코딩테스트 Final 메서드 정리!

: Python

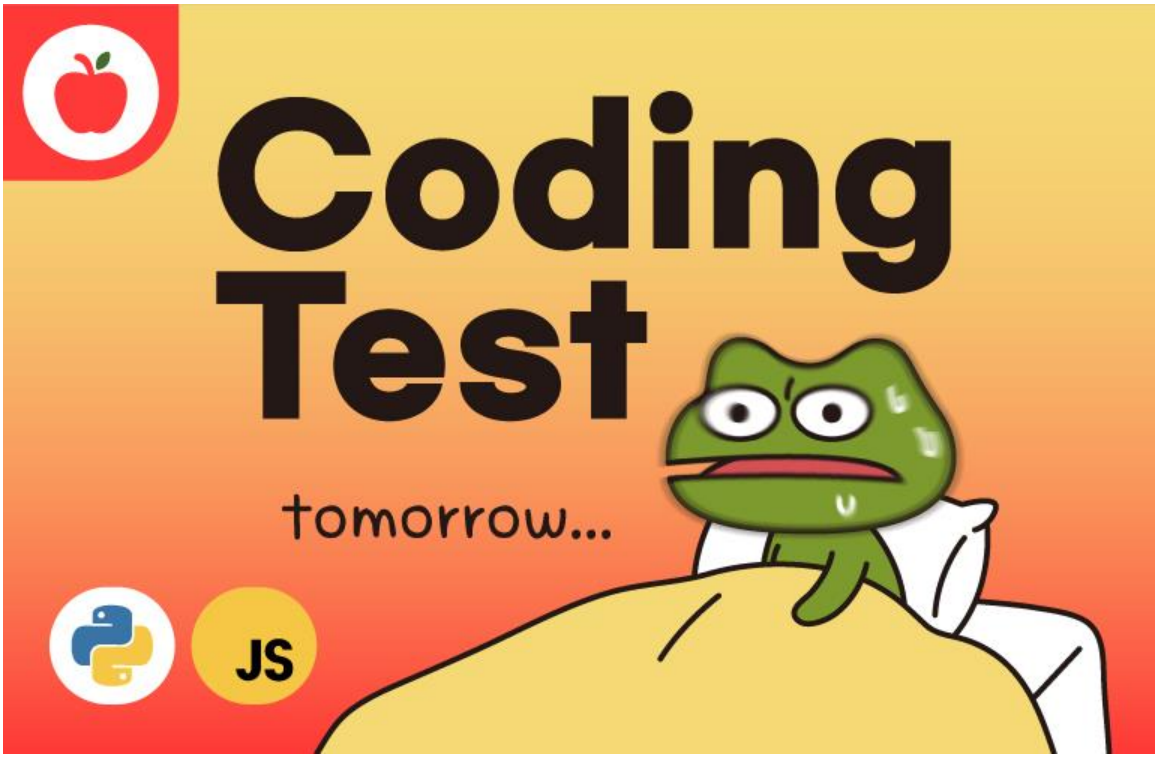
이호준 김대현 김유진 김혜원 이승신 이현창 장하림 차경림

이 책은 인쇄용으로 사용하시고,
아래 링크에서 노션(Notion)으로 된
Code Page를 보실 수 있으십니다.
실습을 하실 때에는 노션에서 Code를 복사해 사용하세요.

