교수의 Python

01_1 자료형과 참조 변수

창원대학교 정보통신공학과 교수 박동규

널널한 교수의 Python

01_1 자료형과 참조 변수

창원대학교 정보통신공학과 교수 박동규

널널한 교수의 고급 Python

01_1 자료형과 참조 변수

창원대학교 정보통신공학과 교수 박동규

다룰 내용

- 고급 파이썬 문법
    - with, pass
- 파이썬의 참조형과 리스트
- 사용하기는 하지만 개념이 애매한 내용들
- 확실하게 깊이있게 파보는 내용입니다.
- 기초편 강의를 꼭 들어주십시오~~~

Everything is an Object

Everything is an Object

- 파이썬은 객체지향 프로그래밍 언어이다

Everything is an Object

- 파이썬은 객체지향 프로그래밍 언어이다
- 파이썬은 객체가 중심이 되며, 참조 변수를 통해 객체에 접근할 수 있다.

Everything is an Object

- 파이썬은 객체지향 프로그래밍 언어이다
- 파이썬은 객체가 중심이 되며, 참조 변수를 통해 객체에 접근할 수 있다.
- C 언어는 변수가 생성되고 변수에 값이 저장되는 구조이다.

Everything is an Object

- 파이썬은 객체지향 프로그래밍 언어이다
- 파이썬은 객체가 중심이 되며, 참조 변수를 통해 객체에 접근할 수 있다.
- C 언어는 변수가 생성되고 변수에 값이 저장되는 구조이다.
- 파이썬의 변수는 동적으로 참조하는 객체가 지정된다.

자료형

- 파이썬은 정수형, 실수형, 복소수형, 문자열, 리스트 등의 다양한 자료형이 있다.
- 자료형에 따라서 지원하는 연산자가 다르며 메소드들도 다르다.
- 사용자의 필요성에 따라 적절한 자료형을 선택하여 사용한다.

자료형	형식	설명

자료형	형식	설명
int 형	n = 100 n += 200	음의 정수, 0, 양의 정수 값을 가지며 사칙연산자를 비롯한 연산자를 사용할 수 있다

자료형	형식	설명
int 형	n = 100 n += 200	음의 정수, 0, 양의 정수 값을 가지며 사칙연산자를 비롯한 연산자를 사용할 수 있다
float 형	f = 12.3 f = f - 30.12	소숫점 아래 숫자를 가질 수 있으며 정밀한 숫자 표현이 가능하다. 정수형에서 사용하는 연산자를 많이 사용할 수 있다.

자료형	형식	설명
int 형	n = 100 n += 200	음의 정수, 0, 양의 정수 값을 가지며 사칙연산자를 비롯한 연산자를 사용할 수 있다
float 형	f = 12.3 f = f - 30.12	소숫점 아래 숫자를 가질 수 있으며 정밀한 숫자 표현이 가능하다. 정수형에서 사용하는 연산자를 많이 사용할 수 있다.
complex 형	a = 3 + 2.0J	실수부와 허수부를 가지는 복소수를 표현할 수 있는 자료형

자료형	형식	설명
int 형	n = 100 n += 200	음의 정수, 0, 양의 정수 값을 가지며 사칙연산자를 비롯한 연산자를 사용할 수 있다
float 형	f = 12.3 f = f - 30.12	소숫점 아래 숫자를 가질 수 있으며 정밀한 숫자 표현이 가능하다. 정수형에서 사용하는 연산자를 많이 사용할 수 있다.
complex 형	a = 3 + 2.0J	실수부와 허수부를 가지는 복소수를 표현할 수 있는 자료형
str 형	s = “hello” s = s * 3 s.upper()	문자열을 저장함 문자열 반복 연산자를 지원 upper(), lower(), split() 등 메소드 지원

정수 객체와 변수

- 정수형 객체가 있으면 이 객체에 연산자를 적용하여 연산을 수행할 수 있다.

- 변수에 데이터를 보관하고 필요할 때 참조하면 편리하다

```
>>> a = 100
>>> print(a * 10)
1000
>>> print(a * 20)
2000
>>> a = 200
>>> print(a * 10)
2000
>>> print(a * 20)
4000
```

C

변수 중심

Python

객체 중심

C

변수 중심

```
int a;        // 정수형 변수 선언
```

Python

객체 중심

C

변수 중심

```
int a;        // 정수형 변수 선언
```

Python

객체 중심

C

변수 중심

```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
```

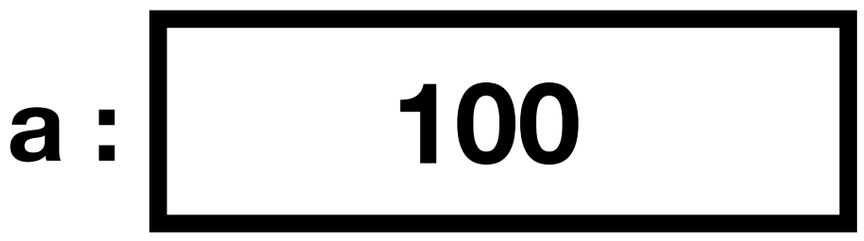
Python

객체 중심

C

변수 중심

```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
```



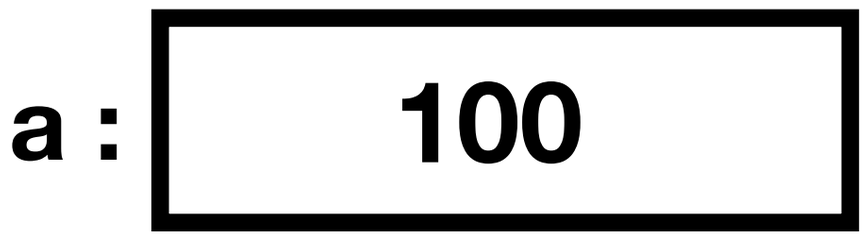
Python

객체 중심

C

변수 중심

```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



Python

객체 중심

C

변수 중심

```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```

a : 

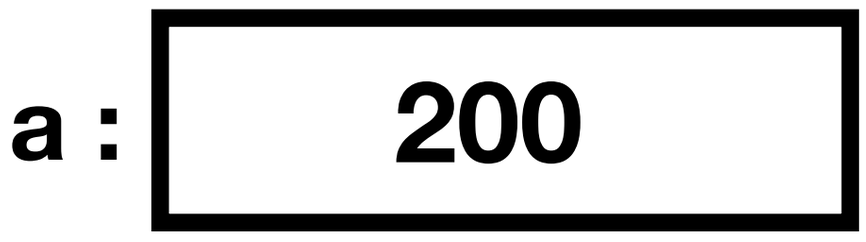
Python

객체 중심

C

변수 중심

```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



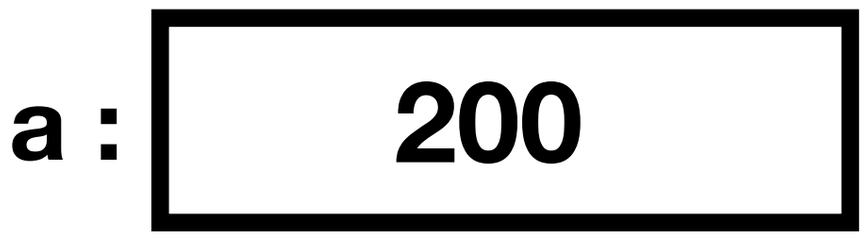
Python

객체 중심

C

변수 중심

```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



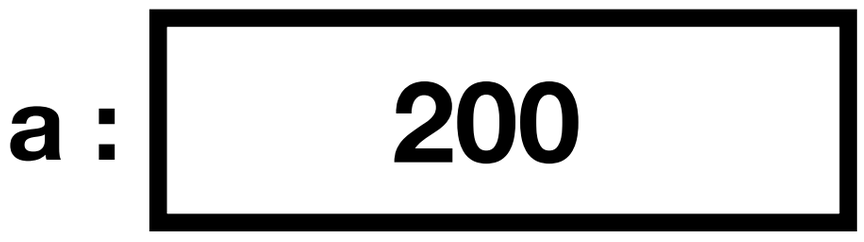
Python

객체 중심

C

변수 중심

```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



Python

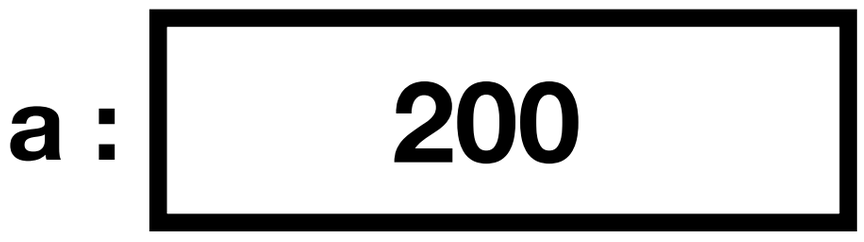
객체 중심

```
a = 100       # 객체 생성
```

C

변수 중심

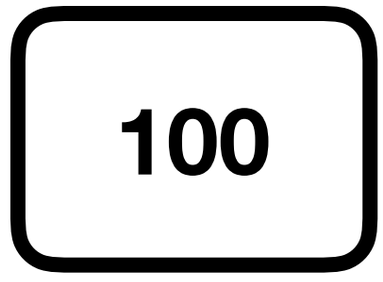
```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



Python

객체 중심

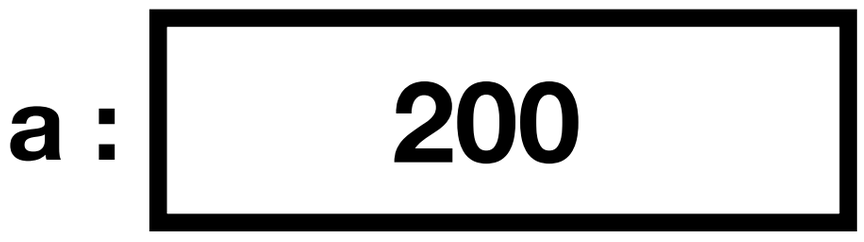
```
a = 100       # 객체 생성
```



C

변수 중심

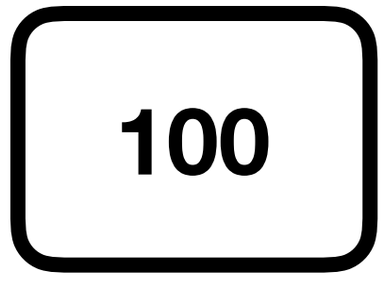
```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



Python

객체 중심

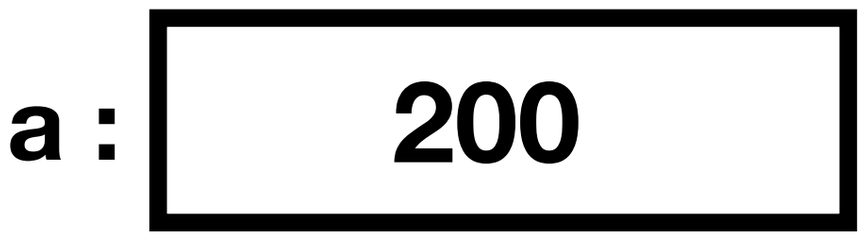
```
a = 100       # 객체 생성
#  객체에 대한 참조 변수 a
```



C

변수 중심

```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



Python

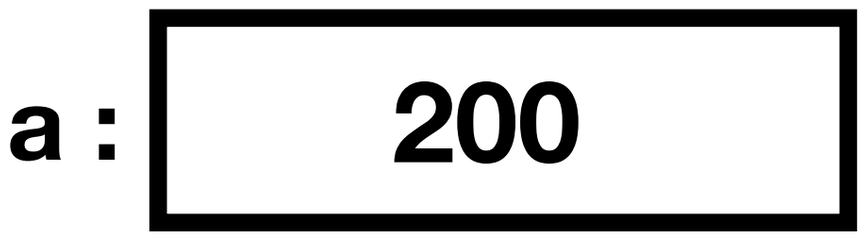
객체 중심

```
a = 100       # 객체 생성
#  객체에 대한 참조 변수 a
```

C

변수 중심

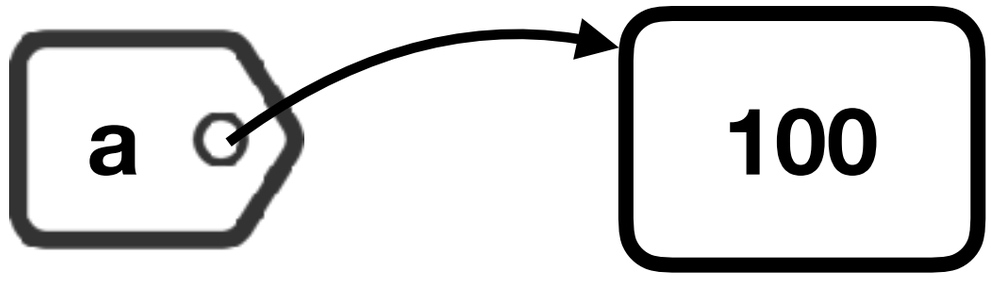
```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



Python

객체 중심

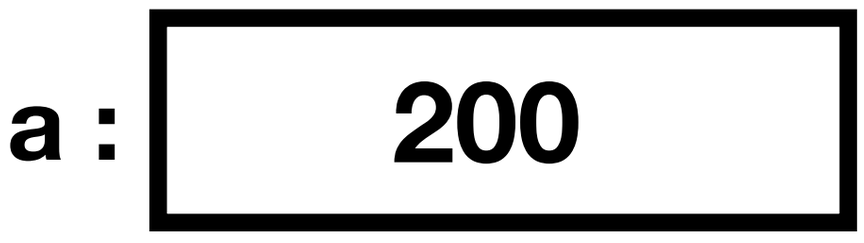
```
a = 100       # 객체 생성
#  객체에 대한 참조 변수 a
```



C

변수 중심

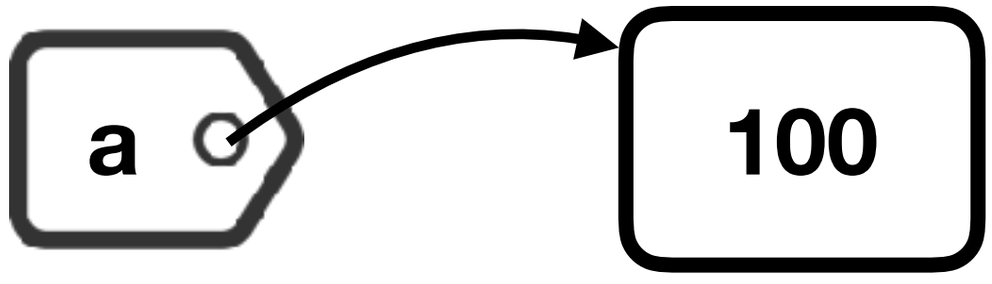
```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



