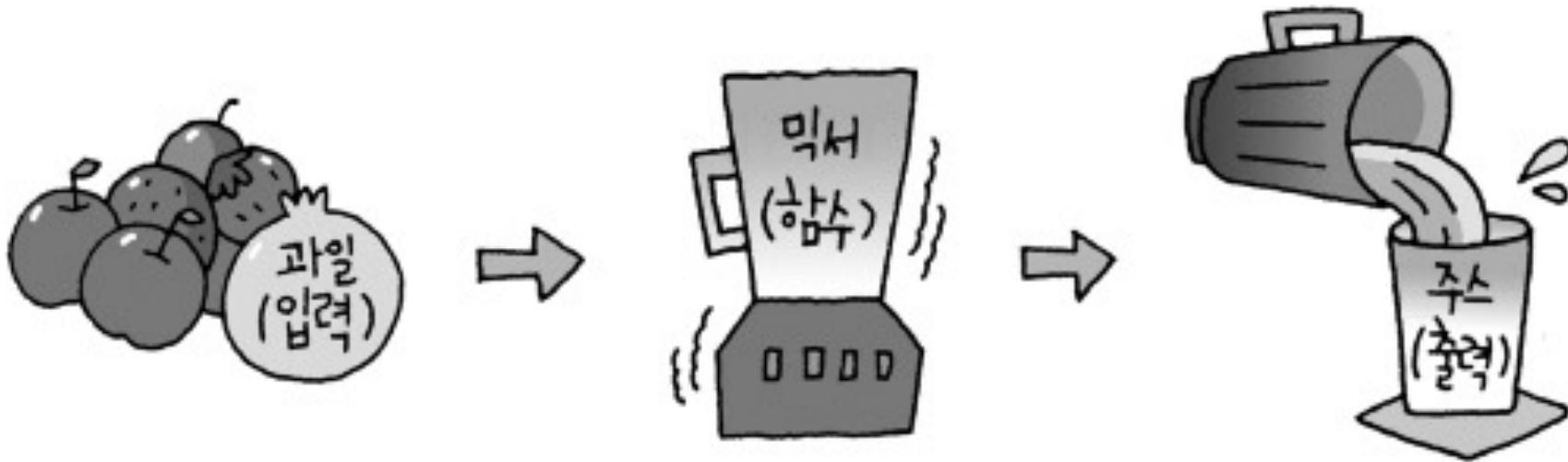
The screenshot shows a code editor window with a dark theme. On the left, there is a vertical toolbar with a green checkmark icon and a play button icon. The status bar at the bottom displays the number '0초' (0 seconds). The main area contains the following Python code:

```
1 n1, n2, n3, n4 = 3, 5, 9, 54
2
3 a = n1+n2
4 print(a)
5
6 a = n3+n4
7 print(a)
```

