Day 7 - Collection and Loop

1. 이터러블과 이터레이터

1. 이터러블(iterable)

a. 이터러블은 for문처럼 **반복문에서 사용할 수 있는 모든 객체**를 의미합니다. 이터러블한 객체는 for item in 객체: 구조에서 사용 가능합니다. 모든 순서 있는 컬렉션은 이터러블입니다. 하지만 모든 이터러블이 순서 있는 것은 아닙니다. (set, dict.keys()는 반복은 되지만 인덱스가 없음)

2. 이터레이터(Iterator)

a. 이터러블 객체는 iter() 함수를 사용해서 이터레이터로 바꿀 수 있습니다. 또한 next()를 사용해서 값을 하나씩 꺼낼 수 있습니다.

이터러블	iter()를 적용할 수 있는 객체	리스트, 튜플, 문자열 등
이터레이터	next()로 값을 꺼낼 수 있는 객체	iter(리스트)로 만든 것

리스트와 for문:

```
# 리스트에서 길이가 5이상인 문자열만 출력
words = ["apple", "hi", "banana", "go", "mango"]

for word in words:
    if [length=:= len(word)] >= 5:
        print(f'{word})(길이: {length})')
```

- len(word) >> 6
- 6이 length에 저장 >> 6 >=5 연산
- 프린트

딕셔너리와 for문:

```
dic1 = {'no':1, 'userid':'apple', 'name':'김사과', 'hp':'010-1111-1111'}

for i in dic1:
    print(i, end=' ') # 키만 복사

To userid name hp
```

- 딕셔너리 같은경우 for문으로 돌렸을때 키만 가져온다.

```
[21] for i in dic1:
print(dic1[i], end=' ') # 이런식으로 dic1이 키만 복사하는 성질을 이용해서 value를 가져온다.
```

- dic1.keys() >> 키만 뽑기
- dic1.values() >> 값만뽑기
 - dic1.get(i) >> 값만 뽑기

```
for i in dic1:
print(dic1.get(i), end= ' ') # get(i) 메서드를 이용해서 값을 가져온다.
```

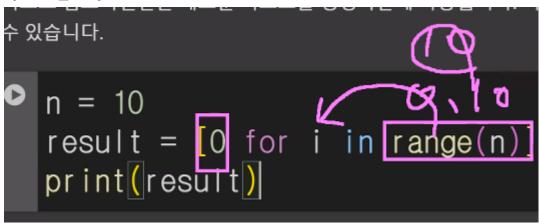
- dic1.items() >> 키, 값을 튜플형태로 뽑기

```
[25] for key, value in dic1.items():
print(key, value) # 키, 값을 따로 저장할 수 있다.
```

컴프리헨션(Comprehension)

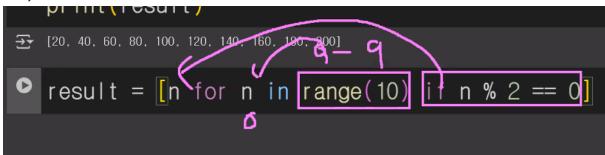
- 컴프리헨션(Comprehension)은 파이썬에서 리스트, 세트, 딕셔너리 등의 컬렉션을 간단하게 생성하거나 변형하는 방법 중 하나입니다. 컴프리헨션은 반복문과 조건문을 사용하여 간결하게 컬렉션을 생성하는 기법으로, 코드를 더 간단하고 가독성 좋게 작성할 수 있도록 도와줍니다.

리스트 컴프리헨션



- range(10) >> 10바퀴를 돈다
- 0 for i >> 10바퀴동안 0을 리턴시킨다.
- 그리고 [] 리스트 안에 넣는다
- 따라서 output: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

ex1)



- 0~9까지 뽑아서 리스트에 담는데 짝수만 담을것.
- [0, 2, 4, 6, 8]

ex2)

```
result = [n for n in range(10) if n % 2 == 0]
print(result)
[0, 2, 4, 6, 8]
```

- 오른쪽에 있는 if는 필터, 걸러서 꺼내는 기능
- 0~9까지 뽑아서 리스트에 담는데 짝수만 담을것.

ex3)

```
# 양수는 리스트에 그대로 저장하고, 음수는 0으로 변환해서 저장하기 li = [-1, 0, -4, 24, 5, -10, 2, 20] result = [n·if·n>0·else·0·for·n in li] #li에 담긴 내용을 하나씩 꺼내서 print(result)
```

- 왼쪽에 있는 if는 뭔가 조작해서 꺼내는 기능
- li에 담긴 내용을 하나씩 꺼내서 0보다 크면 리턴, 아니면 0을 리턴

ex4)

- 일반 for문을 사용한 풀이

```
li = [i*j for i in range(1, 4) for j in range(1, 3)]
print(li)

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

- List Comprehension 풀이

세트(Set) 컴프리헨션

- 세트 컴프리헨션은 새로운 세트를 생성하는데 사용됩니다. 기존 세트의 각 요소를 반복하면서 조건을 적용하여 새로운 세트를 생성할 수 있습니다.

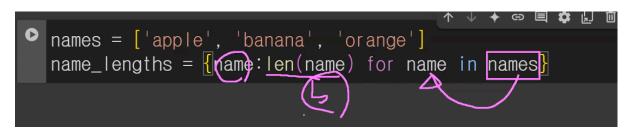
```
Ii = [1, 2, 3, 4, 5, 2, 3, 4]
unique_numbers = set(Ii)
print(unique_numbers)

Ii = [1, 2, 3, 4, 5, 2, 3, 4]
unique_numbers = {x for x in Ii} # List >> Set으로 변환함으로서 중복값을 제거한다.
print(unique_numbers)

1, 2, 3, 4, 5}
{1, 2, 3, 4, 5}
```

딕셔너리(Dictionary) 컴프리헨션

- 딕셔너리 컴프리헨션은 새로운 딕셔너리를 생성하는데 사용됩니다. 기존 딕셔너리의 키와 값을 반복하면서 조건을 적용하여 새로운 딕셔너리를 생성할 수 있습니다.



- names 리스트 안에 있는 내용을 하나씩 꺼낸다
- len(name)을 계산한다.
- name을 key, len(name)을 value로 저장한다.
- {} 딕셔너리로 만든다.