Day 9 - Python Special(Magic) Method

1. __repr__()

```
[1]  class Dog:
         def __init__(self, name, age):
             self.name = name
             self.age = age

         def __repr__(self):
             return f"Dog(name='{self.name}', age = {self.age})" # 객체를 문자열로 변환할 때 자동으로 호출.
```

```
rucy = Dog('루시', 15)
print(rucy)
```

```
Dog(name='루시', age = 15)
```

- 객체를 호출할때 나오는 형식을 변환시킬 수 있다.
- 각각의 클래스를 만든 사람들이 객체가 찍혔을때 어떻게 나와라 라는걸 정의

```
rucy = Dog('루시', 15)
print(rucy)
print(str(rucy))
print(repr(rucy))
```

```
Dog(name='루시', age = 15)
Dog(name='루시', age = 15)
Dog(name='루시', age = 15)
```

- 문자열로 찍으면 모두 정의된 형식으로 나온다.

eval() function:

```
# eval() : 주어진 문자열을 파이썬 표현식으로 평가하고 실행하여 그 결과를 반환하는 내장 함수
x = 10
y = 3
result = x + y
print(result)
result = eval("x + y")  #문자열의 표현식을 실행 할 수 있다.
print(result)
```

```
13
13
```

- 문자열의 표현식을 실행가능하다.

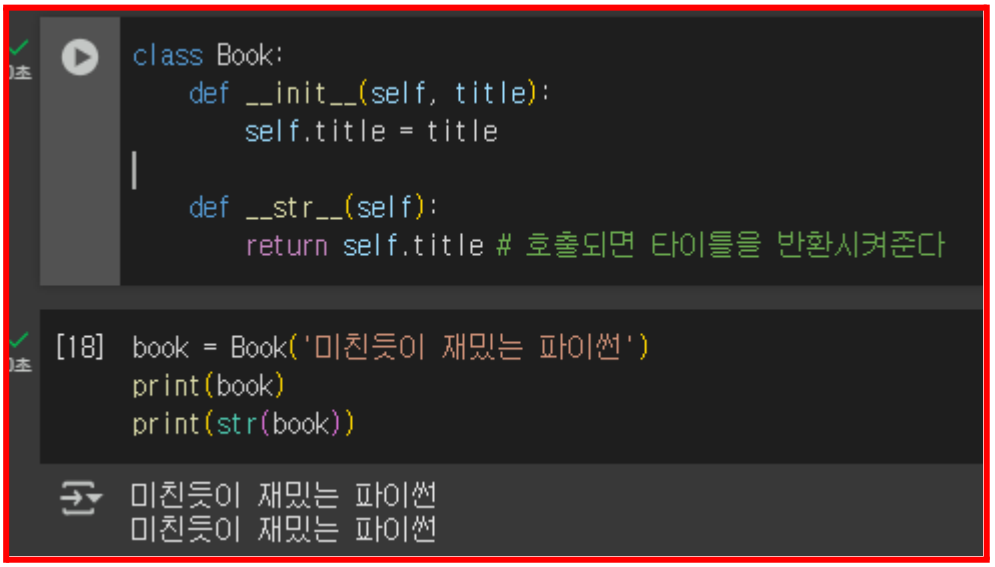
```
rucy_repr = repr(rucy)
print(rucy_repr)
result = eval(rucy_repr)
print(result)
print(result == rucy) # False, 같은 값을 가진 다른 객체 생성
```

```
Dog(name='루시', age = 15)
Dog(name='루시', age = 15)
False
```

(객체 복사도 가능)