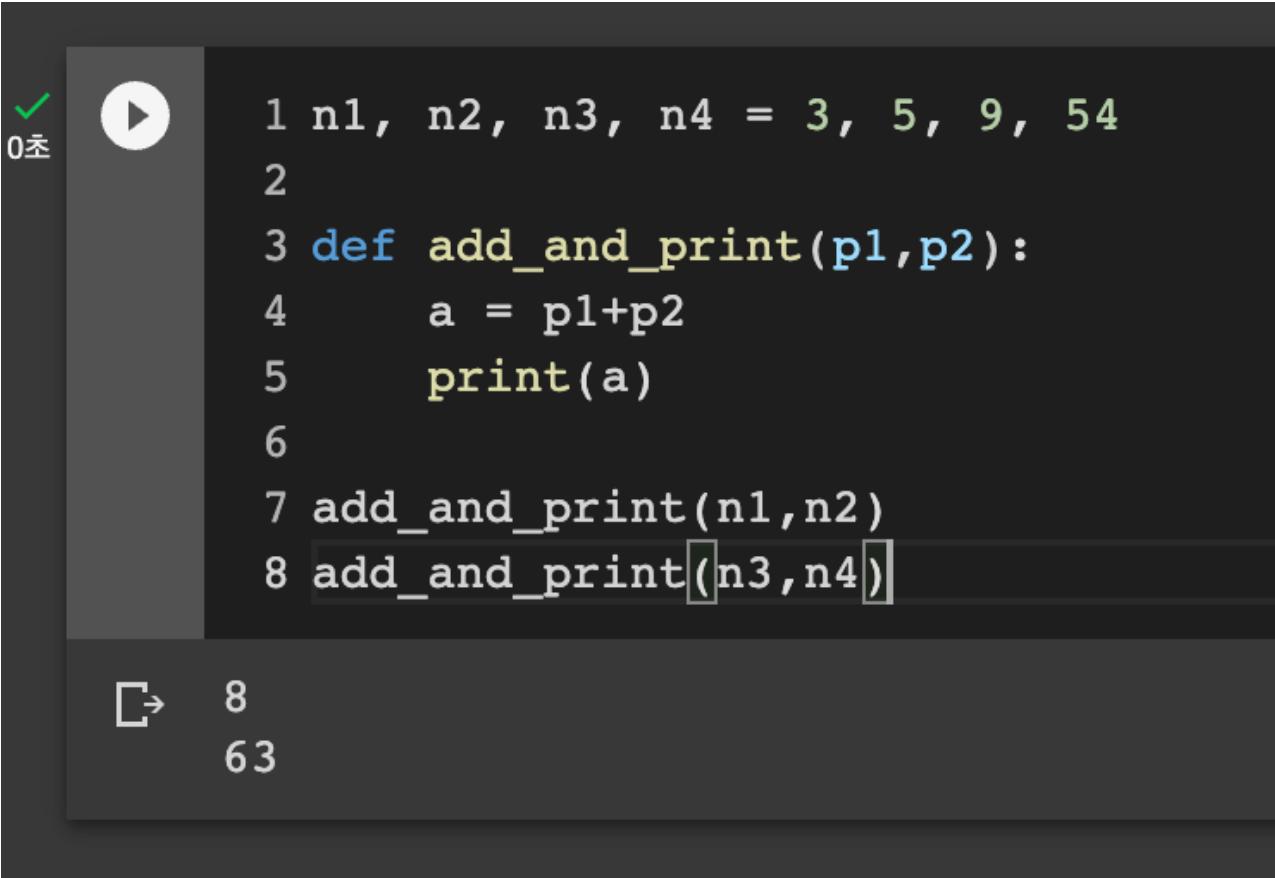
Below the code, the output is displayed in two lines:

```
8
63
```

| 함수



함수



The screenshot shows a code editor window with a dark theme. In the top-left corner, there is a green checkmark icon and the text "0초". To its right is a play button icon. The main area contains the following Python code:

```
1 n1, n2, n3, n4 = 3, 5, 9, 54
2
3 def add_and_print(p1,p2):
4     a = p1+p2
5     print(a)
6
7 add_and_print(n1,n2)
8 add_and_print([n3,n4])
```