위니브즈와 함께하는 벼락치기 코딩 테스트

http://bitly.kr/L5iw8nC8X

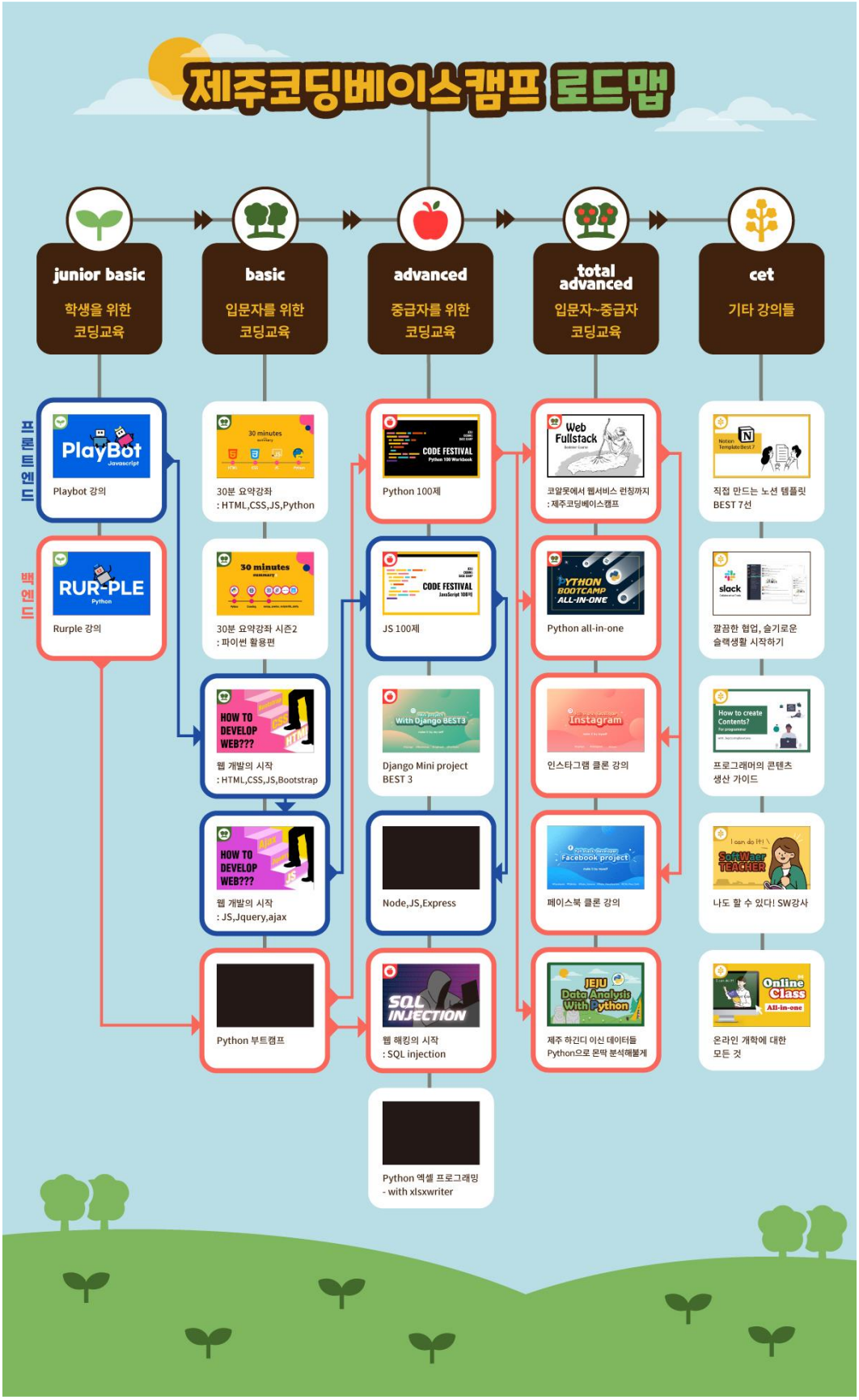
이 책에 있는 눈떠보니 코딩 테스트 전날! 강좌는 인프런에서 만나보실 수 있습니다.



