

Fastcampus Web Programming SCHOOL

Python

List, Tuple

List

```
animals = [' ', ' ', ' ']
```

Tuple

```
animals = (' ', ' ', ' ')
```

List

빈 list를 선언합니다. 선언과 동시에 값을 채워넣을 수 있습니다.

```
lang = ["python", "c", "java", "golang"]  
lang = []
```

list에 요소를 추가합니다.

```
lang.append("python")  
lang.append("java")  
lang.append("golang")  
print(lang)
```

혹은 특정한 위치에 원하는 값을 추가할 수 있습니다.

```
lang.insert(1, "c")
```

```
print(lang)
```

특정 요소를 삭제할 수도 있습니다.

```
lang.remove("golang")
```

```
print(lang)
```

혹은 리스트에 있던 값을 빼낼 수도 있습니다.

```
java = lang.pop(2)
```

```
print(lang)
```

```
print(java)
```

리스트를 정렬하는 법을 알아봅니다.

```
numbers = [2, 1, 4, 3]  
print(numbers)
```

```
numbers.sort()  
print(numbers)
```

리스트를 역순으로 출력하고 싶을땐 이렇게 한답니다.

```
numbers = [2, 1, 4, 3]  
numbers.reverse()  
print(numbers)
```

리스트를 내림차순으로 정렬하려면??

1. sort -> reverse

```
numbers.sort()  
numbers.reverse()
```

2. sort(reverse=True)

```
numbers.sort(reverse=True)
```

특정 값의 위치를 출력할땐 이렇게 합니다.

```
index_of_two = numbers.index(2)  
print(index_of_two)
```

리스트끼리 더할 땐 extend를 활용합니다.

```
numbers += [5, 6]  
print(numbers)  
numbers.extend([7, 8])  
print(numbers)
```

Membership Operator

```
in - 'cat' in ['fox', 'dog', 'cat'] => True
```

```
not in - 'wolf' not in ['fox', 'dog', 'cat'] => True
```

Tuple

Tuple은 괄호를 이용해 선언할 수 있습니다.

```
tuple1 = (1, 2, 3, 4)
```

tuple은 삭제나 추가가 불가능합니다.

```
del tuple[1]  
tuple1[1] = 'c'
```

tuple끼리 더하거나 반복하는 것은 가능합니다.

```
tuple2 = (5, 6)
```

```
print(tuple1 + tuple2)
```

```
print(tuple1 * 3)
```

tuple은 값을 편하게 바꿀 수 있습니다.

```
x = y  
y = x (x)  
  
temp = x  
x = y  
y = temp  
  
(x,y) = (y,x)
```

혹은 함수에서 하나 이상의 값을 반환할 때 사용합니다.

```
def quot_and_rem(x,y):  
    quot = x // y  
    rem = x % y  
    return (quot, rem)  
  
(quot, rem) = quot_and_rem(3,10)
```

List <-> Tuple

```
list((1,2))  
tuple([1,2])
```

Practice(1)

list와 tuple을 이용하여 다음의 문제를 해결하세요.

1. 변수 `animals` 에 다음의 자료를 입력하세요.
`rabbit, cat, dog, aligator, tiger, lion`
2. `animals`에 새로운 동물을 3마리 추가하세요.
3. `animals`를 알파벳 내림차순으로 정렬하세요.
4. 정렬된 `animals`의 짝수(index 기준) 동물을 `even_animals`에 할당하세요.

Dictionary, Set

Dictionary

- 매팅 탑입의 끕음자료형
- hash table을 이용하여 구현 -> 일정한 속도 -> 키의 수가 많아도 일정한 탐색속도를 유지



->



- hash:
 - 하나의 문자열을 보다 빨리 찾을 수 있도록 주소에 직접 접근할 수 있는 짧은 길이의 값이나 키로 변환

List, Tuple과의 차이

- index가 존재하지 않는다.
- key와 value로 이루어져있고, key로 value를 얻을 수 있다.