Below the code, the output is displayed in a light gray box:

```
8
63
```

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
```



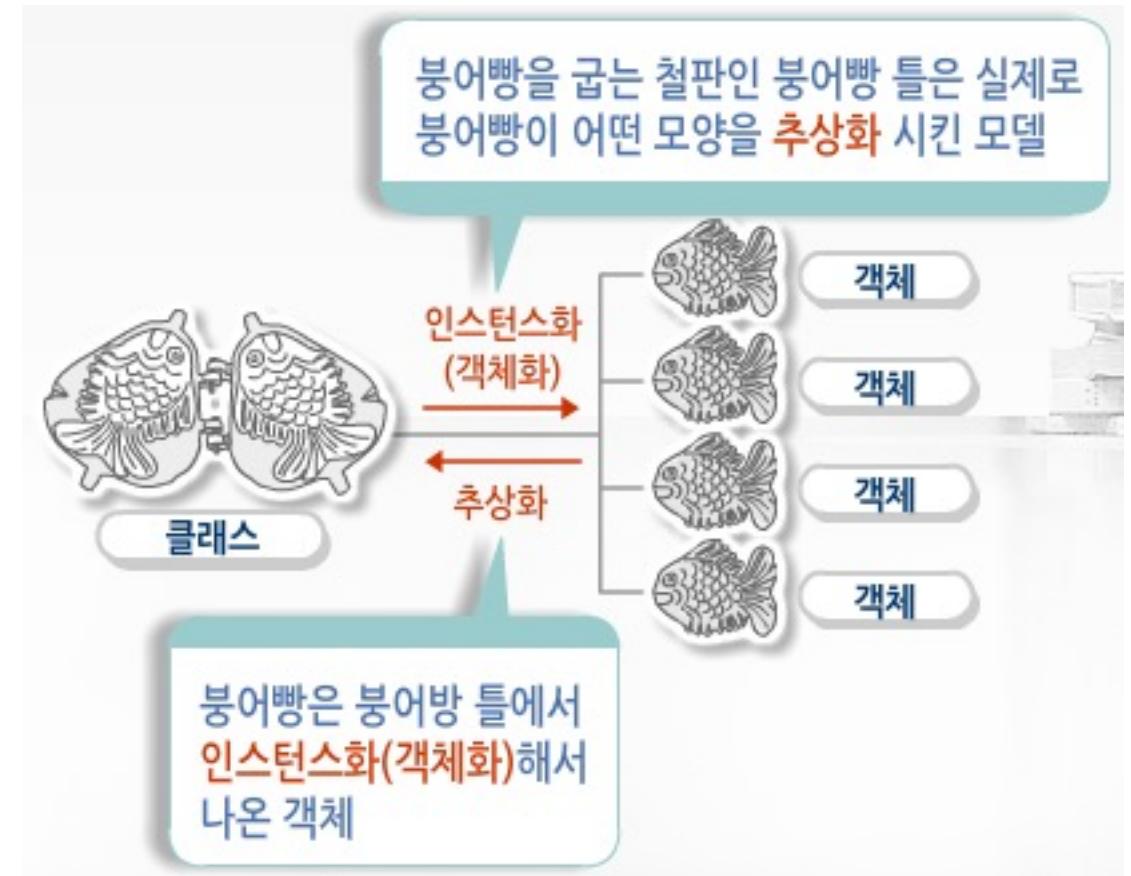
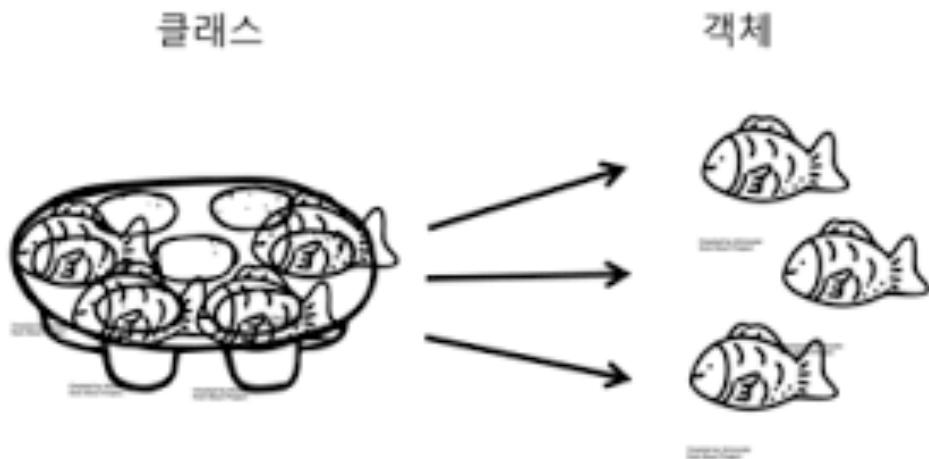
```
1 class Fishbread:  
2     def __init__(self, flavor, price, amount):  
3         self.flavor = flavor  
4         self.price = price  
5         self.amount = amount  
6  
7     def one_more(self):  
8         self.price += self.price  
9         self.amount += 1  
10  
11 fishbread1 = Fishbread("팥", 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?



