Chapter 02. 파이썬 문법

3.1 이름과 객체

1) 파이썬에서 **모든 자료(데이터)들은 객체의 형태로 저장**된다.

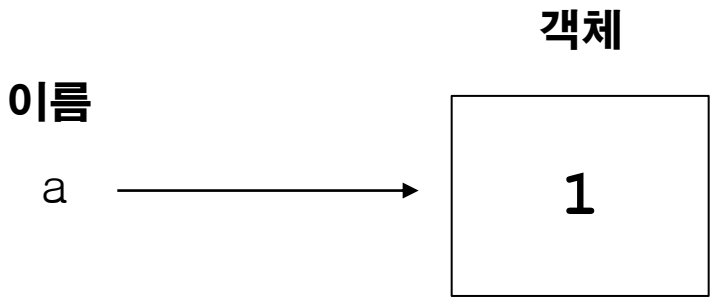
2) 파이썬의 변수는 컴파일러 언어처럼 변수에 할당된 **값을 저장하는 저장공간(메모리)의 주소 심볼릭이 아니다.**

3) 객체의 상태(변수, 값)들이 저장된 저장공간(메모리)의 주소를 저장하거나(C, C++) 또는, 다른 형태로 변환된 레퍼런스(객체 ID)와 같은 값을 저장하지(Java) 않는다.

3) 즉, **변수는 단지 객체의 이름(심볼)일 뿐**이다.

4) 파이썬의 객체 이름(변수)과 객체의 ID(주소)는 **심볼 테이블(Symbol Table)**에 함께 저장되어 관계를 가지게 된다.

a = 1



3.2 심볼 테이블(Symbol Table)

1) 심볼 테이블은 변수의 이름과 저장된 데이터의 주소를 저장하는 테이블이다.
2) 심볼 테이블을 내용을 살펴 보기위해 global(), locals() 내장 함수를 사용한다.
3) 두 함수는 해당 스코프의 심볼 테이블의 내용의 사전(dict) 타입의 객체를 반환한다.

[예제 symbol-table.py]

```
# 심볼 테이블 내용확인
g_a = 1
g_b = 'symbol'

def f():
    l_a = 2
    l_b = 'table'
    print(locals())

for i in range(10):
    g_c = 3
    g_d = 'python'
    print(locals())

f()
print(globals())
```

3. 객체

[예제 symbol-table.py]

객체의 심볼테이블을 접근하기 위해서는 __dict__ 속성의 내용을 확인한다.

```
print(f.__dict__)

class MyClass:
    x = 10
    y = 20

print(MyClass.__dict__)
```

심볼 테이블(심볼 즉 이름 또는 변수가 저장되는 공간)을 네임스페이스(namespace, 이름공간)이라 한다.

1. 모듈
2. 클래스 객체
3. 인스탄스 객체
4. 정의된 함수

모든 객체가 심볼 테이블을 가지고 있는 것은 아니다.

3.3 객체 레퍼런스 카운트(Reference Count)와 쓰레기 수집(Garbage Collection)

1) 레퍼런스 카운트는 객체를 참조 수이다.
2) 레퍼런스 카운트가 0 가 되면 더 이상 사용하지 객체이므로 자동으로 사라진다.
3) 이 러한 작업을 GC라 부른다.

[예제 ref_count.py]

```
x = object()
print(sys.getrefcount(x))

y = x
print(sys.getrefcount(x))

# 레퍼런스 값이 준다.
del x
print(sys.getrefcount(y))
```

3.4 객체 ID

id() 함수를 이용하면 객체의 주소를 식별할 수 있다.

[예제 obj_id.py]

만일, 두 객체의 ID가 동일하면, 같은 객체를 참조하고 있는 것이다.

```
i1 = 10
i2 = 10

print(hex(id(i1)), hex(id(i2)))

l1 = [1, 2, 3]
l2 = [1, 2, 3]
print(hex(id(l1)), hex(id(l2)))

s1 = 'hello'
s2 = 'hello'
print(hex(id(s1)), hex(id(s2)))



print(i1 is i2)
print(l1 is l2)
print(s1 is s2)
```

3.5 객체의 복사

1) 레퍼런스 복사

객체를 참조하는 주소만 복사하는 것

[예제 obj_copy.py]

```
a = 1
b = a

a = [1, 2, 3]
b = [4, 5, a]
x = [a, b, 100]
y = x

print(x)
print(y)
```

따라서 x의 변경은 곧 y의 변경과 같다.

3.5 객체의 복사

2) copy 모듈을 이용한 복사

얕은복사 (Shallow Copy)

1단계 복합객체를 생성하고 원래 객체로 부터 내용을 복사한다.

[예제 obj_copy.py]

```
a = 1
b = a

a = [1, 2, 3]
b = [4, 5, a]
x = [a, b, 100]
y = x

print(x)
print(y)
```

깊은 복사(Deep Copy)

복합객체를 재귀적으로 생성하고 복사한다.

```
a = [1, 2, 3]
b = [4, 5, a]
x = [a, b, 100]
y = copy.deepcopy(x)

print(x)
print(y)
```

깊은 복사가 복합 객체만을 생성하기 때문에 복합객체가 한개 만 있을 경우 얕은 복사와 깊은 복사의 차이가 없다.

```
a = ["hello", "world"]
b = copy.copy(a)
print(a is b)
print(a[0] is b[0])

c = copy.deepcopy(a)
print(a is c)
print(a[0] is c[0])
```