눈떠보니 코딩 테스트 전날! 강좌

https://www.inflearn.com/course/코딩-테스트-전날

Road Map


제주코딩베이스캠프 로드맵
junior basic
학생을 위한 코딩교육
basic
입문자를 위한 코딩교육
advanced
중급자를 위한 코딩교육
total advanced
입문자~중급자 코딩교육
cet
기타 강의들
프론트엔드
백엔드
Playbot 강의
30분 요약강좌 : HTML,CSS,JS,Python
Python 100제
코알못에서 웹서비스 런칭까지 : 제주코딩베이스캠프
직접 만드는 노션 템플릿 BEST 7선
Rurple 강의
30분 요약강좌 시즌2 : 파이썬 활용편
JS 100제
Python all-in-one
깔끔한 협업, 슬기로운 슬랙생활 시작하기
웹 개발의 시작 : HTML,CSS,JS,Bootstrap
Django Mini project BEST 3
인스타그램 클론 강의
프로그래머의 콘텐츠 생산 가이드
웹 개발의 시작 : JS,Jquery,ajax
Node,JS,Express
페이스북 클론 강의
나도 할 수 있다! SW강사
Python 부트캠프
웹 해킹의 시작 : SQL injection
제주 하긴디 이신 데이터들 Python으로 몬딱 분석해볼게
온라인 개학에 대한 모든 것
Python 엑셀 프로그래밍 - with xlsxwriter

목차

Built-in Function

List

```python
# list
dir([1, 2, 3, 4])
```

Python

```
Out[-]
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__delitem__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__gt__',
 '__hash__',
 '__iadd__',
 '__imul__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mul__',
 '__ne__',
 '__new__',
```

```
'__reduce__',
'__reduce_ex__',
'__repr__',
'__reversed__',
'__rmul__',
'__setattr__',
'__setitem__',
'__sizeof__',
'__str__',
'__subclasshook__',
'append',
'clear',
'copy',
'count',
'extend',
'index',
'insert',
'pop',
'remove',
'reverse',
'sort']
```

Python

- append : 리스트 끝에 원소를 추가

```python
# append
x = [1,2,3,4]
x.append([3,4])
print(x)
```

Python

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

Python

- extend : 리스트 끝에 모든 원소 추가

```python
# extend
x = [1,2,3,4]
x.extend([3,4])
print(x)
```

Python

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

Python

- copy : 원본에 영향을 주지 않기 위해서 사용

```python
x = [1, 2, 3, 4]
y = x
y.extend([3,4])
print(x)
print(y)
```

Python

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

Python

```python
x = [1, 2, 3, 4]
y = x.copy()
y.extend([3,4])
print(x)
print(y)
```

Python

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

Python

- clear : 리스트 안의 원소 모두 삭제

```python
x = [1, 2, 3, 4, 5]
print(x)
x.clear()
print(x)
```

Python

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

Python

- count(a) : 리스트 안에 원소 a의 개수 반환

```python
x = [1, 6, 3, 6, 5, 6, 7]
x.count(6)
```

Python

```
Out[-]
3
```

Python

- index : 리스트 값의 위치를 반환

```python
x = [5, 8, 3]
x.index(8)
```

Python

```
Out[-]
1
```

Python

- insert (a,b): 리스트의 a위치에 b 원소 삽입

```python
x = [1, 7, 2, 8, 4]
x.insert(2,5)
x
```

Python

```
Out[-]
[1, 7, 5, 2, 8, 4]
```

Python

- pop : 리스트의 지정한 위치의 원소 삭제

```python
x = [0, 2, 4, 6, 8]
x.pop(1)
x
```

Python

```
Out[-]
[0, 4, 6, 8]
```

Python

- remove : 리스트에 들어있는 원소 중 지정한 원소 삭제

```python
x = [0, 2, 4, 6, 8]
x.remove(6)
x
```

Python

```
Out[-]
[0, 2, 4, 8]
```

Python

Format( )

- format(value, format_spec)
- 형식 지정자가 제어하는 주어진 값의 형식화 된 표현을 출력

```python
# 단위 : 원
format(10000000000, ',')
```

Python

```
Out[-]
'10,000,000,000'
```

Python

```python
# 지수
format(10000000000, 'e')
```

Python

```
Out[-]
'1.000000e+10'
```

Python

```python
# 16진수
format(10000000000, 'x')
# hex(10000000000) <- 16진수로 변환
```

Python

```
Out[-]
'2540be400'
```

Python

```
# 오른쪽 정렬
format(100000, '0>020,.4f')
# 왼쪽 정렬
format(100000, '0<020,.4f')
# 가운데 정렬
format(100000, '0=020,.4f')
```

Python

```
Out[-]
'00000000100,000.0000'
'100,000.000000000000'
'000,000,100,000.0000'
```