클래스?

```
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()
```

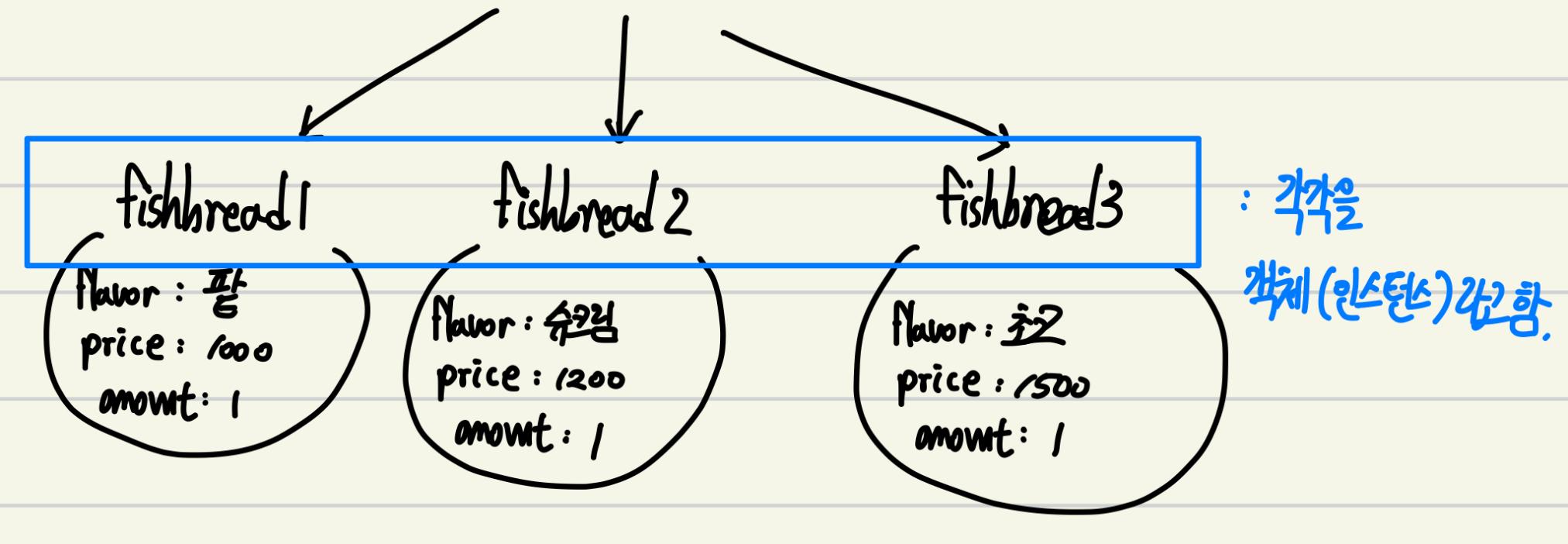
클래스 명은 항상 대문자로 시작!