dictionary의 선언

```
dict1 = {}
```

```
print(dict1)
```

```
dict2 = {'ja':0, 'chuk':1, 'in':2}
```

```
dict3 = dict(ja=0, chuk=1, in=2)
```

```
dict4 = dict([('ja',0), ('chuk',1),('in',2)])
```

dictionary는 key와 value로 이루어져 있으며, 추가하는 법은 다음과 같습니다.

```
dict1 = {'name': 'foo bar'}  
print(dict1)
```

```
dict1 = {'korean': 95, 'math': 100, 'science': [80, 70, 90, 60]}  
print(dict1)
```

```
dict1['english'] = "pass"  
print(dict1)
```

요소 삭제는 del을 활용합니다.

```
del dict1['math']  
print(dict1)
```

key를 활용해 **value**를 출력하는 법을 알아봅시다.

```
print(dict1['korean'])
```

key만 출력하는 법을 알아봅시다.

```
print(dict1.keys())
```

value만 출력할땐 이렇게 합니다.

```
print(dict1.values())
```

key와 **value**를 함께 출력합니다.

```
print(dict1.items())
```

pop을 할 수도 있습니다

```
dict1.pop('english')
```

key, value 모두 pop 할 경우엔

```
dict1.popitem('science')
```

값을 얻으려 했지만..

```
>>> dict1['english']
```

```
KeyError: 'english'
```

get()

```
>>> dict1.get('english', 0)
```

```
>>> dict1.get('english')
```

setdefault()

```
>>> dict1.setdefault('english', [])
```

```
>>> dict1.get('english')
```

```
>>> dict1['english']
```

dictionary의 길이를 구할땐

```
len(dict1)
```

dictionary를 업데이트할 땐

```
dict1.update({'korean':40, 'math':50, 'ethics':60})
```

hash table

```
>>> hash_dict={}
```

index	key	value
0		
1		
2		
3		
4		
5		
-		

hash table

```
>>> hash_dict['ja']=0  
>>> hash('ja')  
287026653433178398  
>>> hash('ja')%8  
6
```

index	key	value
0		
1		
2		
3		
-		

hash table

```
>>> hash_dict['chuk']=1
>>> hash('chuk')
-2672060011921522239
>>> hash('chuk')%8
1
```

index	key	value
0		
1	'chuk'	1
2		
3		
-		

hash table

```
>>> hashed_dict['in']=2
>>> hash('in')
4800002792290664974
>>> hash('in')%8
6 ????????
```

index	key	value
0		
1	'chuk'	1
2		
3		
-		

hash table

index	key&value
0	
1	[('chuk',1)]
2	
3	
4	
5	
6	[('ja',0),('in',2)]
7	

dictionary with string format

```
contacts = {'name':'Guido', 'country':'Netherland'}  
'{name} is born in {country}'.format(**contacts)
```

Small Quiz

A = 'fastcampus'

B = 'python'

$A \cup B$

$A \cap B$

$A - B$

$A \Delta B$

Set

- 수학 집합 연산을 쉽게 하기 위해 만든 자료형
- 순서없음
- 중복없음

Set

Set 선언

```
ppap = {'pen', 'apple', 'pineapple', 'pen'}  
print(ppap)
```

```
'apple' in ppap  
'applepen' in ppap
```

```
pineapple = set('pineapple')  
pineapple
```

Set

```
A = set('fastcampus')
```

```
B = set('python')
```

```
A ∪ B == A | B
```

```
A ∩ B == A & B
```

```
A - B == A - B
```

```
A Δ B == A ^ B
```

Practice(2)

dictionary와 set을 이용하여 다음 문제를 해결하세요.

1. 다음 인적사항을 입력하세요.
(list에 각 정보가 dict로 존재)

Name	Locale	Age	Username
Johnny Silverhand	Night City	120	Johnny
John Doe	CA, USA	40	Doh
Jane Doe	Seoul, Korea	24	jane-doe
Foo Bar	Busan, Korea	31	foo-bar