Python

Filter( )

- filter(function, iterable)
- iterable의 각 요소가 true인지 아닌지 확인후 출력

```python
def hojun(value):
    if value % 2 == 0:
        return True
    else:
        return False

list(filter(hojun, range(20)))
```

Python

```
Out[-]
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
```

Python

```python
list(filter(lambda x: x % 2 == 0, range(20)))
```

Python

```
Out[-]
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
```

Python

```python
[i for i in range(20) if i % 2 == 0]
```

Python

```
Out[-]
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
```

Python

```python
len([1, 2, 3, 4])
```

Python

```
Out[-]
4
```

Python

Map( )

- map(function, iterable, ...)
- 주어진 function를 iterable의 각 항목에 적용하고 결과 목록을 출력

```python
list(map(hojun, range(20)))
list(map(lambda x: x % 2 == 0, range(20)))
list(map(lambda x: x**2, range(20)))
list(map(lambda x: x**2, range(20)))
```

Python

```
Out[-]
[True, False, True, False, True, False, True, False, True, False, True, False, True, Fals
e, True, False, True, False, True, False]
[True, False, True, False, True, False, True, False, True, False, True, False, True, Fals
e, True, False, True, False, True, False]
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361]
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361]
```

Python

Zip( )

- zip(*iterables)
- iterables을 가져와 튜플로 집계하여 출력

```python
x = ['a','b','c']
y = [1,3,2]

z = list(zip(x,y))
print(z)
```

Python

```
[('a', 1), ('b', 3), ('c', 2)]
```

Python

```python
list(zip(['a', 'b', 'c', 'd'], [1, 2, 3, 4], [10, 20, 30, 40], 'ABCD'))
set(zip(['a', 'b', 'c', 'd'], [1, 2, 3, 4], [10, 20, 30, 40], 'ABCD'))
```

Python

```
Out[-]
[('a', 1, 10, 'A'), ('b', 2, 20, 'B'), ('c', 3, 30, 'C'), ('d', 4, 40, 'D')]
{('a', 1, 10, 'A'), ('b', 2, 20, 'B'), ('c', 3, 30, 'C'), ('d', 4, 40, 'D')}
```

Python

Max/Min( )

- max(iterable, *iterables, key, default)
- min(arg1, arg2, *args, key)
- iterable에서 가장 큰 항목을 반환합니다,
- min(iterable, *iterables, key, default)
- max(arg1, arg2, *args, key)
- iterable에서 가장 작은 항목을 반환한다

```python
max([1, 2, 3, 4])
min([1, 2, 3, 4])
```

Python

```
Out[-]
4
1
```

Python

```python
x = {8: 1, 3: 9, -2: 1, 10:-1}

result1 = max(x) # key값의 min
print(result1)

result2 = max(x, key = lambda k: x[k])
print(result2)   # value가 큰 key 값
print(x[result2])# key 값을 적용시킨 value
```

Python

```
Out[-]
10
3
9
```

Python

```python
x = {8: 1, 3: 9, -2: 1, 10:-1}

result1 = min(x) # key값의 min
print(result1)

result2 = min(x, key = lambda k: x[k])
print(result2)   # value가 가장 작은 key 값
print(x[result2])# key 값을 적용시킨 value
```

Python

```
Out[-]
-2
10
-1
```

Python

Sort( )

- sort : 리스트의 원본을 직접 정렬

```python
x = [10, 5, 8]
y = x.sort()

print(x)
print(y)
```

Python

```
Out[-]
[5, 8, 10]
None
```

Python

- sorted : 리스트의 원본은 그대로 보존하고 정렬된 값을 반환

```python
x = [10, 5, 8]
y = sorted(x)
print(x)
print(y)
```

Python

```
Out[-]
[10, 5, 8]
[5, 8, 10]
```

Python

```python
testCaseOne = ['abc', 'def', 'hello world', 'hello', 'python']
testCaseTwo = 'Life is too short, You need python'.split()
testCaseThree = list(zip('anvfe', [1, 2, 5, 4, 3]))

sorted(testCaseOne, key=len, reverse=True)
sorted(testCaseTwo, key=str.lower)
sorted(testCaseThree, key=lambda x:x[1])
sorted(testCaseThree, key=lambda x:x[0])
```