생성자는 `_init_()`

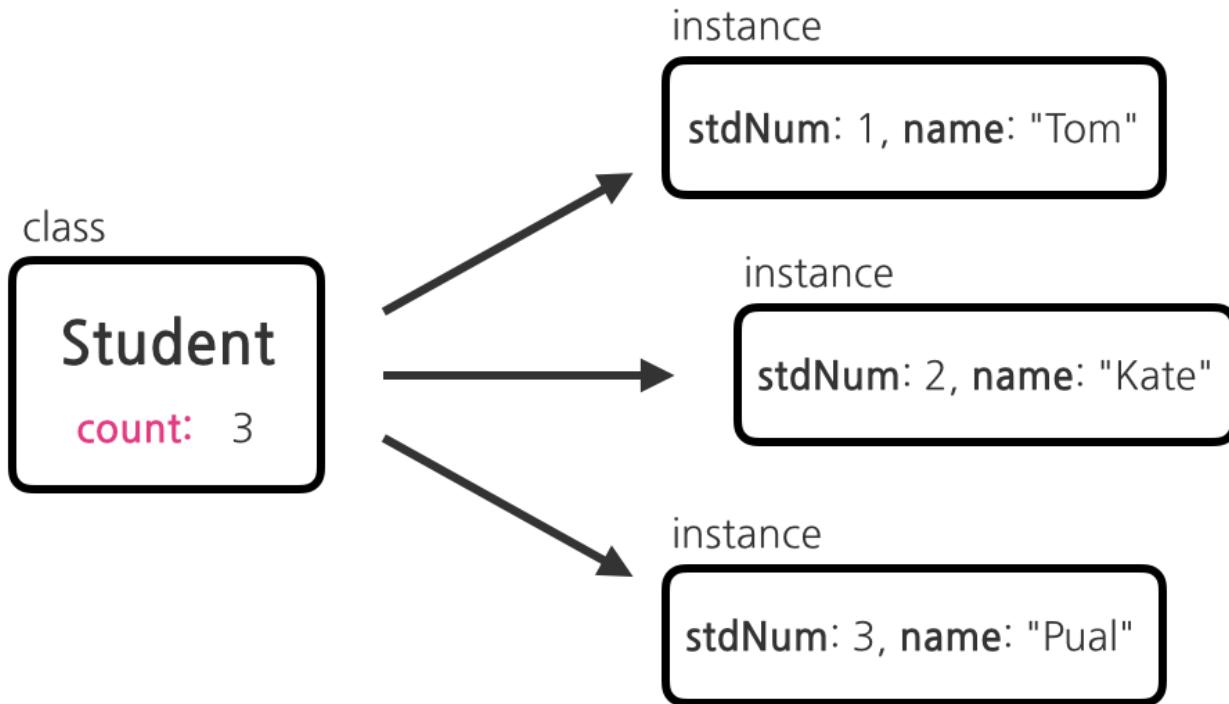
메서드를 정의할 때는 첫 매개변수에 `self`가 포함되어 있음

클래스?

class 라는 설계도.



클래스 변수와 인스턴스 변수



클래스 변수와 인스턴스 변수

```
1 class Student:
2     count = 0
3     def __init__(self, std_num, name, age):
4         self.std_num = std_num
5         self.name = name
6         self.age = age
7         Student.count += 1
8     def yell(self):
9         print(f"나는 {self.std_num}번 {self.age}살 {self.name}이다!")
10
11 print(Student.count)
12 st1 = Student(1, "전성운", 22)
13 print(Student.count)
14 st2 = Student(2, "김지성", 23)
15 print(Student.count)
16 st3 = Student(3, "김경진", 21)
17 print(Student.count)
18 print()
19
20 st1.yell()
```

```
0  
1  
2  
3
```

```
나는 1번 22살 전성운이다!
```