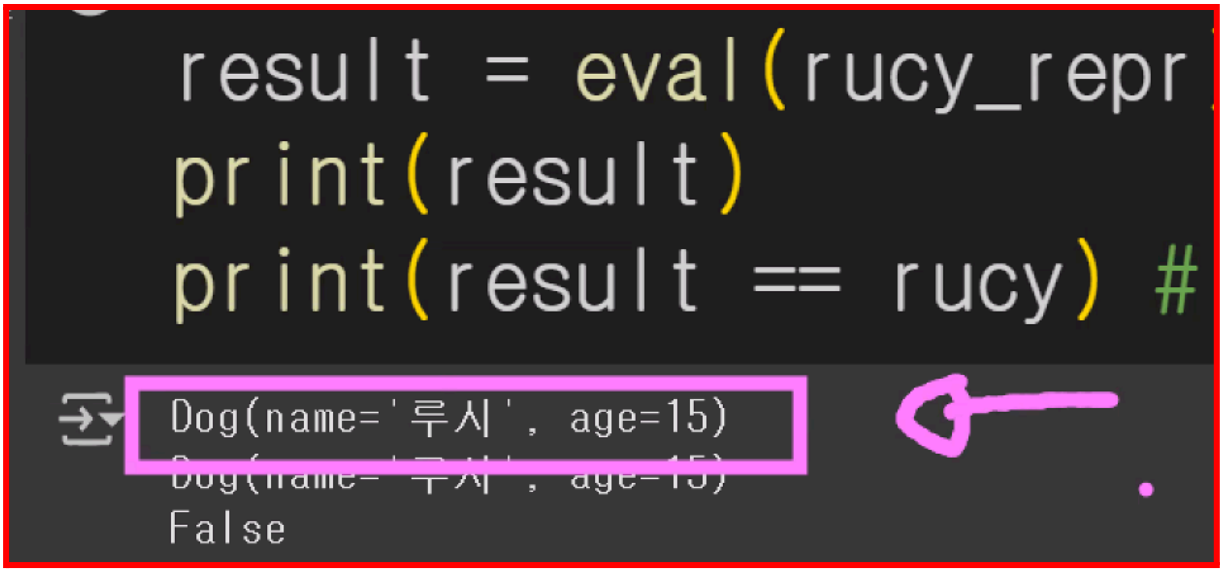
2. __str__

- repr와 비슷하다.
- str()로 호출이 가능하다.
- 사용자가 이해하기 쉽고 읽기 좋은 형태의 문자열 표현을 반환하는 메서드로, 주로 객체를 출력할 때 사용됩니다



- 똑같이 형식을 변환시켜서 객체를 호출할때 반환된다.