2. Jane Doe의 Locale을 London, UK로 수정하세요.
3. list의 모든 아이템에 대해 key와 value를 모두 출력하세요.(반복문 쓰지 말 것)
4. 다음 list의 중복 아이템을 제거한 뒤, 내림차순으로 정렬하여 출력하세요.

```
cities = [  
    'Daejeon', 'Ulsan', 'Seoul', 'Jeju',  
    'Busan', 'Ulsan', 'Daegu', 'Daejeon',  
    'Seoul', 'Seoul', 'Daejeon', 'Gwangju',  
    'Busan', 'Daegu', 'Gwangju', 'Daejeon',  
    'Ulsan', 'Jeju', 'Gwangju', 'Seoul'  
]
```

Pseudocode

Pseudocode

== 의사코드 != Doctorcode

So, pseudocode is,

가짜코드: 프로그램이나 알고리즘이 수행할 내용을 인간이 이해할 수 있는 언어로 표현하는 것

Let's make fizzbuzz

fizzbuzz

1부터 n까지 반복하면서,

3의 배수는 "fizz"

5의 배수는 "buzz"

15의 배수는 "fizzbuzz"

나머지는 숫자

...

pseudocode는 프로그램을 설계할 때 밑그림의 역할을 하게 됩니다!

목적과 수행과정이 명확해 코드 수정과 분해가 편리합니다!

pseudocode가 comment의 역할을 수행할 수 있습니다.

How to write pseudocode?

nonstandard..

| 자신이 작성할 언어의 스타일에 맞춰 작성하면 끝!

I'm still hungry

1. hunger가 true가 됨
2. 돈이 없고, 현재위치가 집일때,
 1. 밥솥에 밥이 있다면, 해결한다.
 2. 굶는다.
3. 돈이 있고, 현재위치가 밖일때,
 1. 현금이 10만원 초과라면, 레스토랑을 간다.
 2. 현금이 10만원 이하라면, 편의점을 간다.

Let's make fizzbuzz again

fizzbuzz

1부터 n 까지 반복하면서,

3의 배수는 "fizz"

5의 배수는 "buzz"

15의 배수는 "fizzbuzz"

나머지는 숫자

| 영어로 작성하면 더 좋지만.. 한글로!

fizzbuzz

1. 사용자로부터 숫자 하나를 받아 n에 할당한다.
2. 1부터 n까지 숫자를 진행시키면서,
 3. 만약에 해당숫자가 15의 배수라면, "fizzbuzz"를 출력한다.
 4. 만약에 해당숫자가 3의 배수라면, "fizz"를 출력한다.
 5. 만약에 해당숫자가 5의 배수라면, "buzz"를 출력한다.
 6. 3,4,5의 경우를 만족하지 못한 경우, 해당숫자를 그대로 출력한다.

fizzbuzz-Kor

1. 사용자에게 정수 하나를 입력받아 num에 선언한다. i는 1임.
2. 1부터 num까지 반복하면서
3. 만약 그 수(i)가 3의 배수라면 "fizz"를 출력
4. 만약 그 수가 5의 배수라면 "buzz"를 출력
5. 만약 그 수가 15의 배수라면 "fizzbuzz"를 출력
6. 3,4,5의 경우를 만족하지 못한 나머지 경우 그 수를 출력

fizzbuzz - Eng

1. get integer from user ==> num, i == 1
2. WHILE i is less than or equal to num
3. if i is divisible by 3, print "fizz"
4. if i is divisible by 5, print "buzz"
5. if i is divisible by 15, print "fizzbuzz"
6. else, print i

fizzbuzz - python

```
num = int(input("get number for fizzbuzz: "))
i=1
while i <= num:
    if i % 3 == 0:
        print("fizz")
    elif i % 5 == 0:
        print("buzz")
    elif i % 15 == 0:
        print("fizzbuzz")
    else:
        print(i)
```

그럼, 부가가치세를 계산해봅시다.

VAT

- Korea: 10%
- Japan: 8%
- USA: 주마다 다름
- UK: 20%

제품 가격과 나라에 따라 다른 부가가치세를 계산해 그 가격을 보여주도록!

VAT - answer

1. get price of item ==> item_price
2. set tax rate (kor == 0.1, jap == 0.08, usa == "depend on state", uk == 0.2)
3. get country code(example: kor, jap, usa, uk) ==> country_code
4. tax_rate is matched price with country_code
5. sales tax is item_price times tax rate
5. total price is item_price plus sales tax