클래스끼리 겹치는 비효율

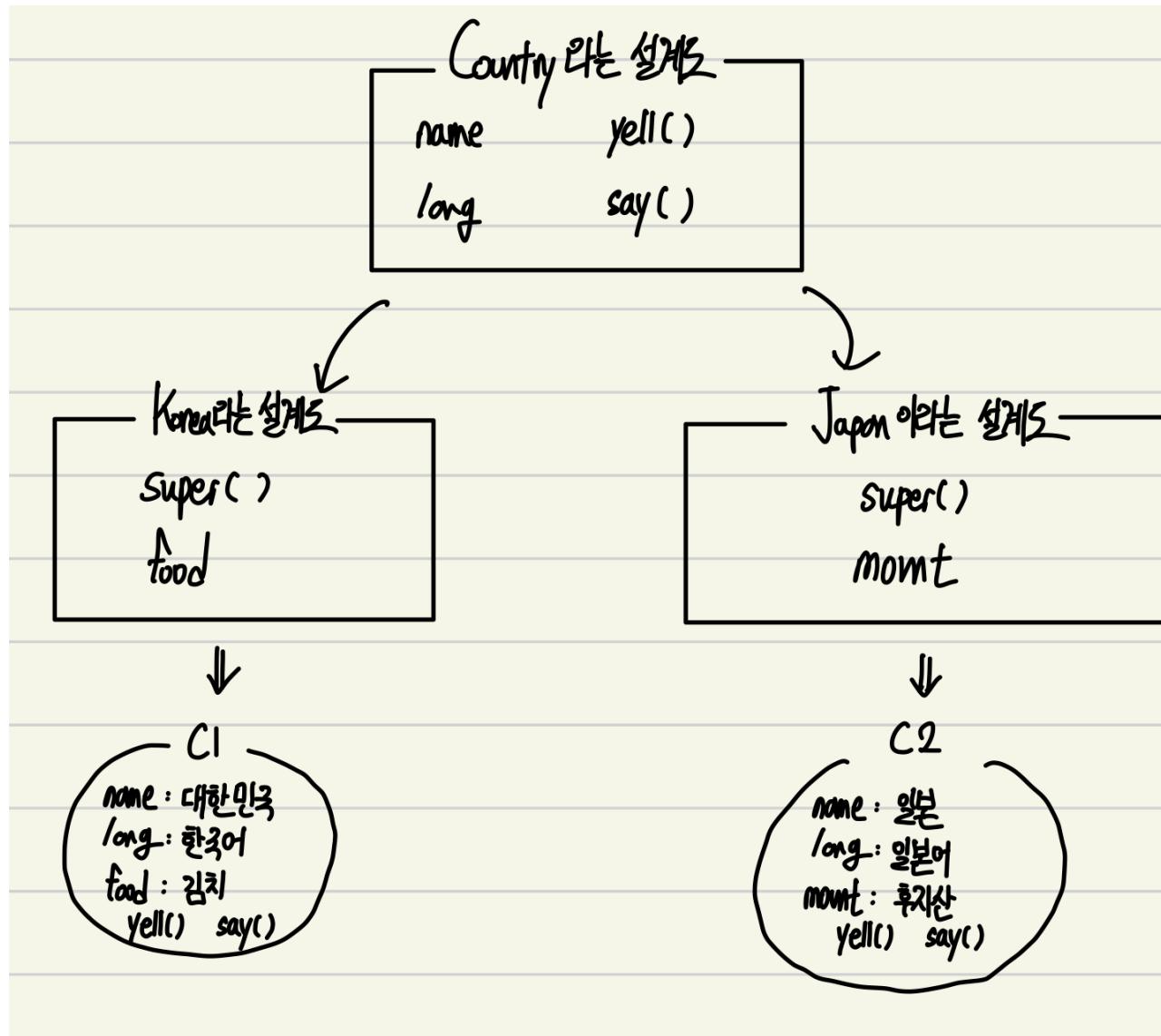
```
1 class Korea:
2     def __init__(self, name, lang, food):
3         self.name = name
4         self.lang = lang
5         self.food = food
6
7     def yell(self):
8         print(f"우리 {self.name}의 기술력은 세계 최강이라고!")