Python

객체 중심

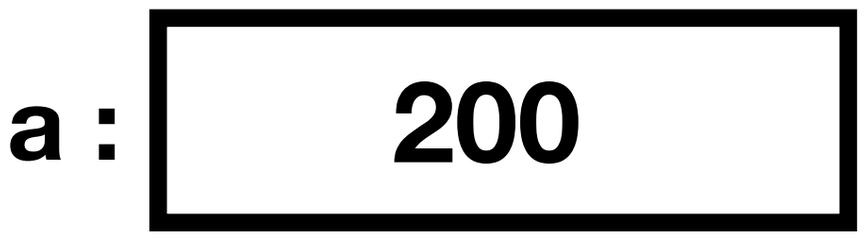
```
a = 100       # 객체 생성
#  객체에 대한 참조 변수 a
a = 200       # 객체 재할당
```



C

변수 중심

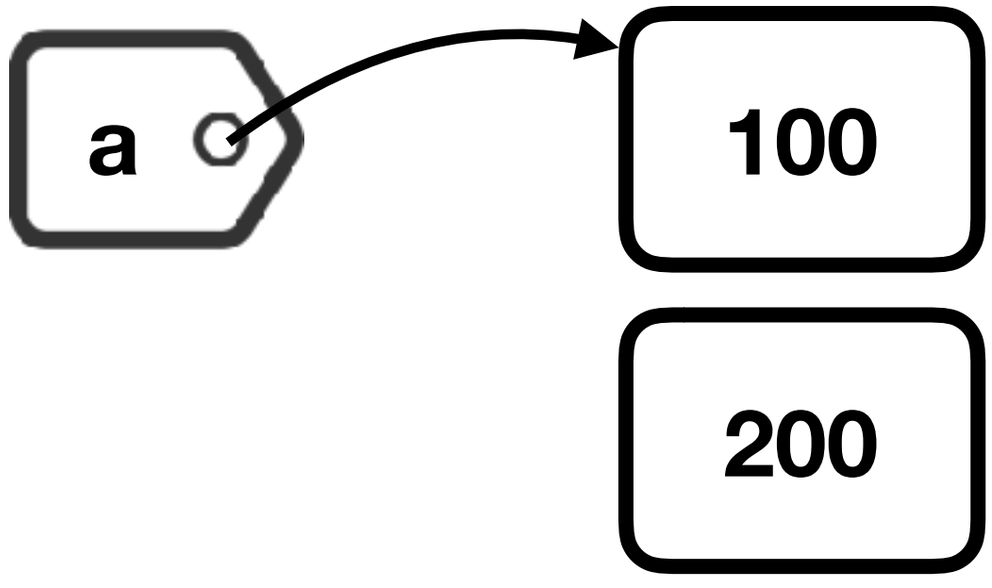
```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



Python

객체 중심

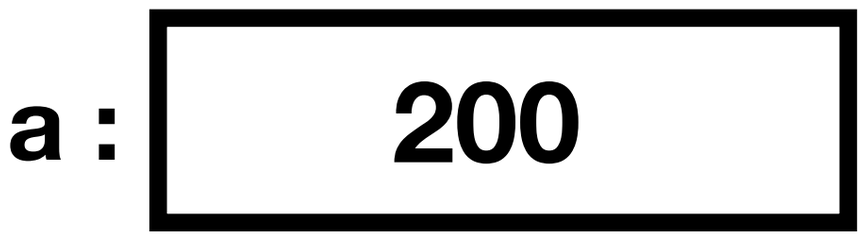
```
a = 100       # 객체 생성
#  객체에 대한 참조 변수 a
a = 200       # 객체 재할당
```



C

변수 중심

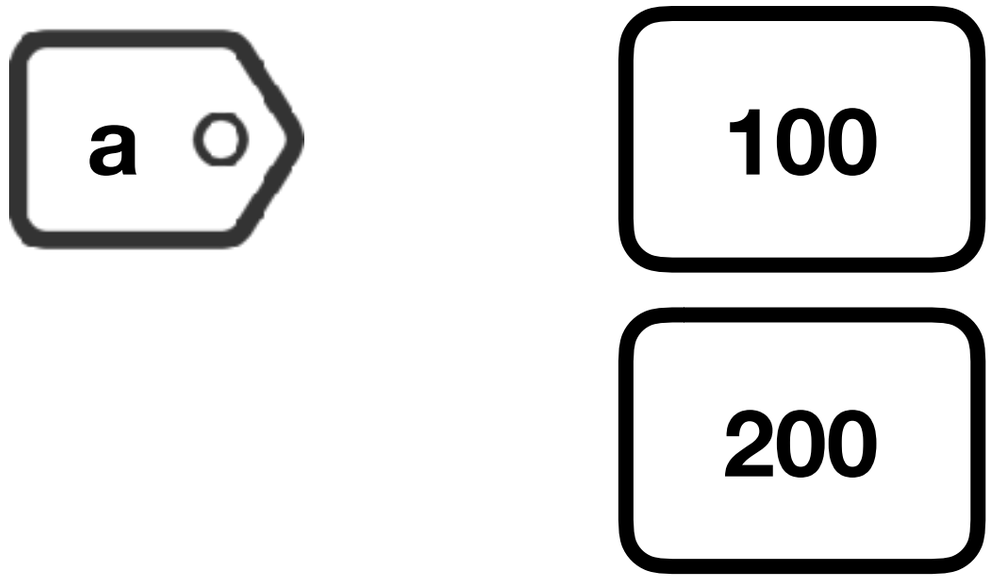
```
int a;        // 정수형 변수 선언
a = 100;      // 정수값 할당
a = 200;      // 정수값 재할당
```



Python

객체 중심

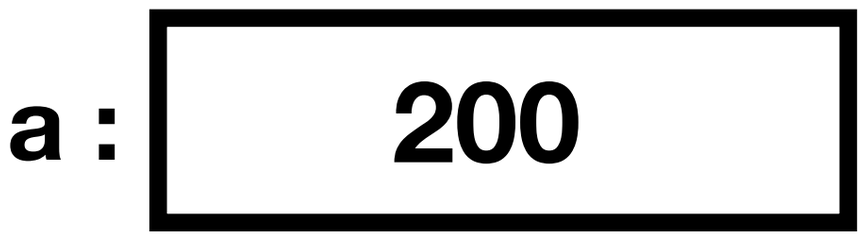
```
a = 100       # 객체 생성
#  객체에 대한 참조 변수 a
a = 200       # 객체 재할당
```



C

변수 중심

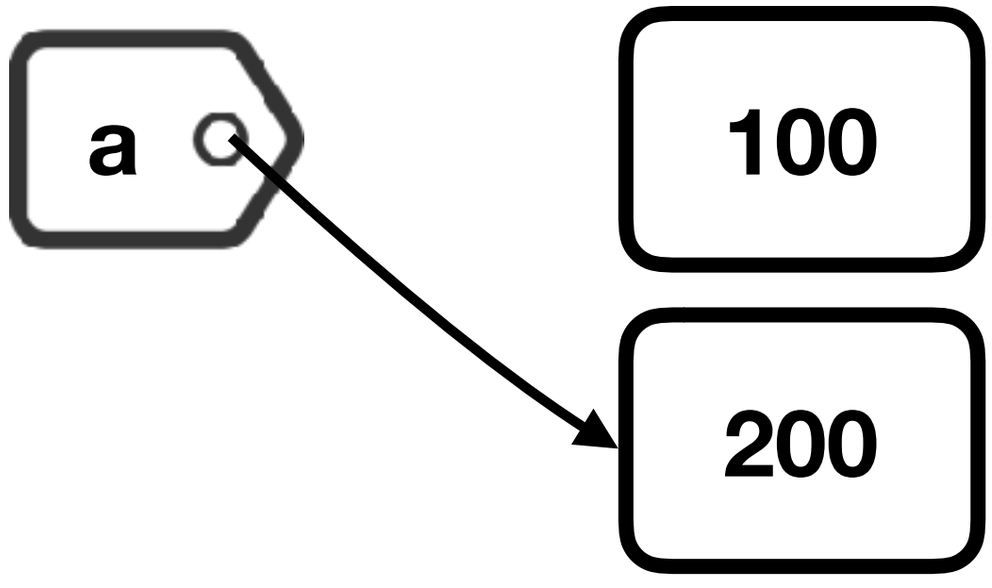
```
int a;          // 정수형 변수 선언
a = 100;        // 정수값 할당
a = 200;        // 정수값 재할당
```



Python

객체 중심

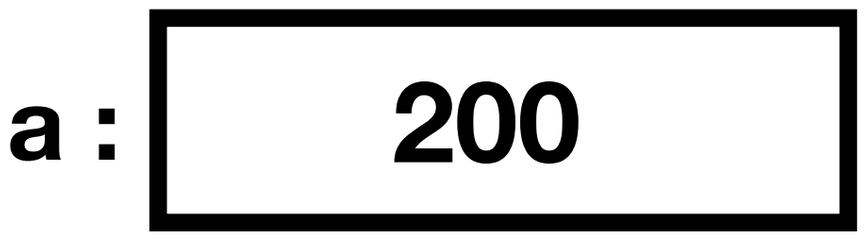
```
a = 100       # 객체 생성
#  객체에 대한 참조 변수 a
a = 200       # 객체 재할당
```



C

변수 중심

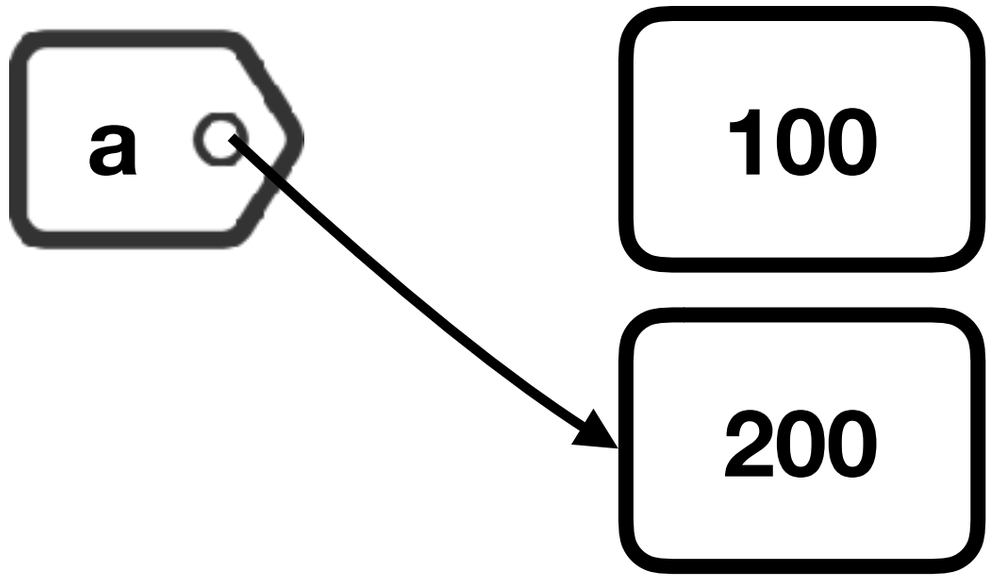
```
int a;          // 정수형 변수 선언
a = 100;        // 정수값 할당
a = 200;        // 정수값 재할당
```



Python

객체 중심

```
a = 100       # 객체 생성
#  객체에 대한 참조 변수 a
a = 200       # 객체 재할당
```

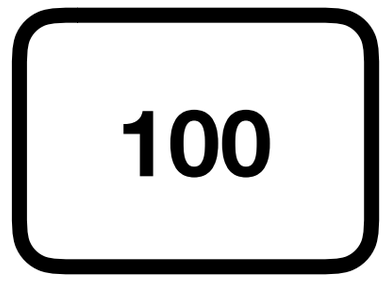


모든 객체는 고유한 id를 가진다

```
>>> a = 100
>>> print(id(a))
4324024528
>>> print(id(100))
4324024528
```

모든 객체는 고유한 id를 가진다

```
>>> a = 100
>>> print(id(a))
4324024528
>>> print(id(100))
4324024528
```



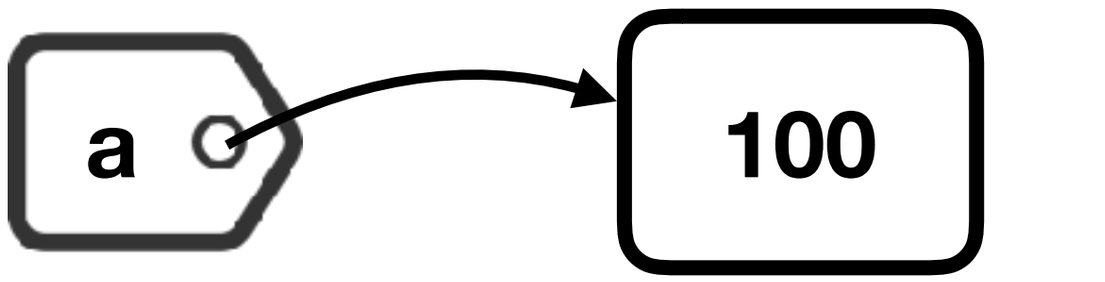
id : 4324024528

모든 객체는 고유한 id를 가진다

