그림 2.6 변수와 객체의 바인딩 예

'변수의 주소'는 아니다!

그렇다면 파이썬에서 메모리에 할당된 객체의 주소(정확히 말하면 주소는 아니지만 의미상으로는 주소로 생각해도 됩니다)는 어떻게 확인할 수 있을까요? 다음과 같이 id라는 함수를 사용하면 됩니다.

위 코드를 실행한 후 결괏값을 보면 id 값이 같습니다. 즉, 두 변수가 서로 같은 객체를 가리키고 있음을 확인할 수 있습니다. 이를 그림으로 나타내면 그림 2.7과 같습니다. 파이썬은 위 코드에 대해 2번과 같은 방식으로 동작했던 것입니다.

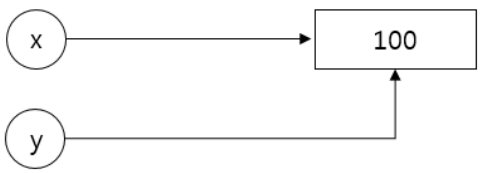

그림 2.7 변수와 객체의 바인딩 예(1)

단, 파이썬이 항상 2번 방식으로 동작하는 것은 아닙니다. 다음 코드를 실행해 보면 id 값이 서로 다르게 나옵니다.

```
>>> x = 10000
>>> y = 10000
>>> id(x), id(y)
(44212048, 48072352)
>>>
```

위 코드에서 변수와 객체의 관계를 그림으로 나타내면 그림 2.8과 같습니다. 변수 x와 변수 y는 서로 다른 객체를 가리키고 있습니다. 그림 2.8에서 파란색으로 나타낸 부분은 객체의 id 값을 의미합니다. 참고로 위 코드를 실행했을 때 반환되는 id 값은 실행할 때마다 달라질 수 있으므로 이책에 나온값과 여러분이 확인한 값이 서로 다를 수 있습니다.

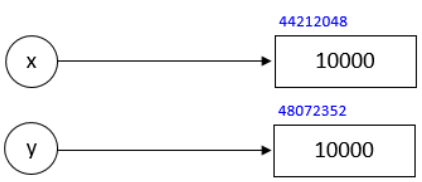

그림 2.8 변수와 객체의 바인딩 예(2)

파이썬이 이렇게 동작하는 이유는 프로그램을 작성할 때 정숫값(Integer) 중 자주 사용할 것 같은 범위의 정숫값은 메모리에 한 번만 올려두고 이를 여러 변수가 가리키게 함으로써 메모리를 효과적으로 사용하기 위해서입니다. 심심하신 분들은 파이썬 IDLE를 통해 위와 같은 방식으로 숫자에 대해 x, y라는 변수로 바인딩해보면 256까지는 id 값이 같지만 257부터는 서로 다른 객체가 생성되는 것을 확인할 수 있습니다.

코드 8-12 객체의 값, 유형, 정체성 구하기

```
>>> year = 1789  # 객체를 만들어 변수에 대입

>>> year         # 객체의 값 (객체 자신) 구하기
1789

>>> type(year)   # 객체의 유형 (클래스) 구하기
<class 'int'>

>>> id(year)     # 객체의 정체성 (고유번호) 구하기
140711867085328
```

값이나 유형이 같더라도 정체성이 다를 수 있다. 데이터를 비교할 때는 비교하는 것이 객체의 값인지, 유형인지, 정체성인지 헷갈리지 않도록 주의해야 한다.