컴프리헨션과 제너레이터

컴프리헨션(Comprehension)

사전적 정의 : 이해력, (언어에 대한) 이해능력

파이썬에서의 컴프리헨션?

iterable 객체에서 특정 조건을 만족하는 객체를 보다 (이해하기) 쉽게 만들어 내기 위한 구문

```
new_list = [expression for item in iterable if condition
```

expression: 각 아이템에 적용할 표현식

for item in iterable: 반복할 아이템

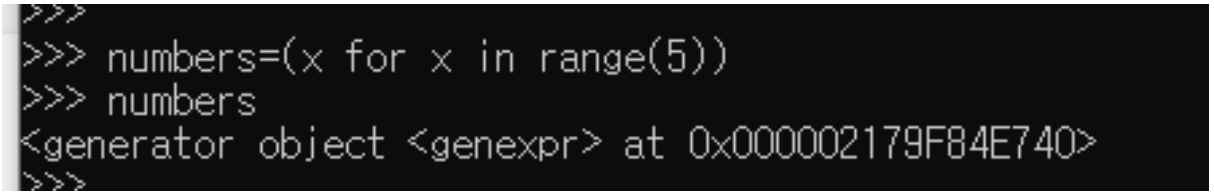
if condition: 조건이 참일 경우에만 아이템을 포함시킨다.

```
matrix = [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]

matrix_e = [[row[i] for row in matrix] for i in range(len(mat

print(matrix_e)
>>> [[1, 5, 9], [2, 6, 10], [3, 7, 11], [4, 8, 12]]
```

list→ `[]` 와 마찬가지로 딕셔너리, set과 같은 `{}` 모양에도 적용이 가능하다.

그렇다면, 튜플 컴프리헨션도 있을까?

기대했던 tuple 자료형



실제로 출력된 결과

→ 안된다...

제너레이터

제너레이터란 , 지연 평가 (lazy evaluation) 구현체이다

즉, 필요할때 값을 계산해주는 객체 이다.

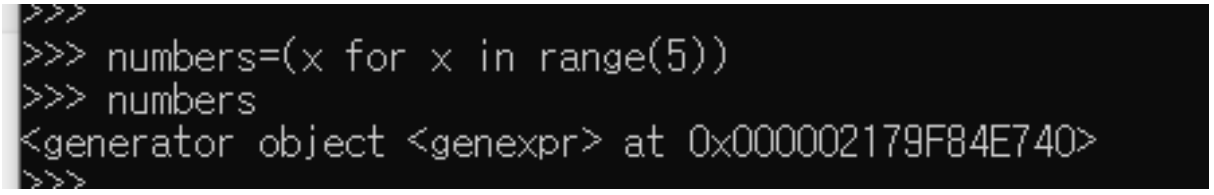
list,set,dictonary는 Container 객체이기 때문에 , 여러 객체들을 포함할 수 있다.

이때 ,포함하고 있는 객체들을 모두 평가되어진 상태(계산된 상태)로 가지고 있다.

그래서 list에 index로,set,dict에 아이템으로 접근이 가능할 수 있는 것!

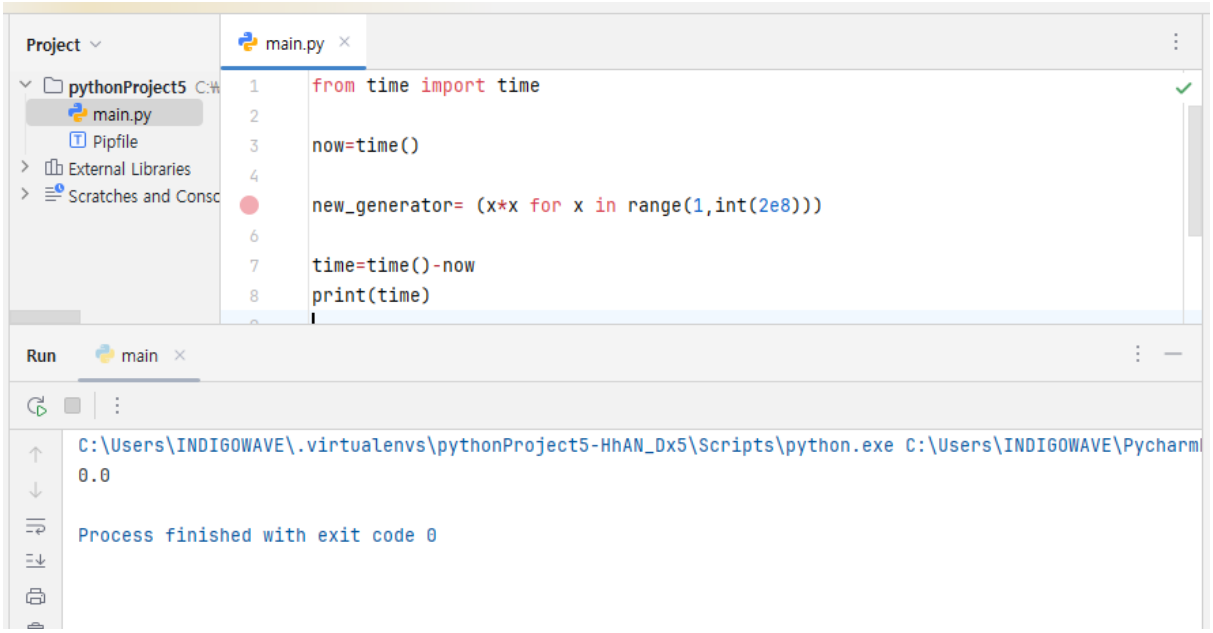
```
numbers=[x for x in range(5)]
numbers[3]
>>> 3
```

하지만 lazy evaluation 객체는 평가되어지지 않고 정보 만 가지고 있다.



요런 식으로

제너레이터는 사용 시점에 평가(계산) 되므로 메모리 효율적이다.



사용한 제너레이터는 두번 사용이 불가능하다.

정리

1. 컴프리헨션은 파이썬에서 객체를 효율적으로 만들 수 있는 구문
2. 컴프리헨션을 통해 리스트,셋,딕셔너리,제너레이터 와 같이 객체를 생성할 수 있다.
3. 제너레이터 컴프리헨션은 다른 객체 컴프리헨션과 다르게 지연 평가를 활용한다.

	리스트,셋,딕셔너리 컴프리헨션	제너레이터 컴프리헨션
평가 방식	즉시 평가	지연 평가
반환 타입	리스트,셋,딕셔너리 등 자료구조	제너레이터 객체
메모리 사용량	메모리에 즉시 전체 저장	메모리 효율적, 값 하나씩 생성
적합한 사용 사례	데이터 변환을 빠르고 간단하게 처리	큰 데이터셋,무한 시퀀스 처리
상태 유지	상태를 유지하지 않음	`yield` 로 상태를 유지하며 중단 및 재개 가능