```
>>> a = 100
>>> print(id(a))
4324024528
>>> print(id(100))
4324024528
```

모든 객체는 고유한 id를 가진다

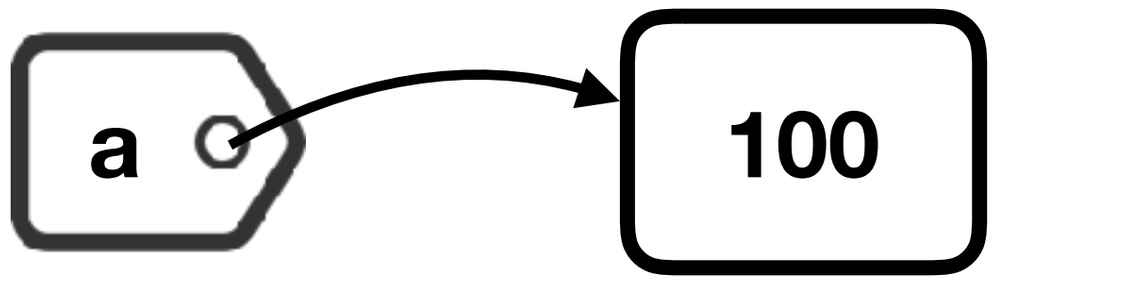
```
>>> a = 100
>>> print(id(a))
4324024528
>>> print(id(100))
4324024528
```



모든 객체는 고유한 id를 가진다

a가 참조하는 객체의
id를 반환

```
>>> a = 100
>>> print(id(a))
4324024528
>>> print(id(100))
4324024528
```

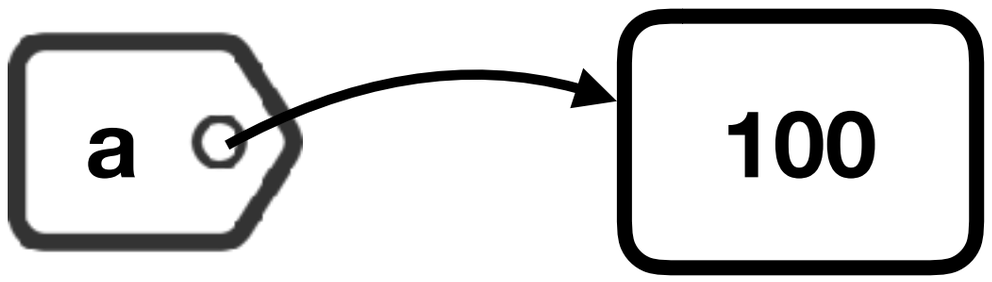


id : 4324024528

모든 객체는 고유한 id를 가진다

a가 참조하는 객체의
id를 반환

```
>>> a = 100
>>> print(id(a))
4324024528
>>> print(id(100))
4324024528
```

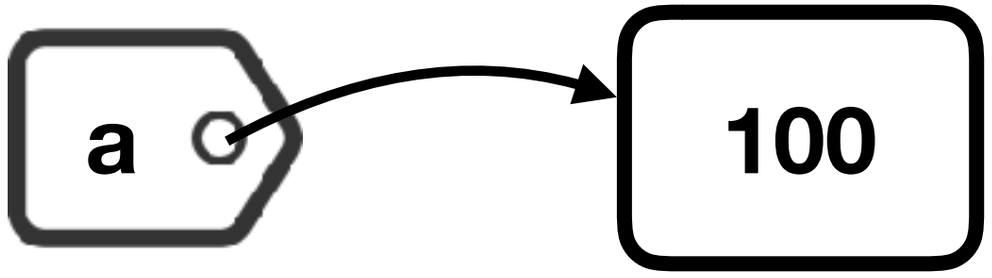


id : 4324024528

모든 객체는 고유한 id를 가진다

a가 참조하는 객체의
id를 반환

```
>>> a = 100
>>> print(id(a))
4324024528
>>> print(id(100))
4324024528
```



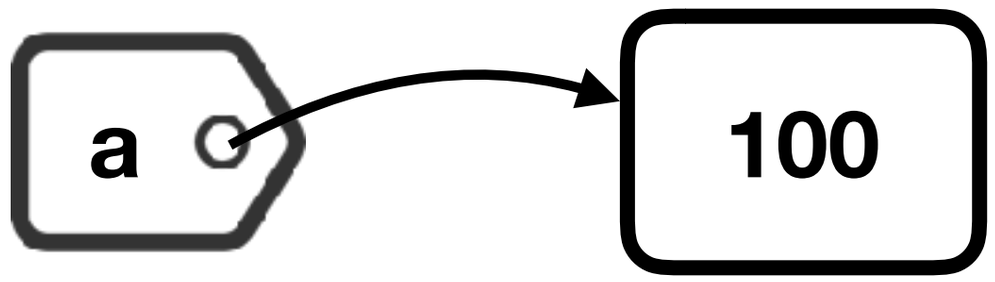
id : 4324024528

100 값을 가지는 정수
형 객체의 id

모든 객체는 고유한 id를 가진다

a가 참조하는 객체의
id를 반환

```
>>> a = 100
>>> print(id(a))
4324024528
>>> print(id(100))
4324024528
```



id : 4324024528

100 값을 가지는 정수
형 객체의 id

= 연산자가 하는 일

- 객체에 대한 참조를 만들거나 변경시킨다.

- 파이썬은 객체 중심적인 프로그래밍 언어이므로 객체와 객체를 참조하는 변수, 그리고 참조의 변경이라는 개념이 중요할 수 밖에 없다.

- 객체를 여러 변수가 동시에 참조할 수 있다.

객체를 여러 변수가 동시에 참조할 수 있다.

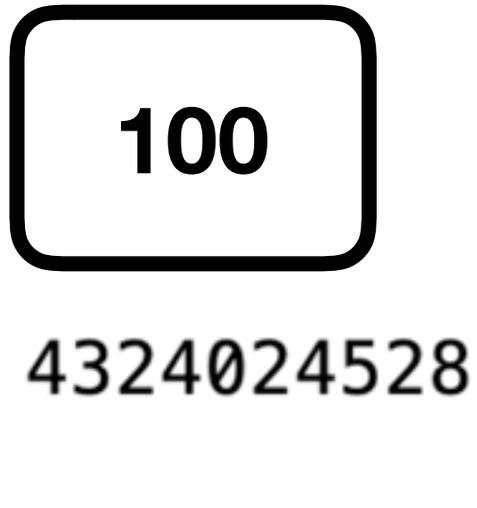
=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```

객체를 여러 변수가 동시에 참조할 수 있다.

=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```



객체를 여러 변수가 동시에 참조할 수 있다.

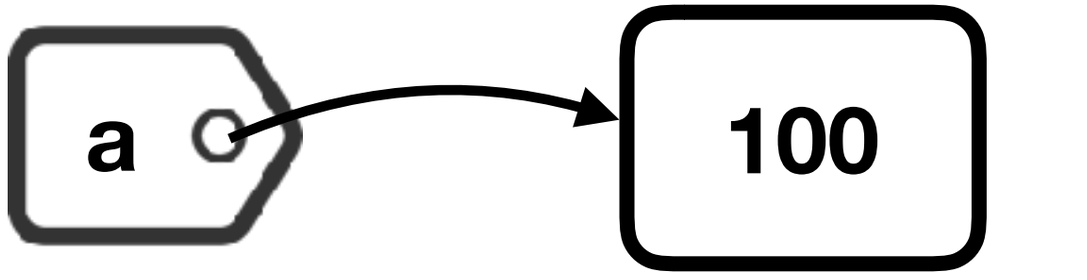
=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```

객체를 여러 변수가 동시에 참조할 수 있다.

=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

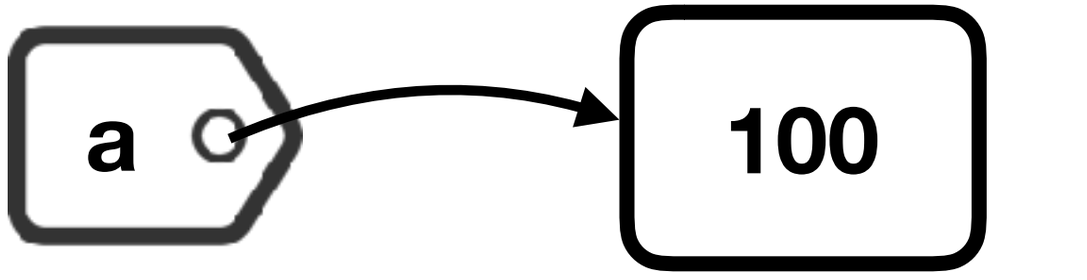
```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```



객체를 여러 변수가 동시에 참조할 수 있다.

=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

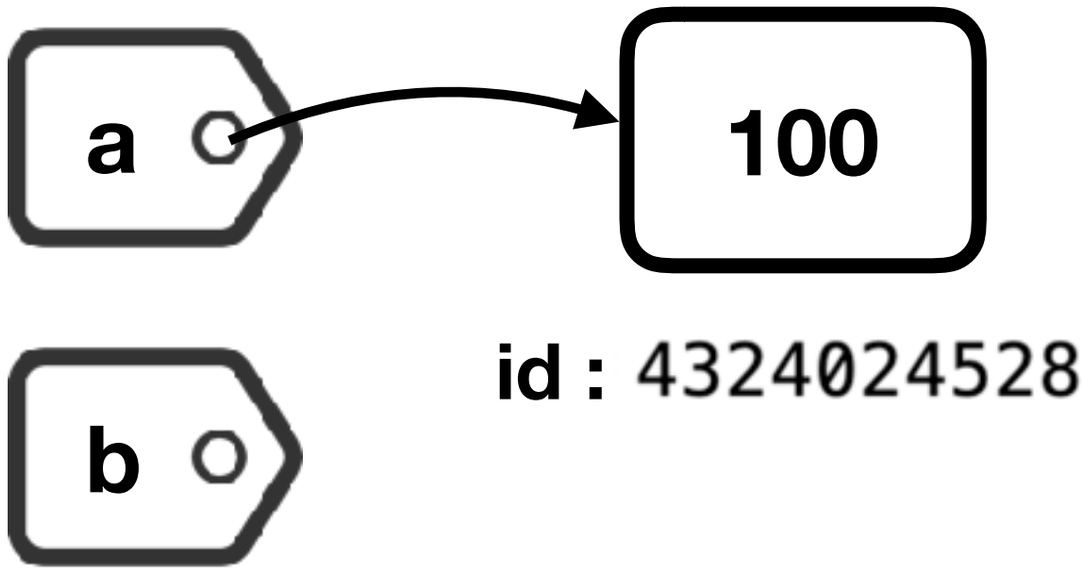
```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```



객체를 여러 변수가 동시에 참조할 수 있다.

=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

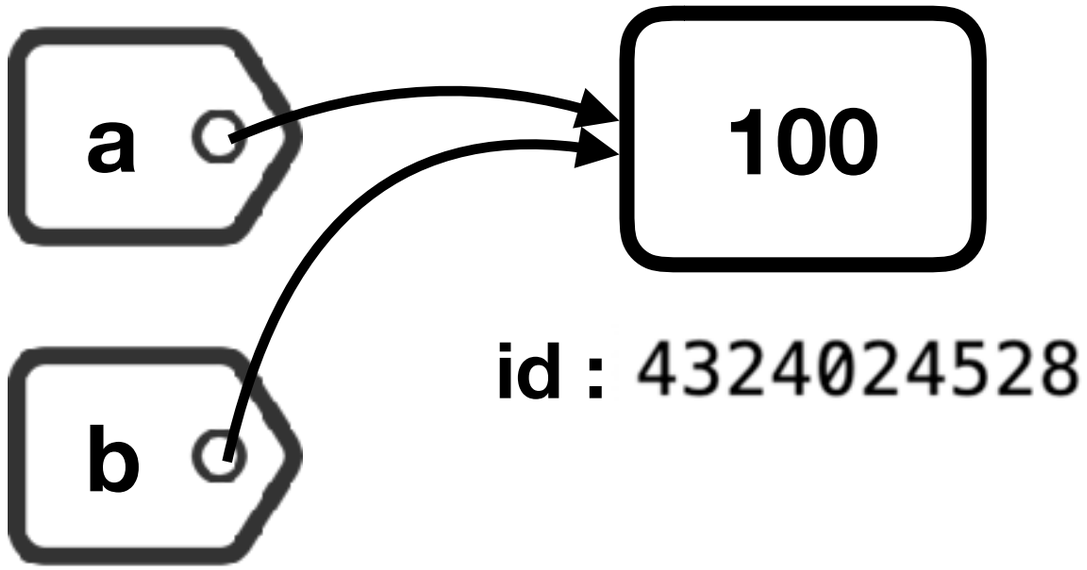
```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```



객체를 여러 변수가 동시에 참조할 수 있다.

