39강. 제너레이터

혼자 공부하는 파이썬 - 윤인성 -

https://www.youtube.com/playlist?list=PLBXuLgInP-5kr0PclHz1ubNZgESmliuB7

제너레이터 들어가기전에 iterable, iterator에 대해서…

iterable: 반복할 수 있는 객체를 의미한다. for 반복문에서 in 뒤에 들어가는 것들…
(*리스트,튜플,딕셔너리 등*)

iterator: 이터러블중에 next()함수로 하나하나 꺼낼 수 있는 요소를 의미함.

이터러블이라고 다 이터레이터인것은 아니다. 하지만 이터레이터는 모두 이터러블이다.

즉 이터레이터는 for 반복문에 사용할 수 있고 next()함수로 조금씩 실행할수도 있고 …

```
a= [1,2,3]
x = iter(a)
print(type(x))
print(next(x))
print(next(x))
```

```
<class 'list_iterator'>
1
2
```

```
a= [1,2,3]
x = iter(a)
print(type(x))
for i in x:
    print(i)
```

```
<class 'list_iterator'>
1
2
3
```

1. 제너레이터란?

제너레이터는 *이터레이터*를 직접 만들 때 사용하는 코드입니다.

함수 내부에 yield 키워드를 사용하면 해당 함수는 제너레이터 함수가 되며, 일반 함수와는 달리 함수를 호출해도 함수 내부의 코드가 실행되지 않습니다.

```
def test():
    print("함수가 호출되었습니다.")
    yield

print(test())
```

```
<generator object test at 0x00002B04BE004A0>
```

test() 함수를 호출했지만 “함수가 호출되었습니다”라는 문자열 대신 generator object…
저쩌구가 나왔습니다. *제너레이터 함수는 제너레이터를 리턴*합니다.
그래서 함수 내부의 코드를 실행하려면 next() 함수를 사용해줘야 합니다.

```
def test():
    print("함수가 호출되었습니다.")
    yield 100

a = test()

print(next(a))
```

```
함수가 호출되었습니다.
100
```

next() 함수를 사용하여 함수 내부 코드를 호출했으며, 이 때 yield 키워드 부분까지만 실행됩니다.
next() 함수의 리턴값으로 yield 키워드 뒤에 입력한 값이 출력됩니다.

```
def test():
    print("출력a")
    yield 1
    print("출력b")
    yield 2
    print("출력c")
    yield 3

a = test()

print(next(a))
print(next(a))
```

```
출력a
1
출력b
2
```

왼쪽 예제에서 next() 함수를 한번 호출하면 yield 첫번째까지
두번 호출하면 yield 두번째까지…
yield가 3개인데 next() 4번 호출하면 Stopiteration 예외 발생

끝으로…

next() 함수를 호출한 이후 yield 키워드를 만나지 못하고 함수가 끝나면 Stopiteration이라는 예외가 발생합니다.

제너레이터 객체는 함수의 코드를 조금씩 실행할 때 사용합니다. 이는 메모리 효율을 위해서 입니다.

제너레이터 객체와 이터레이터 객체는 완전히 같지는 않지만, 기본적인 단계에서는 거의 비슷하다고 봐도 무방합니다.