Python

```
Out[-]
['hello world', 'python', 'hello', 'abc', 'def']
['is', 'Life', 'need', 'python', 'short,', 'too', 'You']
[('a', 1), ('n', 2), ('e', 3), ('f', 4), ('v', 5)]
[('a', 1), ('e', 3), ('f', 4), ('n', 2), ('v', 5)]
```

Python

```python
5 not in [1, 2, 3, 4, 5]
```

Python

```
Out[-]
False
```

Python

Reversed( )

- reverse : 리스트에 순서를 거꾸로 뒤집기

```python
x = [10, 5, 8]
y = x.reverse()
print(x)
print(y)
```

Python

```
Out[-]
[8, 5, 10]
None
```

Python

- reversed : 리스트의 원본은 그대로 보존하고 순서를 거꾸로 뒤집기

```python
x = [10, 5, 8]
y = reversed(x)
print(list(x))
print(list(y))
```

Python

```
Out[-]
[10, 5, 8]
[8, 5, 10]
```

Python

```python
l = [1, 2, 3, 4, 5]
```

Python

```python
Out[-]
dir(l)
# 'append'
# 'clear'
# 'copy'
# 'count'
# 'extend'
# 'index'
# 'insert'
# 'pop'
# 'remove'
# 'reverse'
# 'sort'
def listChange(x):
    x[0] = 1000

l = [1, 2, 3, 4, 5]
listChange(l.copy())
l
```

Python

Quque

```python
l = []

l.append(10)
l.append(20)
l.append(30)
l.pop(0)
l.append()
```

Python

```python
#Queue 라이브러리를 이용한 방법
import queue
q = queue.Queue()
q.put(2) # append
q.put(5)
q.put(7)
q.put(8)
print(q.get()) # pop
print(q.get())
print(q.get())
print(q.get())
```

Python

```
Out[-]
2
5
7
8
```

Python

```python
#리스트 queue
class listq():
    def __init__(self):
        self.queue = []

    def push(self, n ):
        return self.queue.append(n)

    def pop(self):
        if len(self.queue) == 0:
            return -1
        return self.queue.pop(0)

    def printq(self):
        print(self.queue)

    def empty(self):
        if len(self.queue) == 0:
            return print('Yes')
        return print('No')

queue= listq()
queue.push(1)
queue.push(2)
queue.push(3)
queue.empty()
queue.printq()
print(queue.pop())
print(queue.pop())
print(queue.pop())
print(queue.pop())
```

Python

```
Out[-]
No
[1, 2, 3]
1
2
3
-1
```

Python

Stack

```python
#스택
l = []

l.append(10)
l.append(20)
l.append(30)
l.pop()
l.append()
```

Python

```python
# 리스트 stack
class list_s():
    def __init__(self):
        self.stack = []

    def push(self, n ):
        return self.stack.append(n)

    def pop(self):
        if len(self.stack) == 0:
            return -1
        return self.stack.pop()

    def printq(self):
        print(self.stack)

    def empty(self):
        if len(self.stack) == 0:
            return print('Yes')
        return print('No')
```

```python
stack= list_s()
stack.push(1)
stack.push(2)
stack.push(3)
stack.empty()
stack.printq()
print(stack.pop())
print(stack.pop())
print(stack.pop())
print(stack.pop())
```

Python

```
Out[-]
No
[1, 2, 3]
3
2
1
-1
```

Python

집합

```python
# tuple
t = (1, 2, 3)
dir(t)
```

Python

```
Out[-]
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getnewargs__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rmul__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'count',
 'index']
```

Python

```python
d = {'one':'하나', 'two':'둘'}
dir(d)
# 'clear',
# 'copy',
# 'fromkeys',
# 'get',
# 'items',
# 'keys',
# 'pop',
# 'popitem',
# 'setdefault',
# 'update',
# 'values'

d.keys()
d.values()
d.items()
```