=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

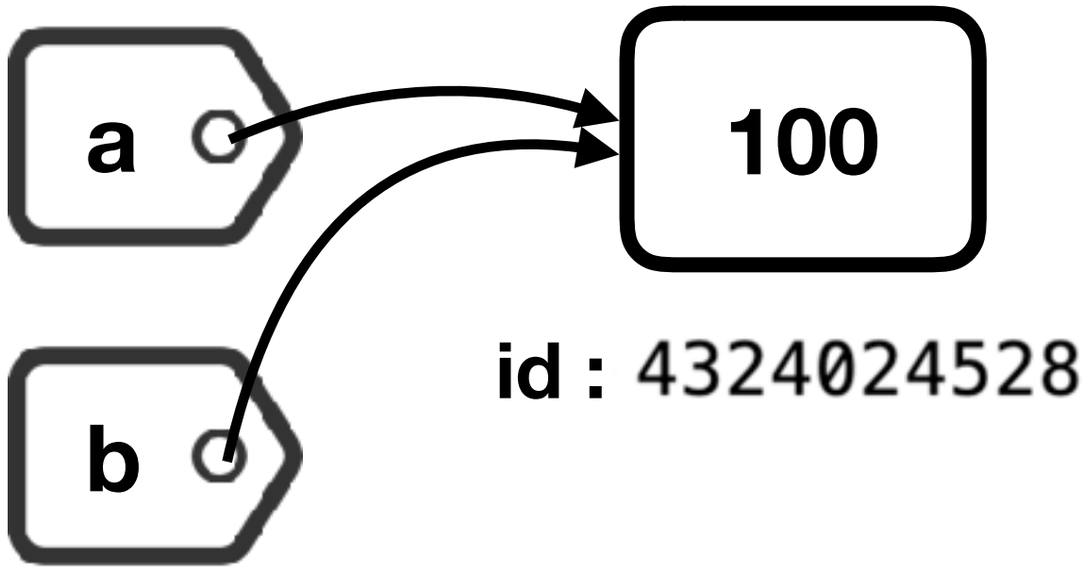
```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```



객체를 여러 변수가 동시에 참조할 수 있다.

=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

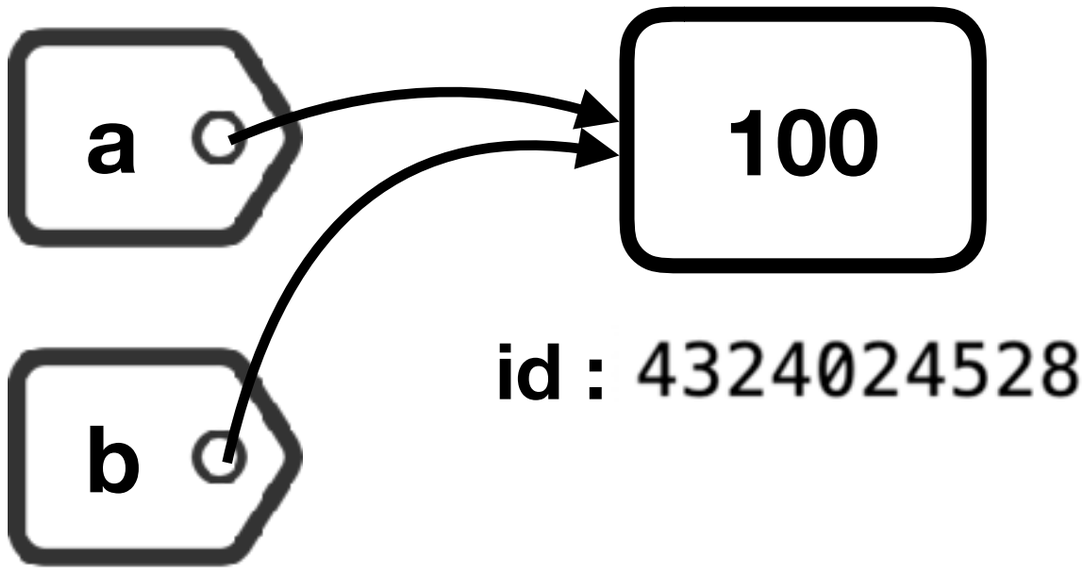
```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```



객체를 여러 변수가 동시에 참조할 수 있다.

=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```

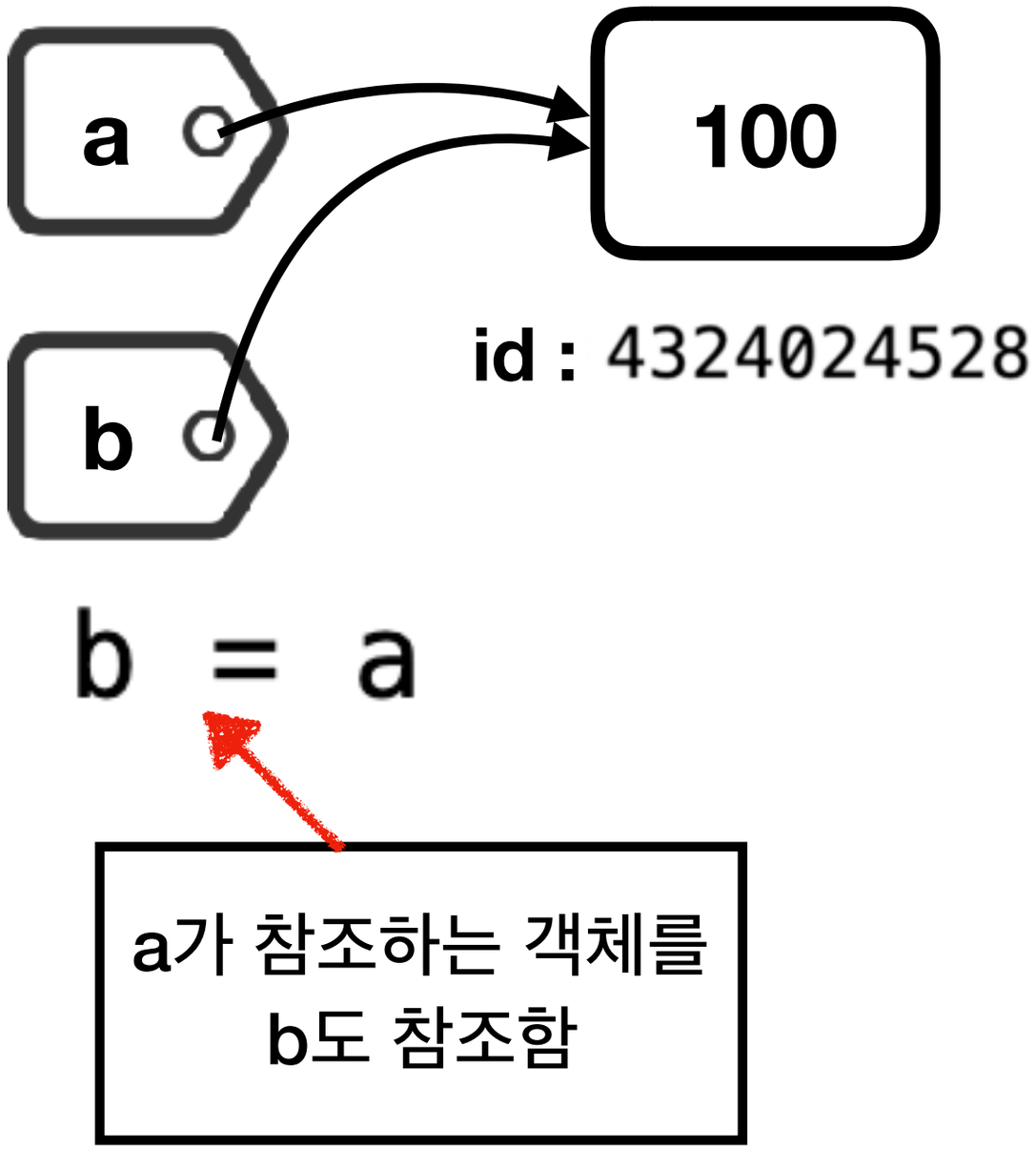


a가 참조하는 객체를
b도 참조함

객체를 여러 변수가 동시에 참조할 수 있다.

=(할당 연산자)는 객체에 대한 참조를 만들어 준다.

```
>>> a = 100
>>> b = a
>>> print(id(a))
4324024528
>>> print(id(b))
4324024528
```



C

변수 중심

Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
```

Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
```

a :

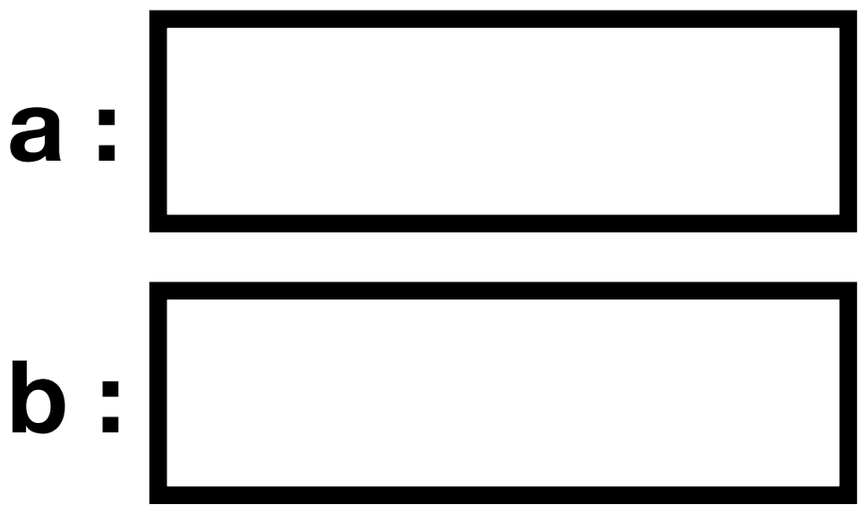
Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
```



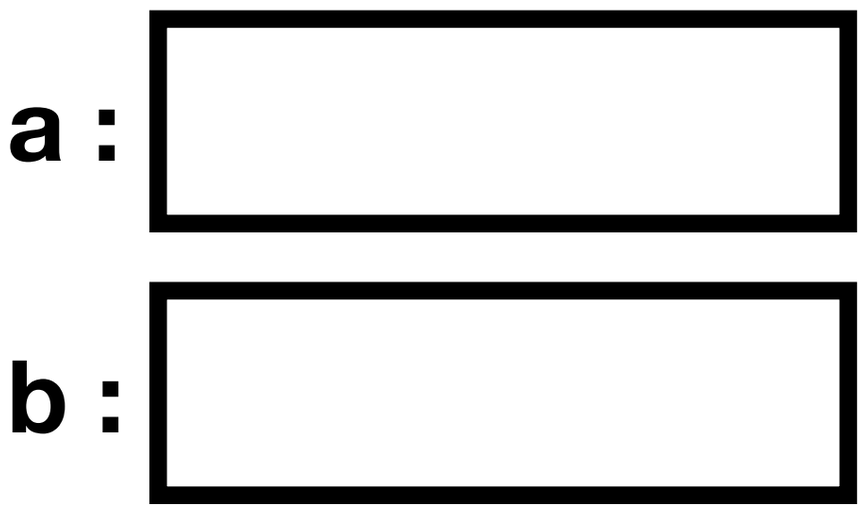
Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
```



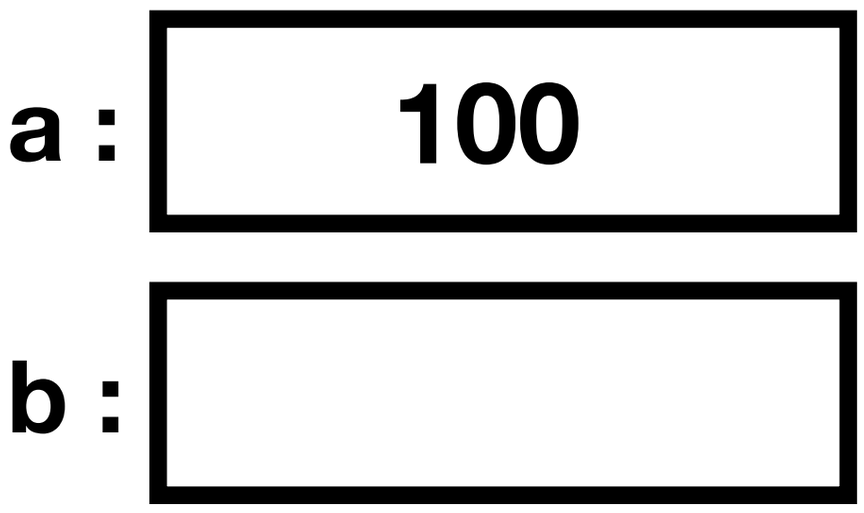
Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
```



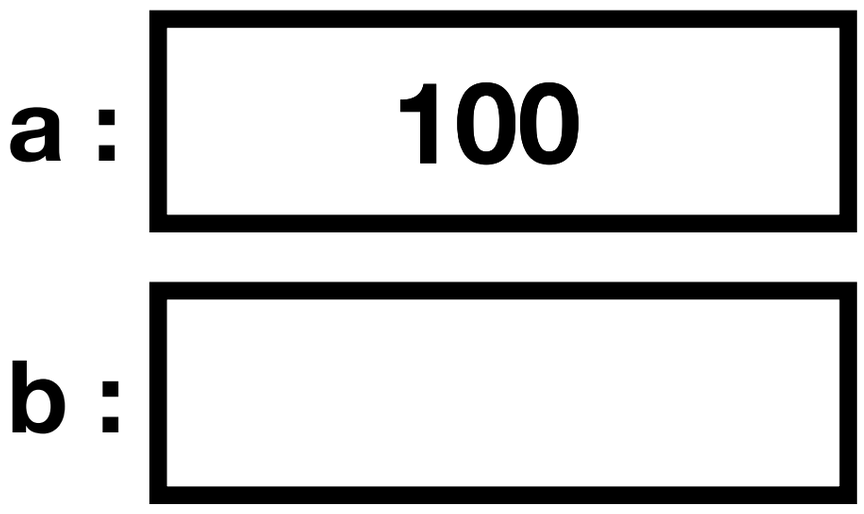
Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
```