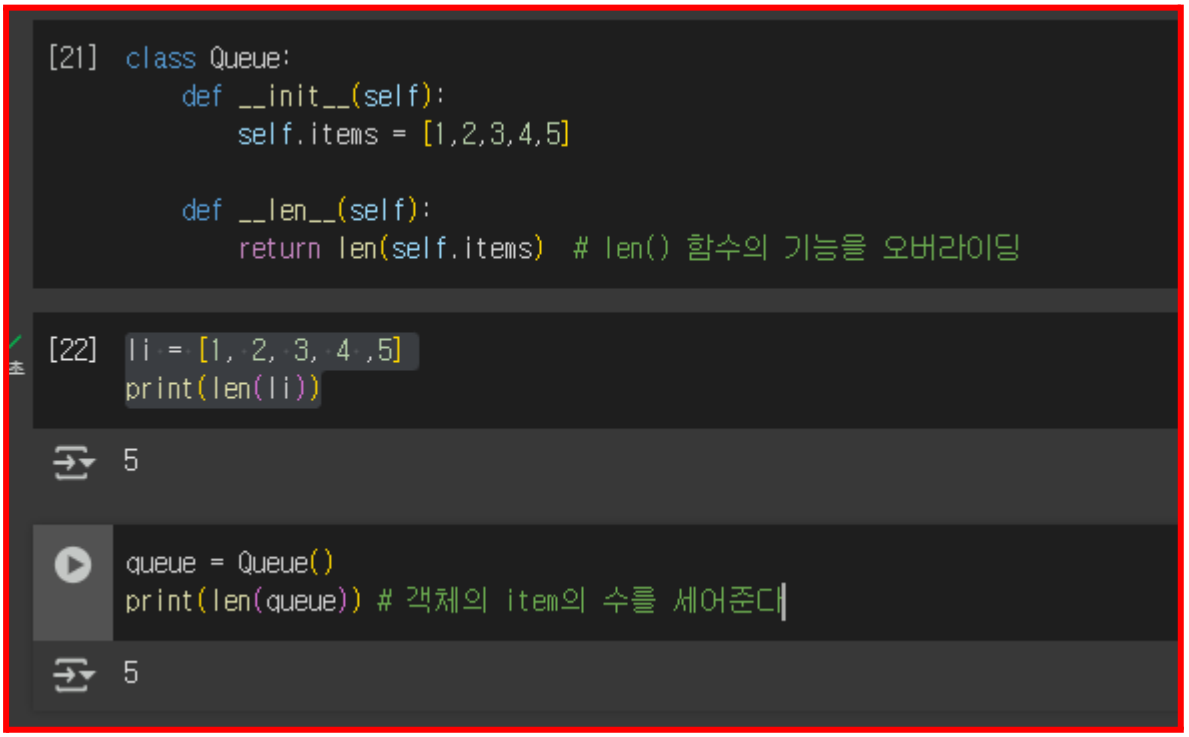
str vs repr



- __repr__
  - 객체를 복원할수 있는 문자열로 만듬
  - 개발할때 사용하는 용도
- __str__
  - 사용자에게 보여줄 깔끔한 설명 (출력용)
  - 디버깅용도
- 둘다 동시에 사용될때 우선순위:
  - print(book) --> __str__ 우선 >> __repr__

3. __len__()

- 파이썬에서 객체의 길이 또는 크기를 반환하는 특별한 메서드로, len() 함수가 호출될 때 자동으로 호출됩니다. 이 메서드는 객체의 항목 수를 측정하거나, 특정한 크기(예: 리스트, 문자열, 튜플 등)를 나타내고자 할 때 구현됩니다.



```
[21] class Queue:
         def __init__(self):
             self.items = [1,2,3,4,5]

         def __len__(self):
             return len(self.items)  # len() 함수의 기능을 오버라이딩

[22] li = [1, 2, 3, 4 ,5]
     print(len(li))

     5

     queue = Queue()
     print(len(queue)) # 객체의 item의 수를 세어준다

     5
```

- Queue라는 객체를 생성했을때 사용가능하다.

4. getitem()

- 파이썬에서 인덱싱을 지원하기 위해 사용되는 특별한 메서드로, 객체의 특정 항목을 가져오기 위해 대괄호([])를 사용할 때 호출됩니다. 이 메서드는 객체의 특정 인덱스나 키에 해당하는 값을 반환하는 역할을 합니다.

```
[28] class Point:
         def __init__(self, x, y):
             self.x = x
             self.y = y

         # 인덱싱 기능을 추가해주는 스페셜 메서드
         def __getitem__(self, index):
             if index == 0:
                 return self.x
             elif index == 0:
                 return self.y
             else:
                 return -1
```

```
pt = Point(5, 3)
print(pt)
print(pt[0]) # __getitem__ 메서드를 사용해서 인덱싱을 해준다.
print(pt[1])
print(pt[100])
```

```
<__main__.Point object at 0x7f8a1314f9d0>
5
-1
-1
```

- 인덱싱을 시켜주는 스페셜 메서드이다.

5. call()

파이썬에서 객체를 함수처럼 호출할 수 있게 해주는 특별한 메서드입니다. 이 메서드를 구현하면, 해당 클래스의 인스턴스를 함수처럼 사용할 수 있으며, 인스턴스에 대해 괄호(())를 사용하여 값을 전달하고 결과를 반환받을 수 있습니다.

+ 코드 + 텍스트

텍스트 셀 추가

```
[33] class CallableObject:
         def __call__(self, *args, **kwargs):
             print(f'args:{args}, kwargs:{kwargs}')

     callable_obj = CallableObject()
     callable_obj(1, 2, 3, a='A', b='B')
```

```
args:(1, 2, 3), kwargs:{'a': 'A', 'b': 'B'}
```