9
10    def say(self):
11        print(f"{self.name}입니다.")
12
13 class Japan:
14 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

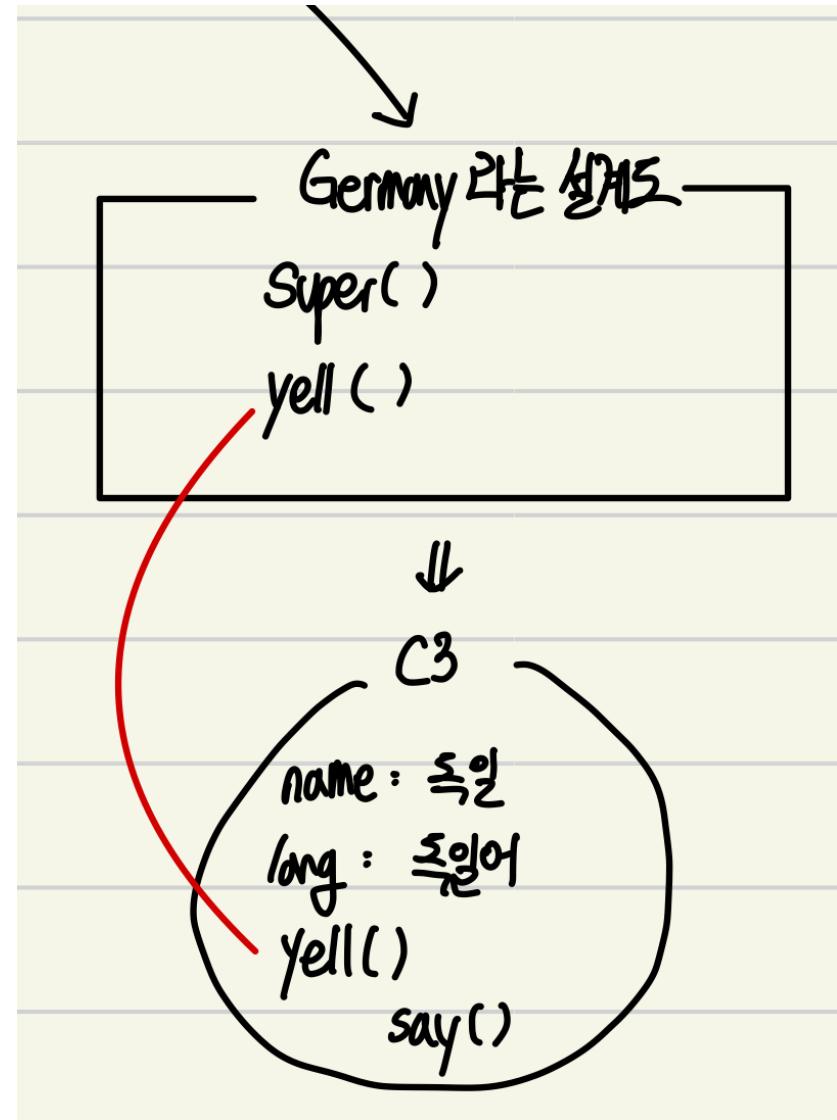
| 클래스의 상속



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!



비공개 속성