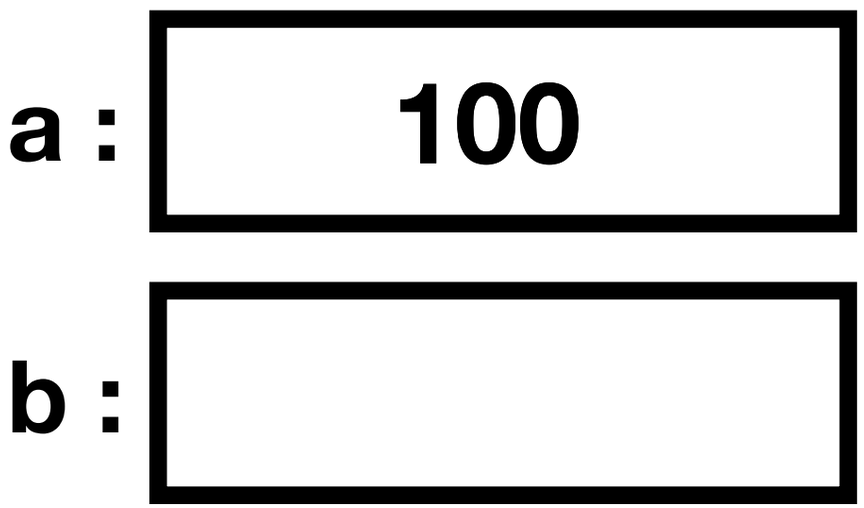
Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



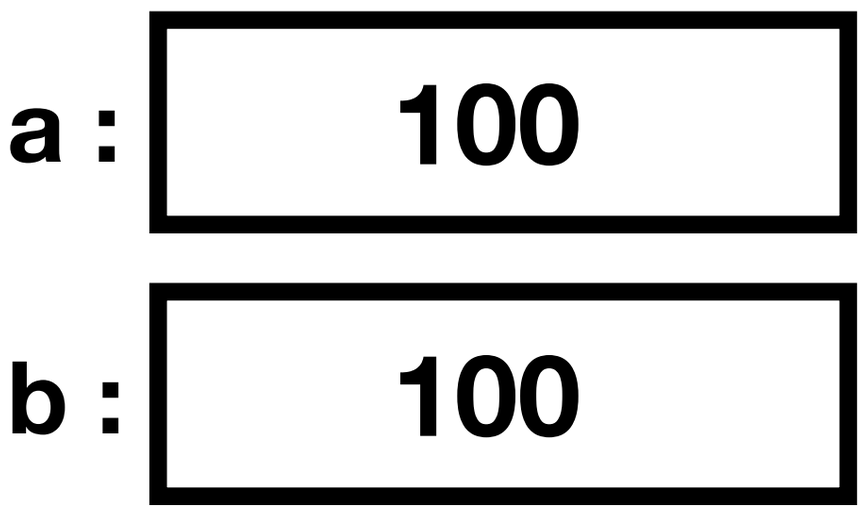
Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



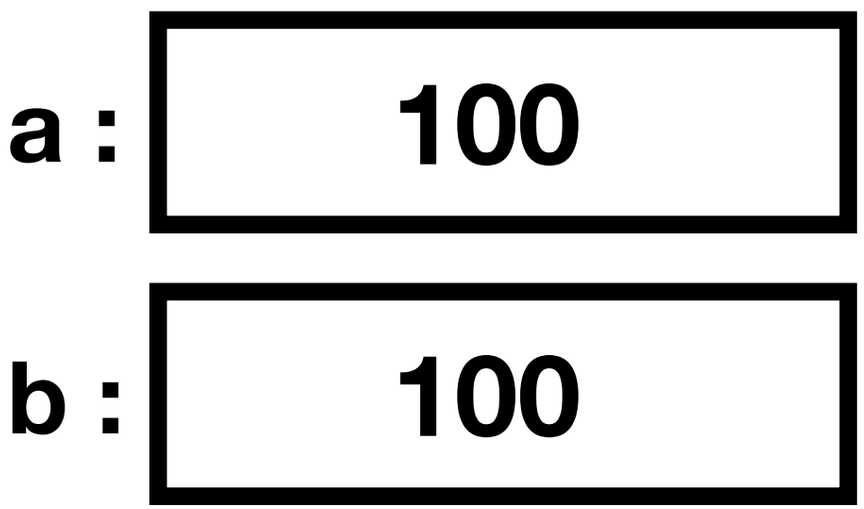
Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



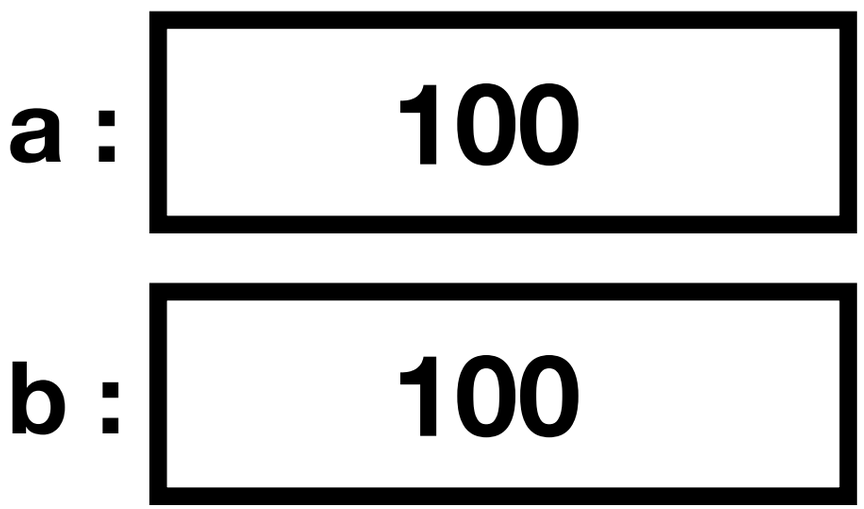
Python

객체 중심

C

변수 중심

```
int a, b;   // 정수형 변수 선언
a = 100;    // 정수값 할당
b = a;      // a의 값을 b에 할당
```



Python

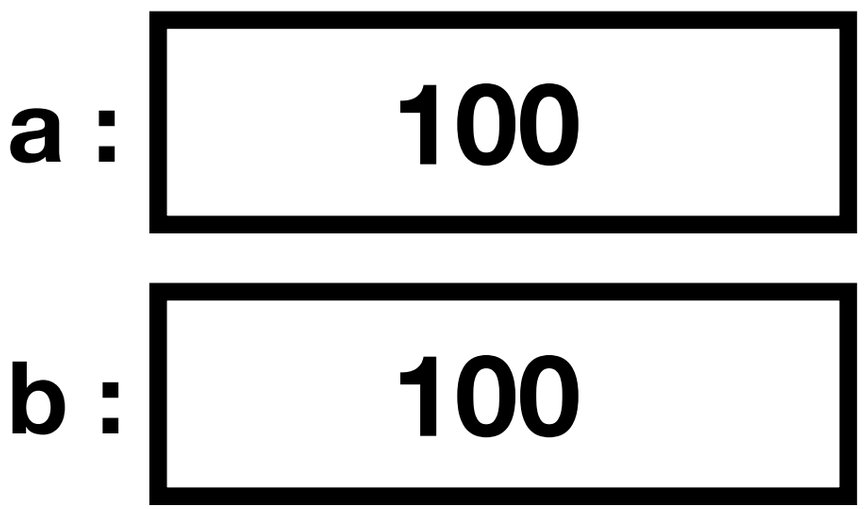
객체 중심

```
# 변수의 형 선언이 불필요함
```

C

변수 중심

```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



Python

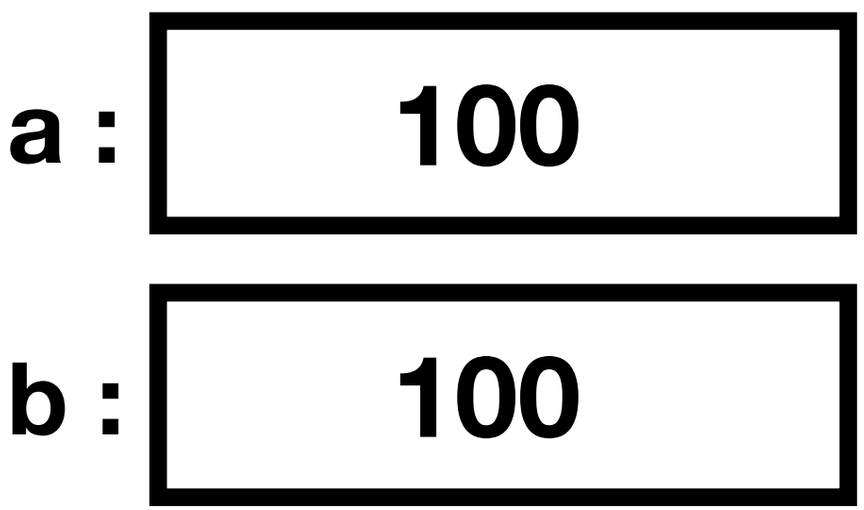
객체 중심

```
# 변수의 형 선언이 불필요함
a = 100 #  100객체 생성과 참조
```

C

변수 중심

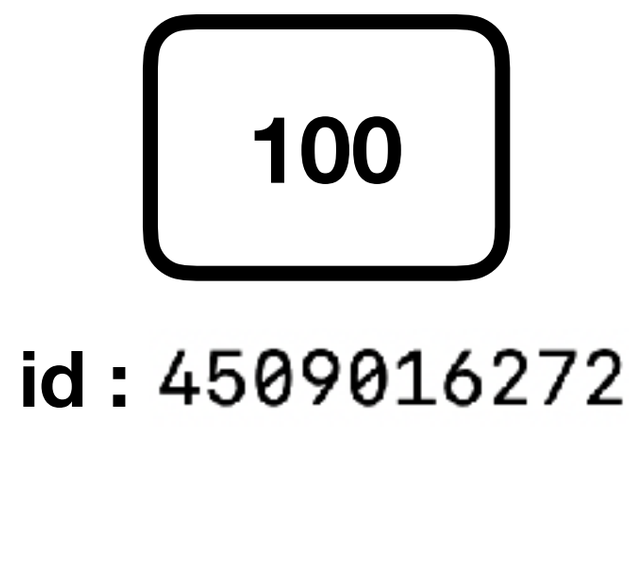
```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



Python

객체 중심

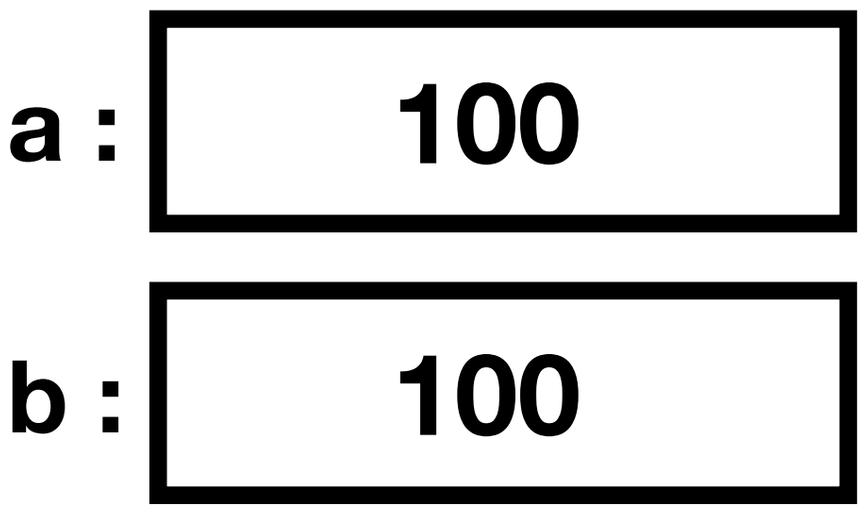
```
# 변수의 형 선언이 불필요함
a = 100 #  100객체 생성과 참조
```