Python

```
Out[-]
dict_keys(['one', 'two'])
dict_values(['하나', '둘'])
dict_items([('one', '하나'), ('two', '둘')])
```

Python

```python
dir(s)

# 'add',
# 'clear',
# 'copy',
# 'difference', : 차집합 집합하나.difference(집합둘)
# 'difference_update',
# 'discard',
# 'intersection',
# 'intersection_update',
# 'isdisjoint',
# 'issubset',
# 'issuperset',
# 'pop',
# 'remove',
# 'symmetric_difference',
# 'symmetric_difference_update',
# 'union', : 합집합
# 'update', : 한꺼번에 많은 데이터를 추가

s.add(7)
s.discard(7)
'1' in s
```

Python

```
Out[-]
{'1', '2', '3', '4', '5', '6', 7}
{'1', '2', '3', '4', '5', '6'}
True
```

Python

```python
판콜에이 = {'A', 'B', 'C'}
타이레놀 = {'A', 'B', 'D'}

print(판콜에이.difference(타이레놀)) #차집합
print(판콜에이.intersection(타이레놀)) #교집합
print(len(판콜에이.intersection(타이레놀))) #교집합
print(판콜에이.union(타이레놀)
```

Python

```
Out[-]
{'C'}
{'B', 'A'}
2
{'B', 'A', 'D', 'C']
```

Python

```python
# 단톡방에 x마리의 동물이 대화를 하고 있습니다.
# 각각의 동물들이 톡을 전송할 때마다 서버에는 아래와 같이 저장됩니다.
# 1. 단톡방에는 모두 몇 마리의 동물이 있을까요? 톡은 무조건 1회 이상 전송합니다.
# 2. 단톡방에 동물들마다 몇 번의 톡을 올렸을까요?

serverData = '개리 라이캣 개리 개리 라이캣 자바독 자바독 파이 썬'

len(set(serverData.split()))

d = {}
for i in set(serverData.split()):
    print(i, serverData.split().count(i))
    d[i] = serverData.split().count(i)
d
```

Python

```
Out[-]
라이캣 2
썬 1
파이 1
자바독 2
개리 3
```

Python

Set( )

- set(iterable)

```python
s = set('11122345666')
s
```

Python

```
Out[-]
{'1', '2', '3', '4', '5', '6'}
```

Python

Split( )

```python
for i in '1 2 3 4 5 6 7'.split():
    print(int(i))

[int(i) for i in '1 2 3 4 5 6 7'.split()]

list(map(int, '1 2 3 4 5 6 7'.split()))
```

Python

```
1
2
3
4
5
6
7
[1, 2, 3, 4, 5, 6, 7]
```

Python

```python
n = list(map(int, input().split()))
print(n)
```

Python

```
In[-]
2 1 4
Out[-]
[2, 1, 4]
```

Python

enumerate( )

- enumerate(iterable, start=0)
- iterable에 counter 추가하고 출력한다

```python
x = ['one','two','three']
for iterable in enumerate(x):
    print(iterable)
for counter , value in enumerate(x):
    print(counter, value)
```

Python

```
Out[-]
(0, 'one')
(1, 'two')
(2, 'three')
0 one
1 two
2 three
```

Python

```python
x = [10,20,30,40,50]

for i , j in enumerate(x,2):
    print(i , j)
```

Python

```
Out[-]
2 10
3 20
4 30
5 40
6 50
```

Python

Abs( )

- abs(num)
- 주어진 숫자의 절대 값을 출력

```python
abs(-1)
abs(0)
abs(-15)
```

Python

```
Out[-]
1
0
15
```

Python

초판 1쇄 발행 | 2020년 5월 4일
지은이 | 이호준 김대현 김유진 김혜원 이숭신 이현창 장하림 차경림
편 집 | 이호준
총 괄 | 이호준
펴낸곳 | 사도출판
주 소 | 제주특별자치도 제주시 동광로 137 대동빌딩 4층
표지디자인 | 차경림
홈페이지 | http://www.paullab.co.kr
E - mail | paul-lab@naver.com