```
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, money):
14         super().__init__(name, lang)
15         self.food = food
16         self.__money = money
17
18 k1 = Korea('대한민국', '한국어', '김치', 1000)
19 print(k1.name)
20 print(k1.__money)
```

대한민국

```
-----
AttributeError                                     Traceback (most recent call last)
<ipython-input-2-47ec8ab019d9> in <module>
      18 k1 = Korea('대한민국', '한국어', '김치', 1000)
      19 print(k1.name)
--> 20 print(k1.__money)

AttributeError: 'Korea' object has no attribute '__money'
```

SEARCH STACK OVERFLOW

비공개 속성

```
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, money):
14         super().__init__(name, lang)
15         self.food = food
16         self.__money = money
17
18     def get_money(self):
19         return self.__money
20
21     def set_money(self, money):
22         self.__money = money
23         return self.__money
24
25 k1 = Korea('대한민국', '한국어', '김치', 1000)
26 print(k1.name)
27 print(k1.get_money())
28 print(k1.set_money(2000))
```

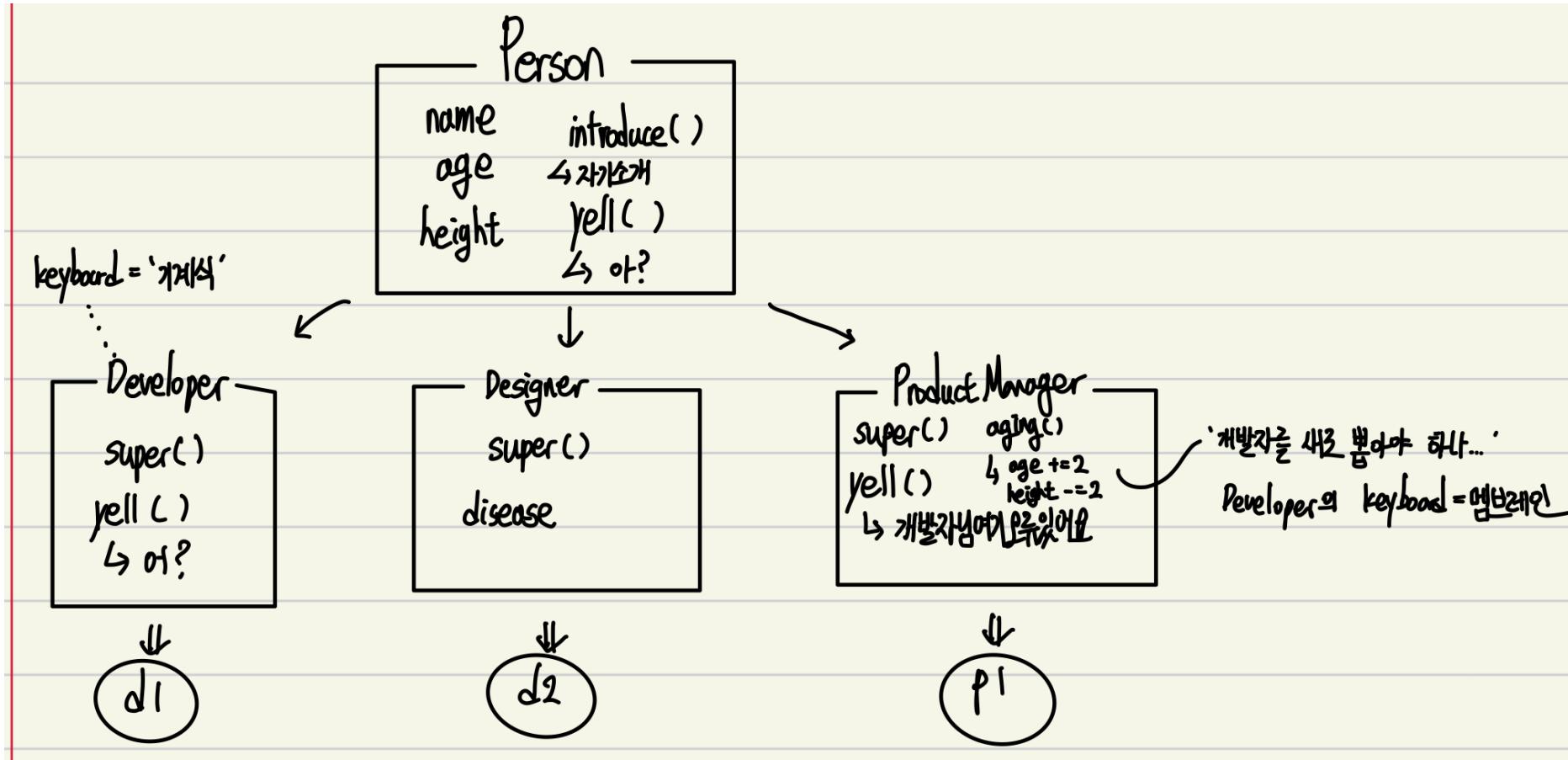
```
대한민국
1000
2000
```

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 나이와 키를 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. 이 클래스는 추가적으로 aging()이라는 메서드를 가지고 있으며, aging()메서드를 실행시 나이가 2살 늘어나고, 키가 5cm 줄어듭니다. 그리고 '개발자를 새로 뽑아야하나...'라고 말하고 Developer의 keyboard 변수를 '멤브레인'으로 바꿉니다.
5. 클래스를 다 설계한 후 d1 이라는 Developer인스턴스, d2 라는 Designer인스턴스, p1 이라는 ProductManager 인스턴스를 생성한 후, 각 인스턴스의 introduce()와 yell()을 실행시켜보세요. 그리고 그 후, p1의 aging()을 실행한 후, introduce()를 다시 실행해주세요.

Class 실습!



인스턴스 변수의 값은 아무거나 해도 상관없습니다. 원하는 값으로 넣어주세요

+ 절대하지마세요. 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>

과제 (3월 23일 전까지)

ㅎㅎ 죄송합니다.