C

변수 중심

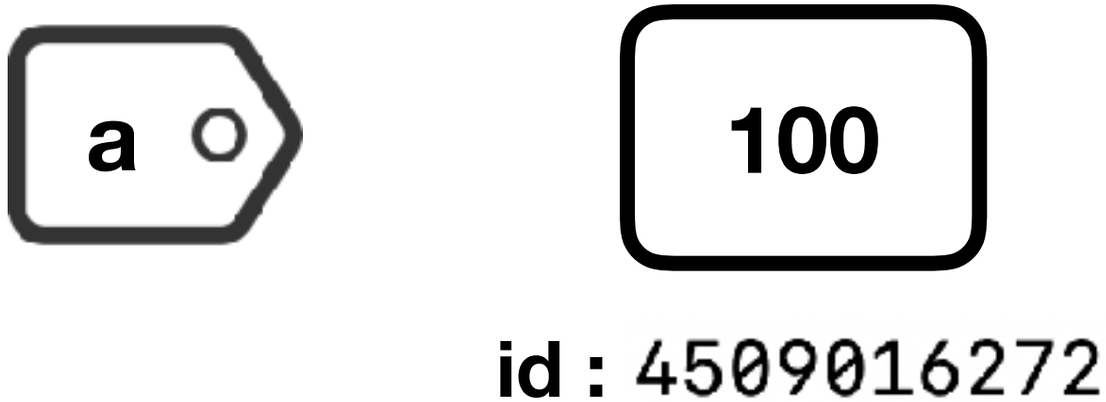
```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



Python

객체 중심

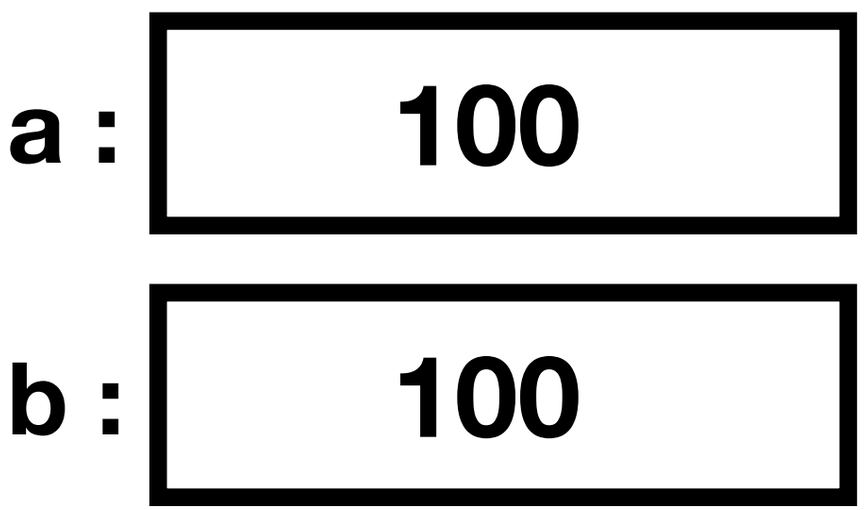
```
# 변수의 형 선언이 불필요함
a = 100 #  100객체 생성과 참조
```



C

변수 중심

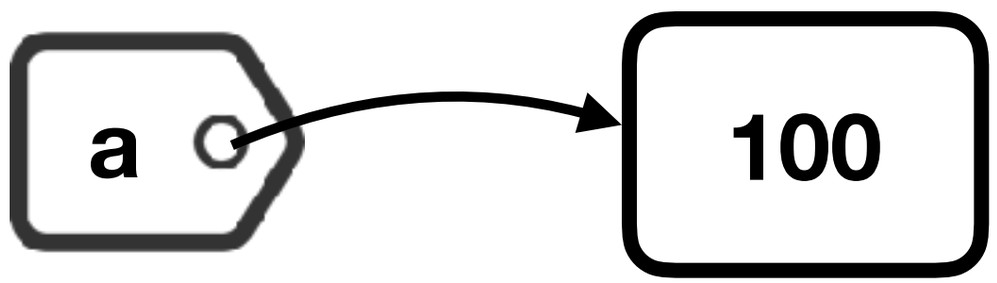
```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



Python

객체 중심

```
# 변수의 형 선언이 불필요함
a = 100 #  100객체 생성과 참조
```

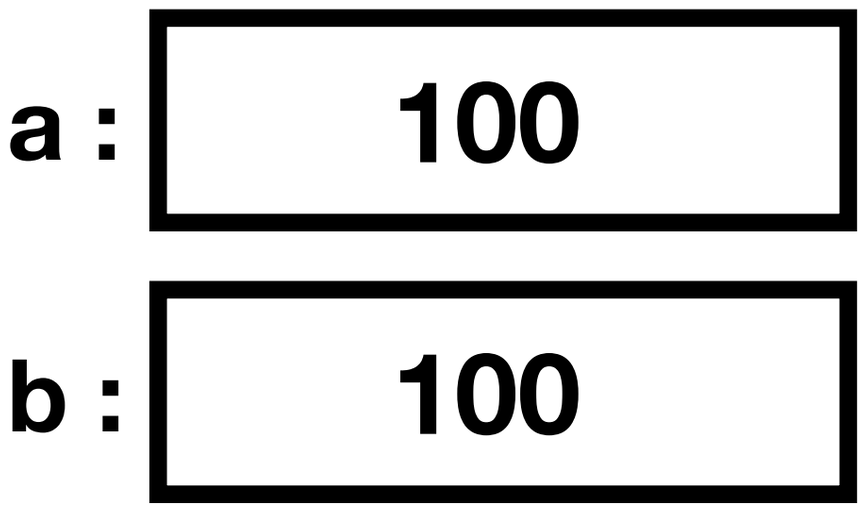


id : 4509016272

C

변수 중심

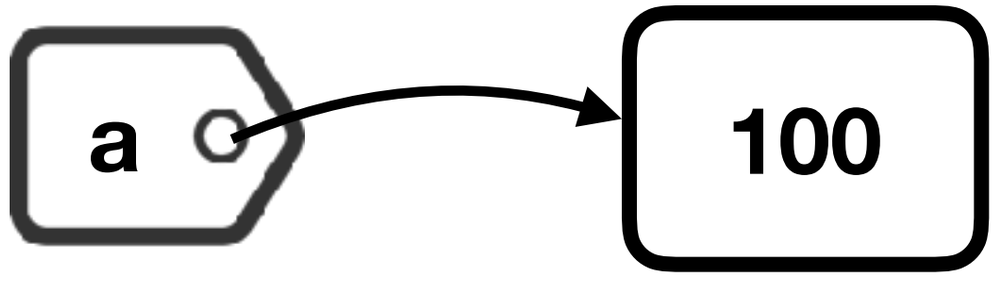
```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



Python

객체 중심

```
# 변수의 형 선언이 불필요함
a = 100 #  100객체 생성과 참조
b = a   # 100 객체를 참조
```

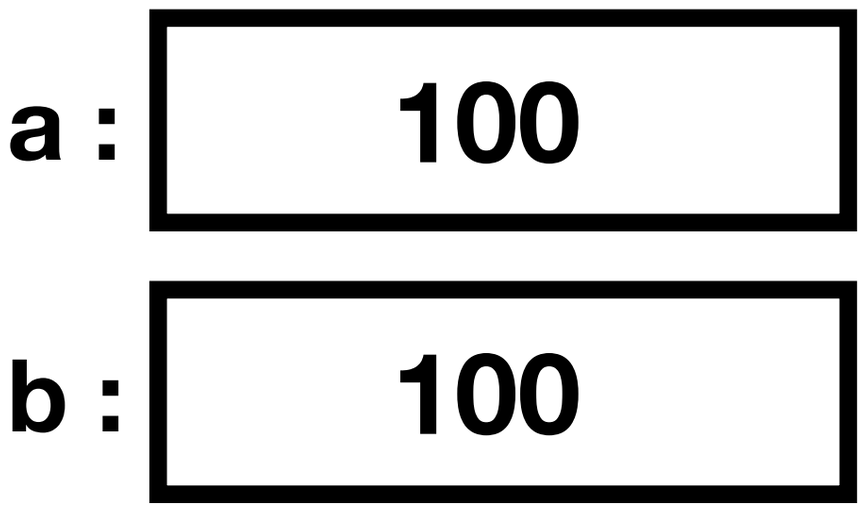


id : 4509016272

C

변수 중심

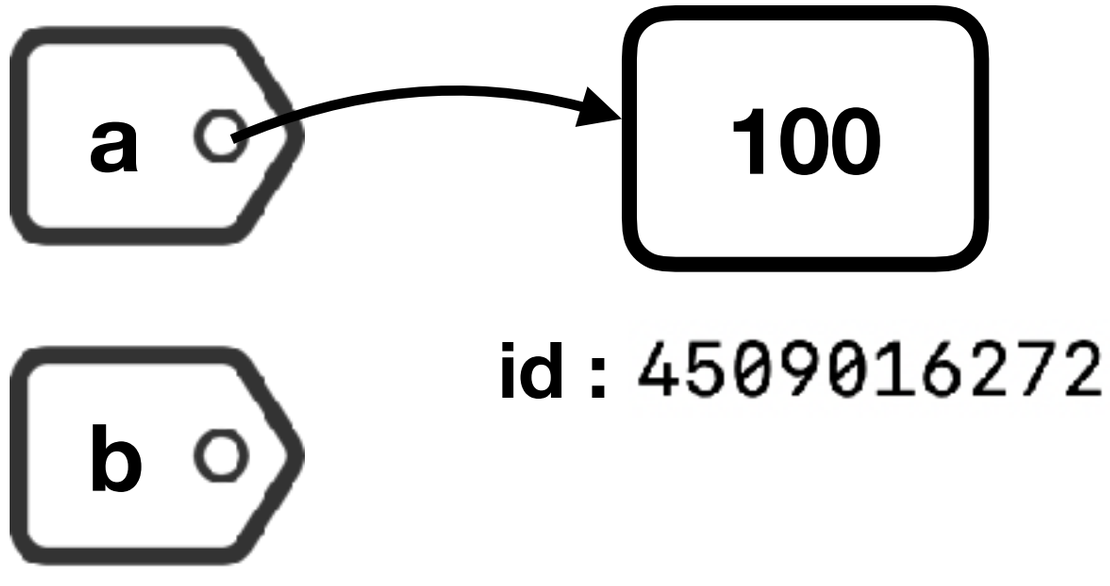
```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



Python

객체 중심

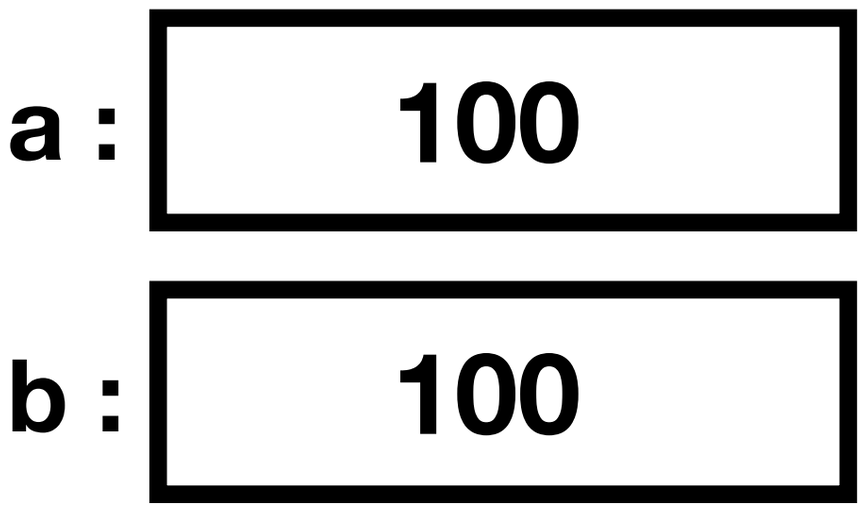
```
# 변수의 형 선언이 불필요함
a = 100 #  100객체 생성과 참조
b = a   # 100 객체를 참조
```



C

변수 중심

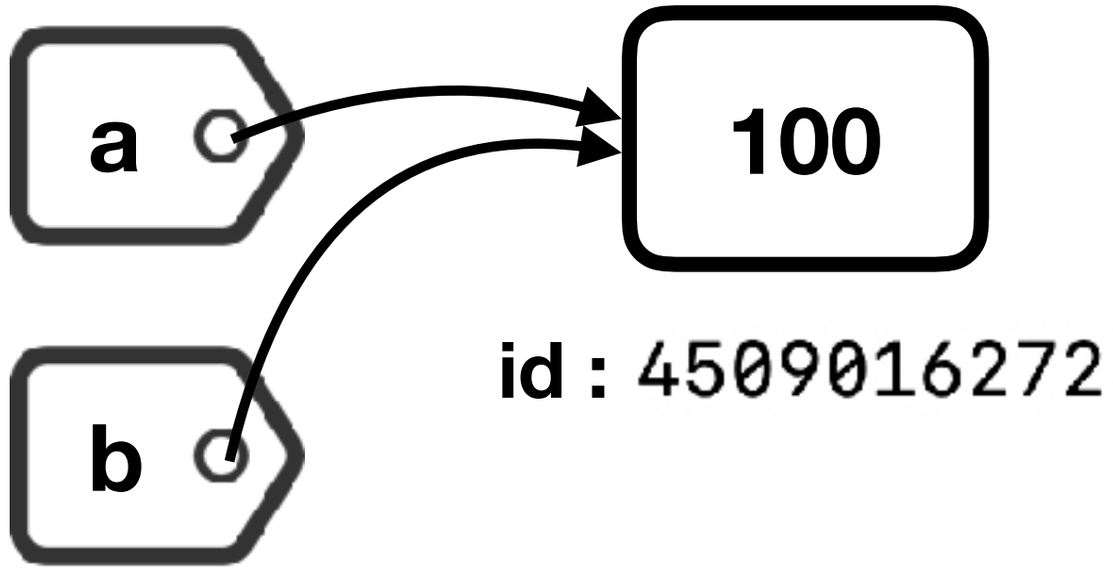
```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



Python

객체 중심

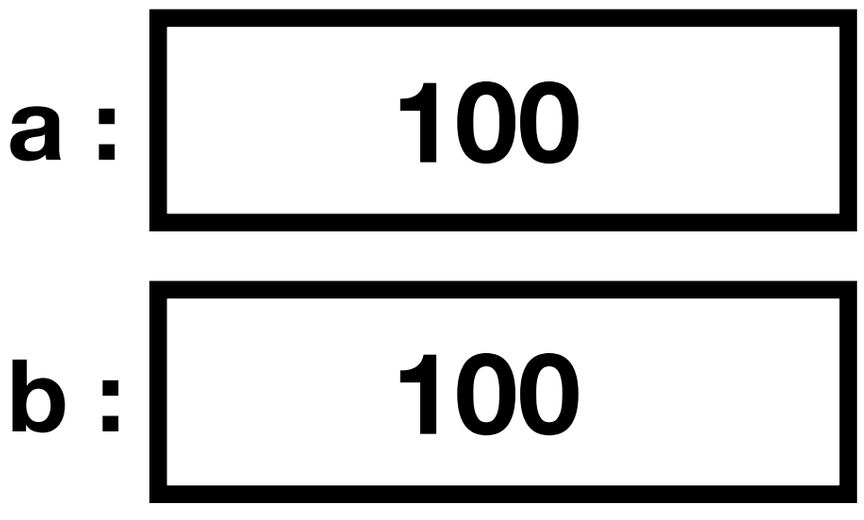
```
# 변수의 형 선언이 불필요함
a = 100 #  100객체 생성과 참조
b = a   # 100 객체를 참조
```



C

변수 중심

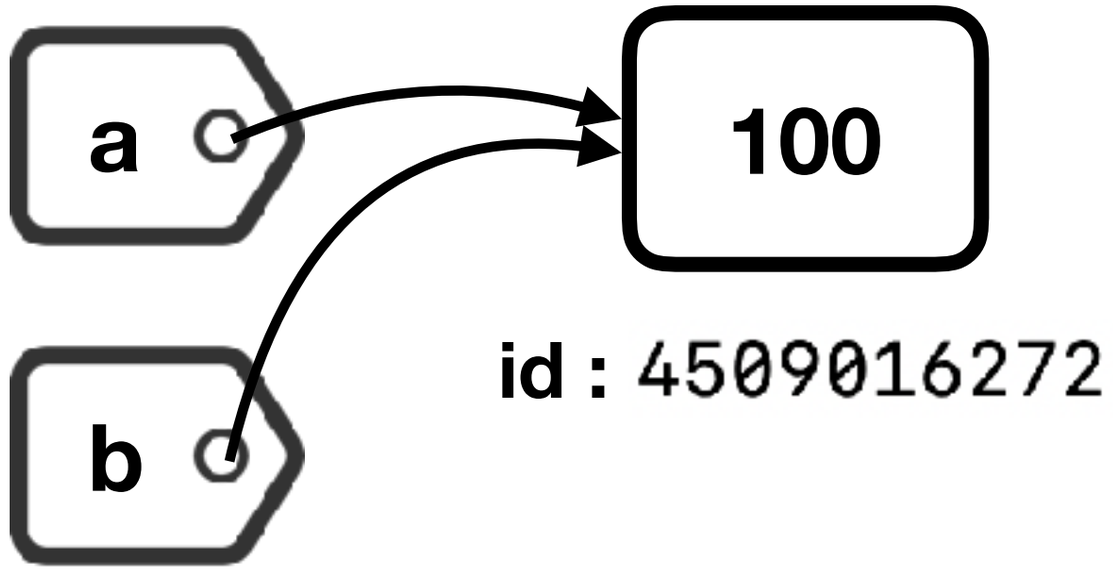
```
int a, b;   // 정수형 변수 선언
a = 100;       // 정수값 할당
b = a;    // a의 값을 b에 할당
```



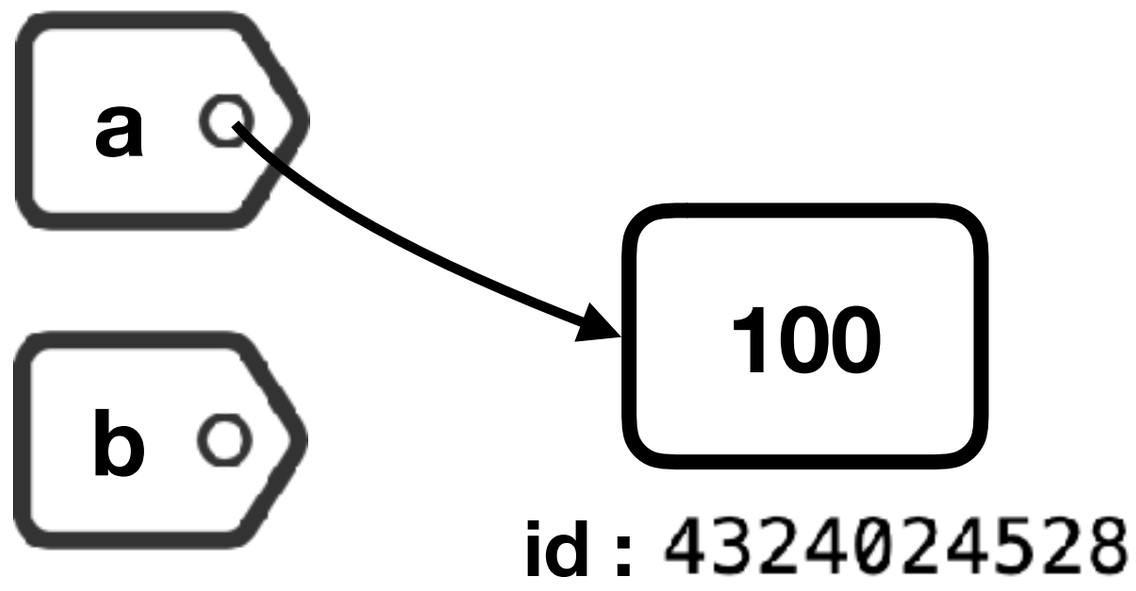
Python

객체 중심

```
# 변수의 형 선언이 불필요함
a = 100 #  100객체 생성과 참조
b = a   # 100 객체를 참조
```

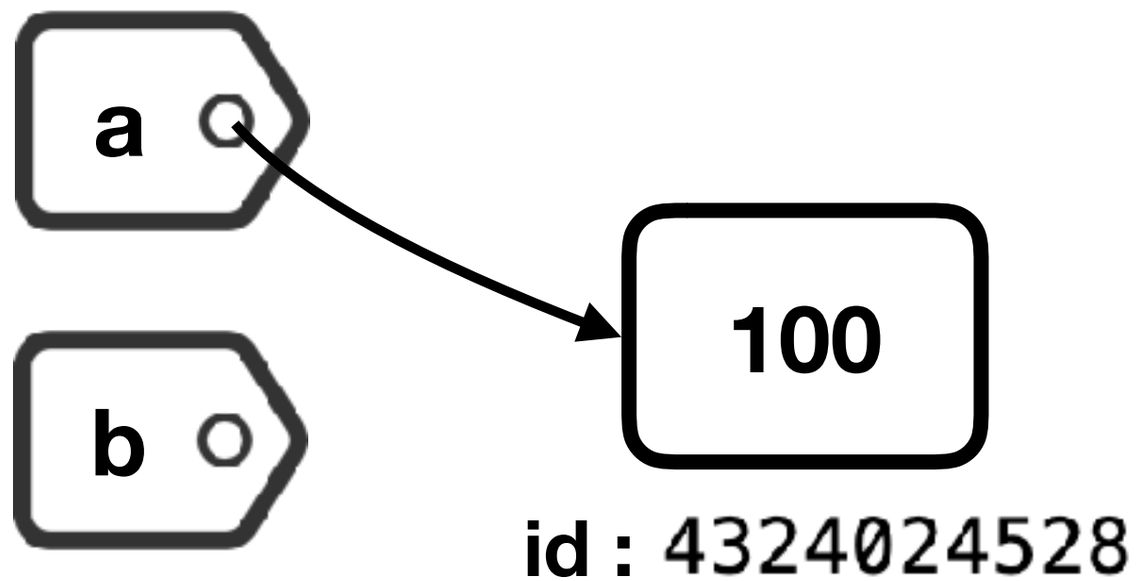


```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



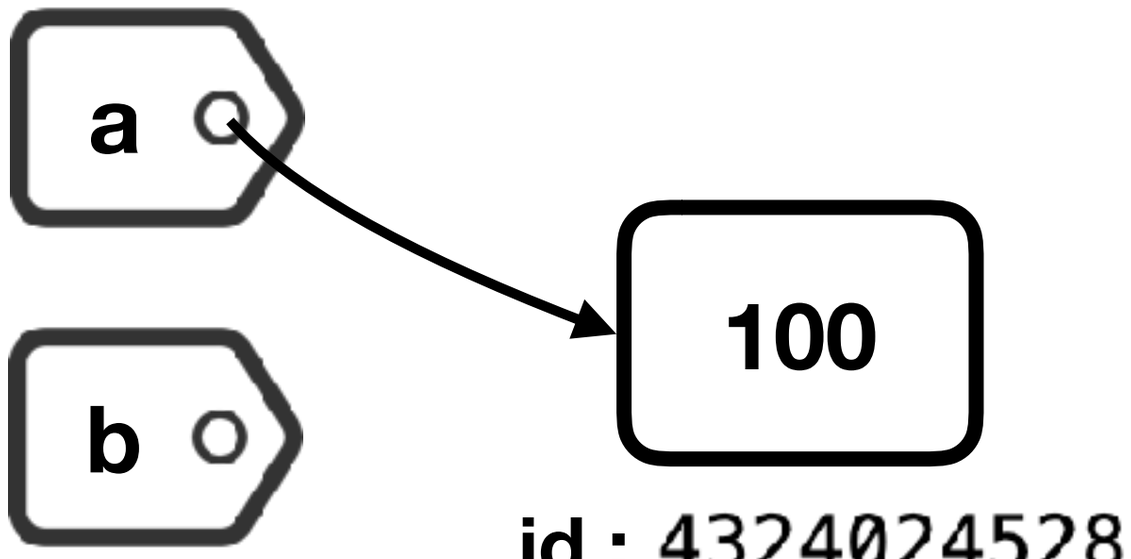
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



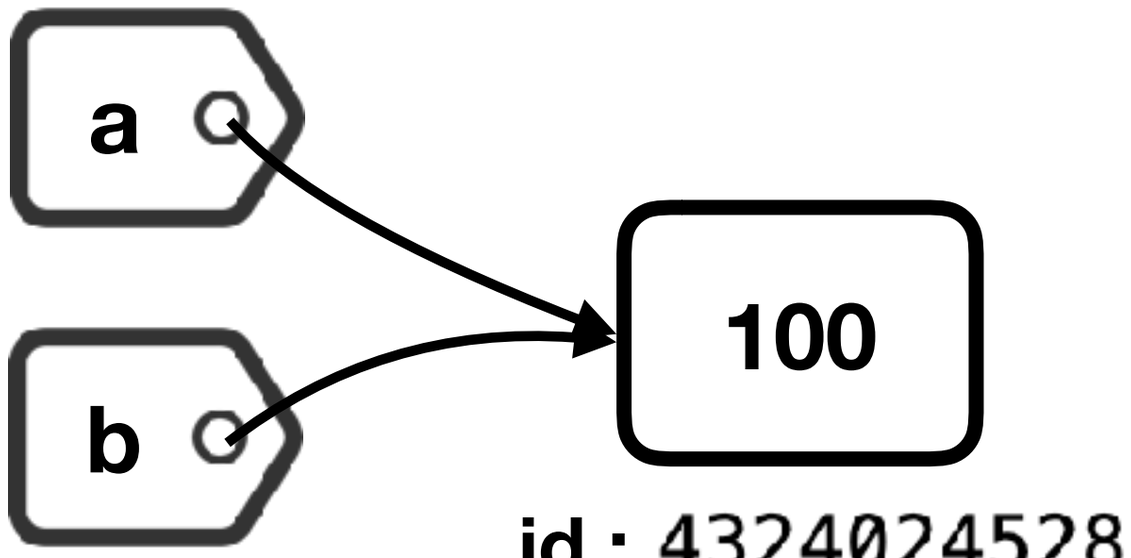
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



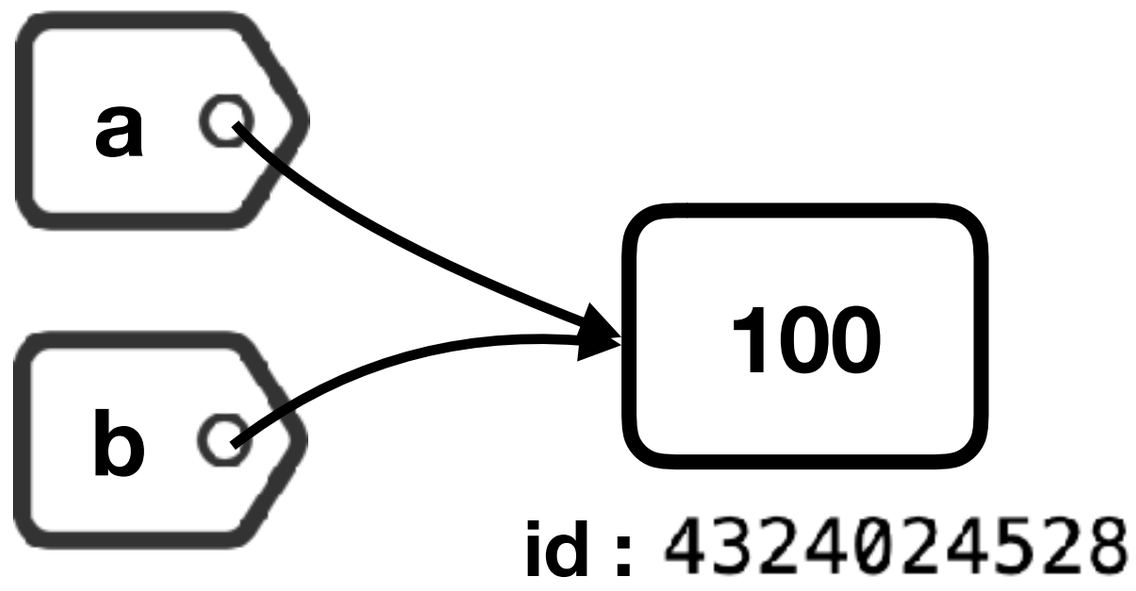
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



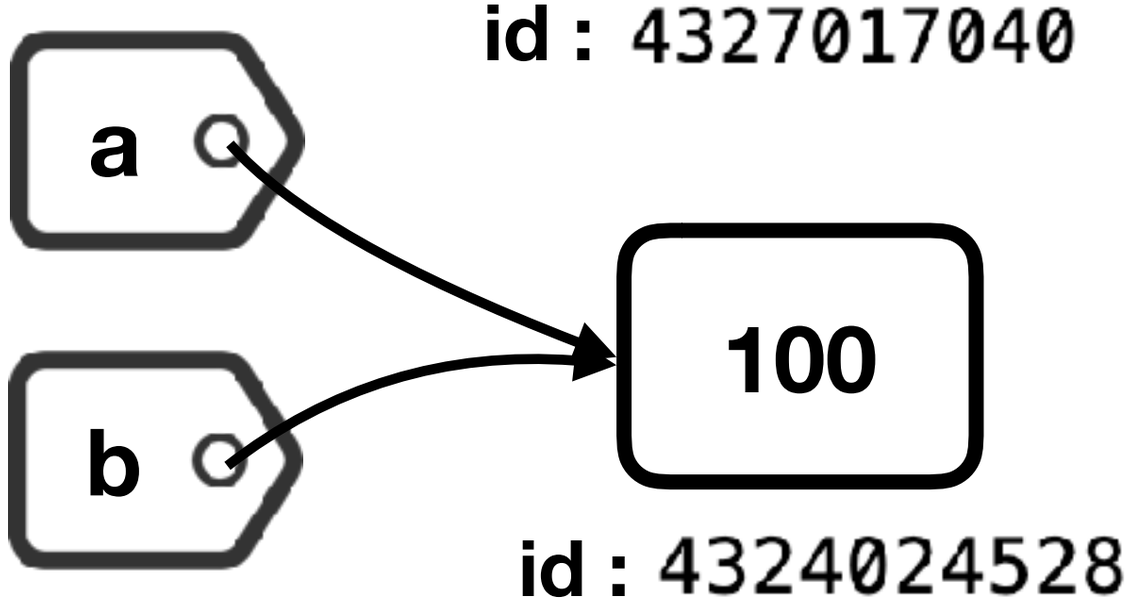
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



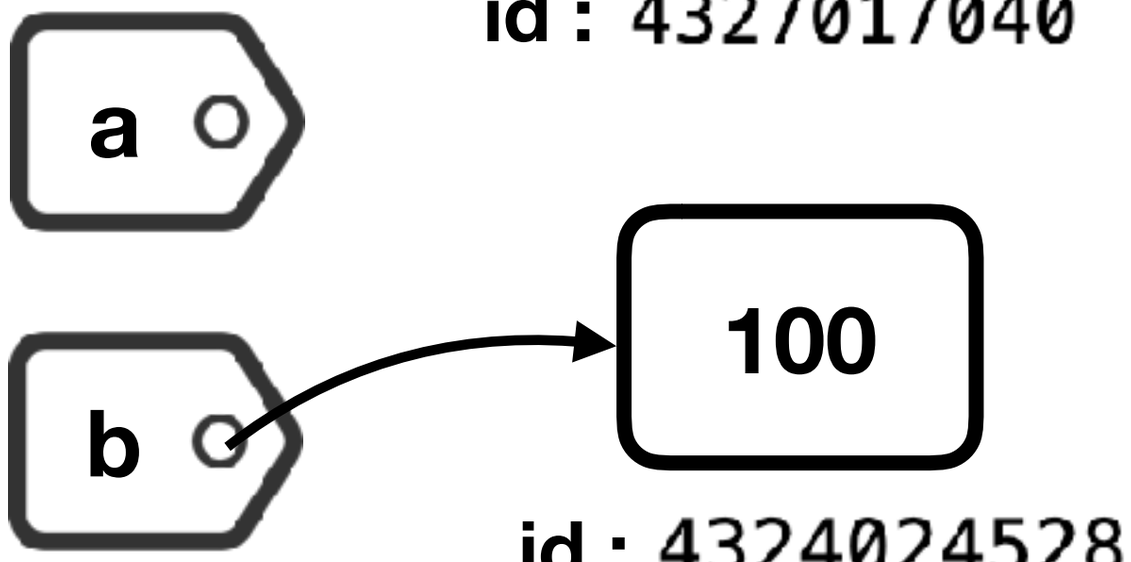
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



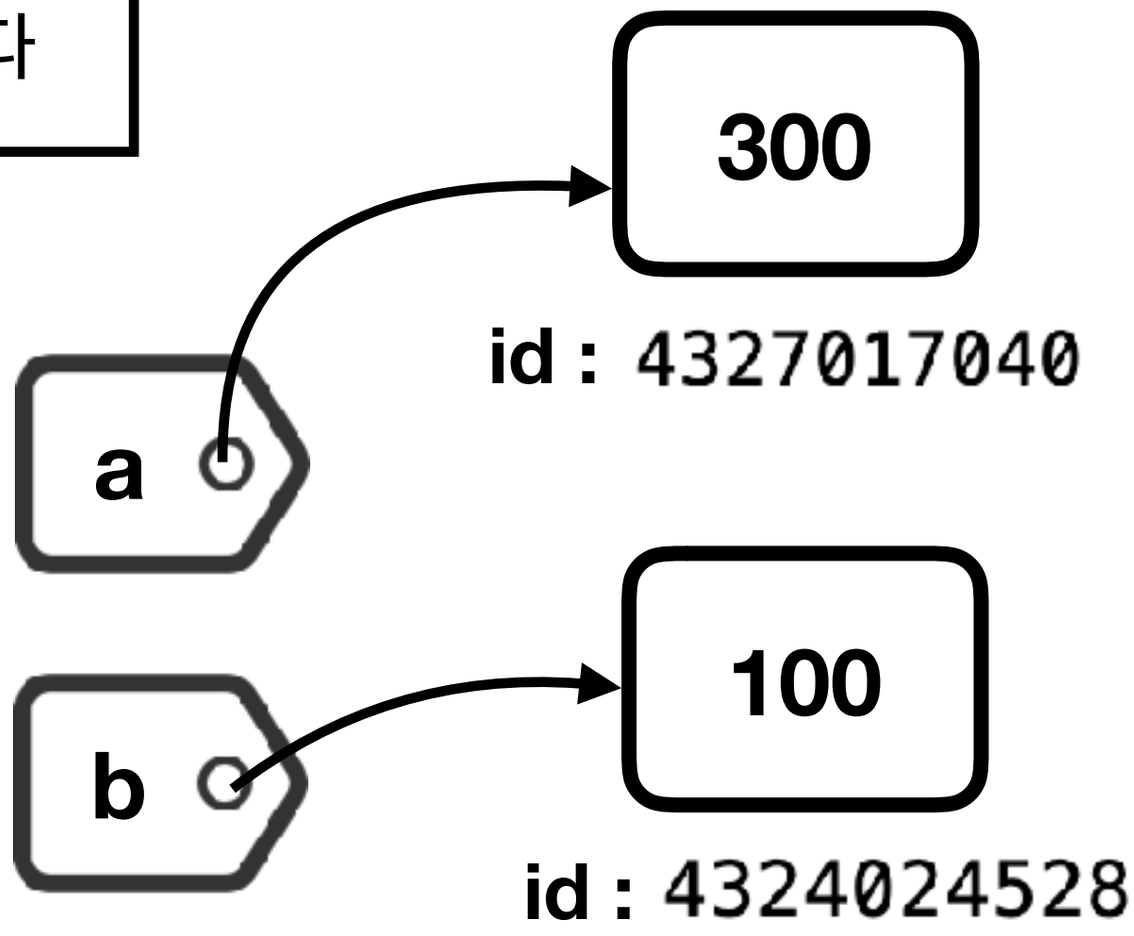
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



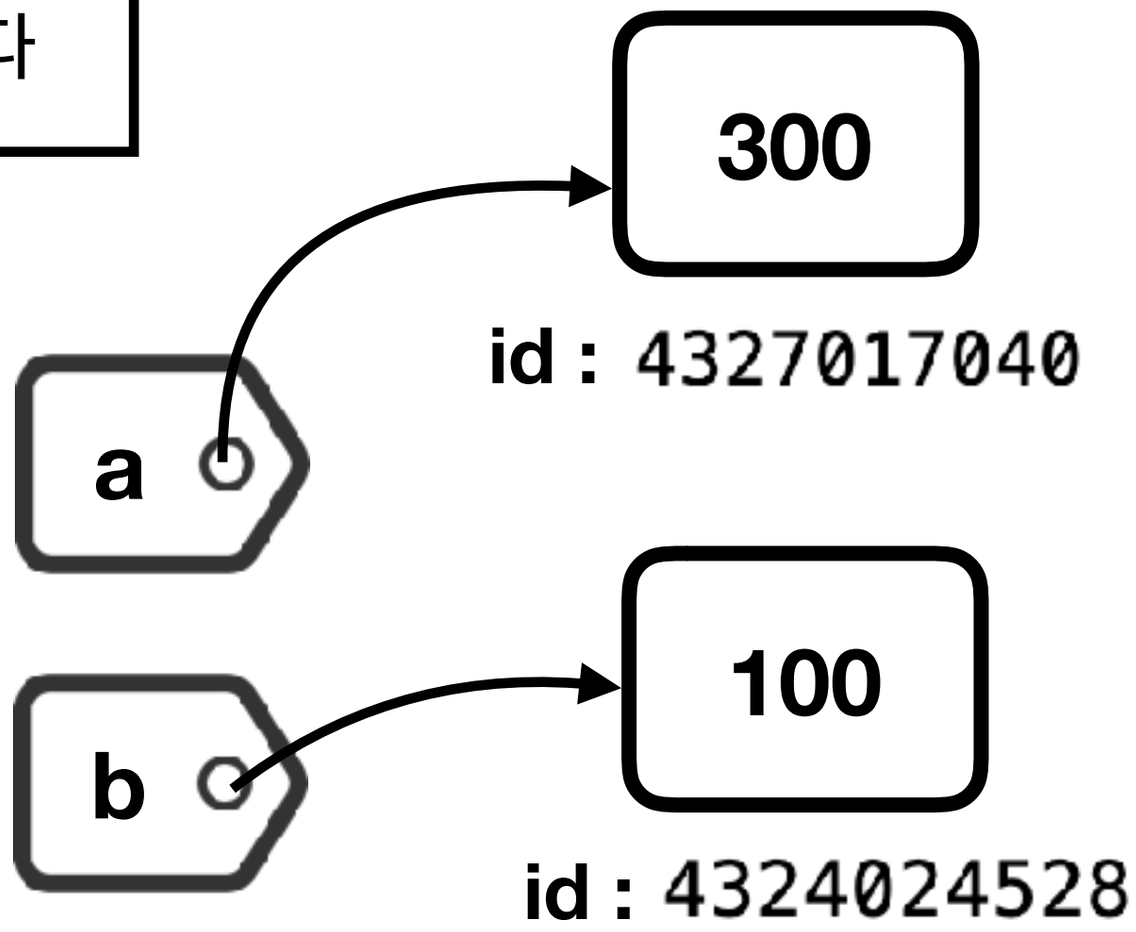
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



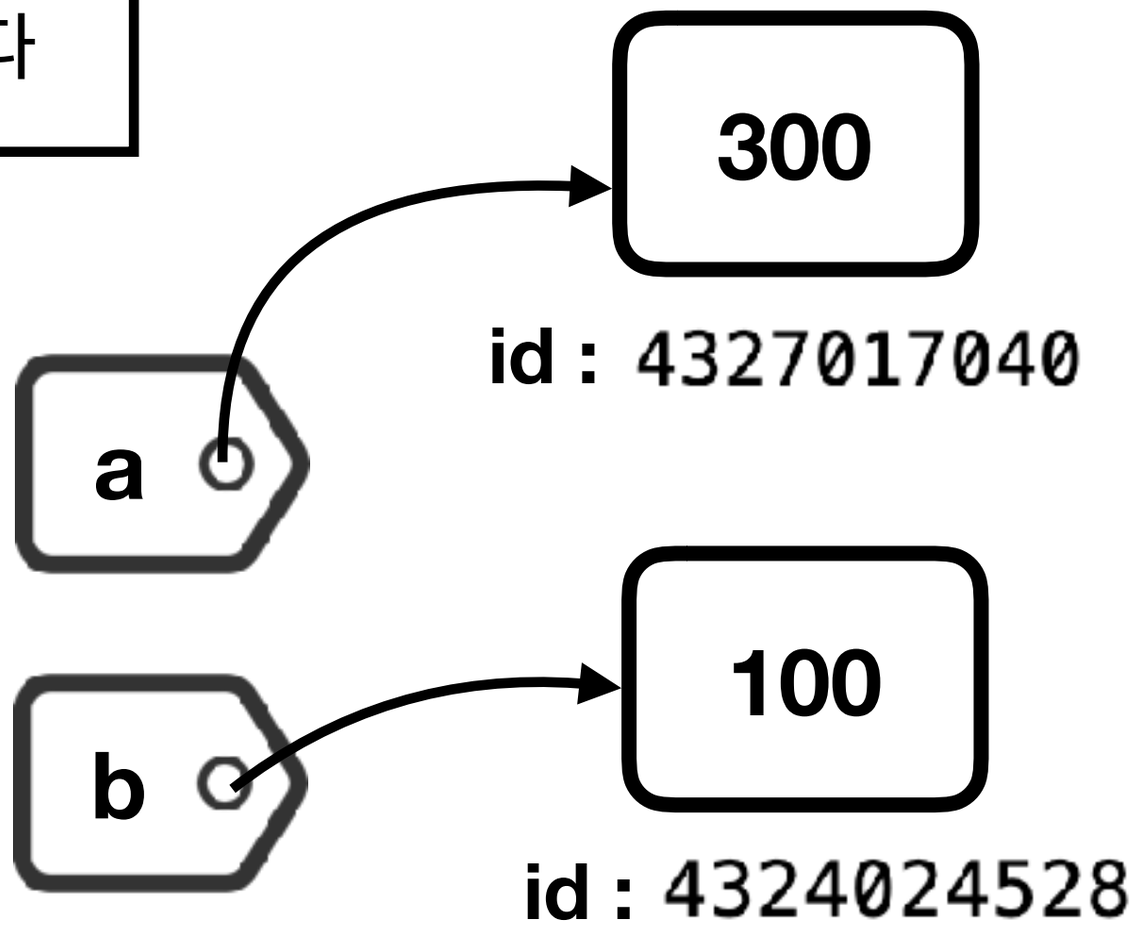
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



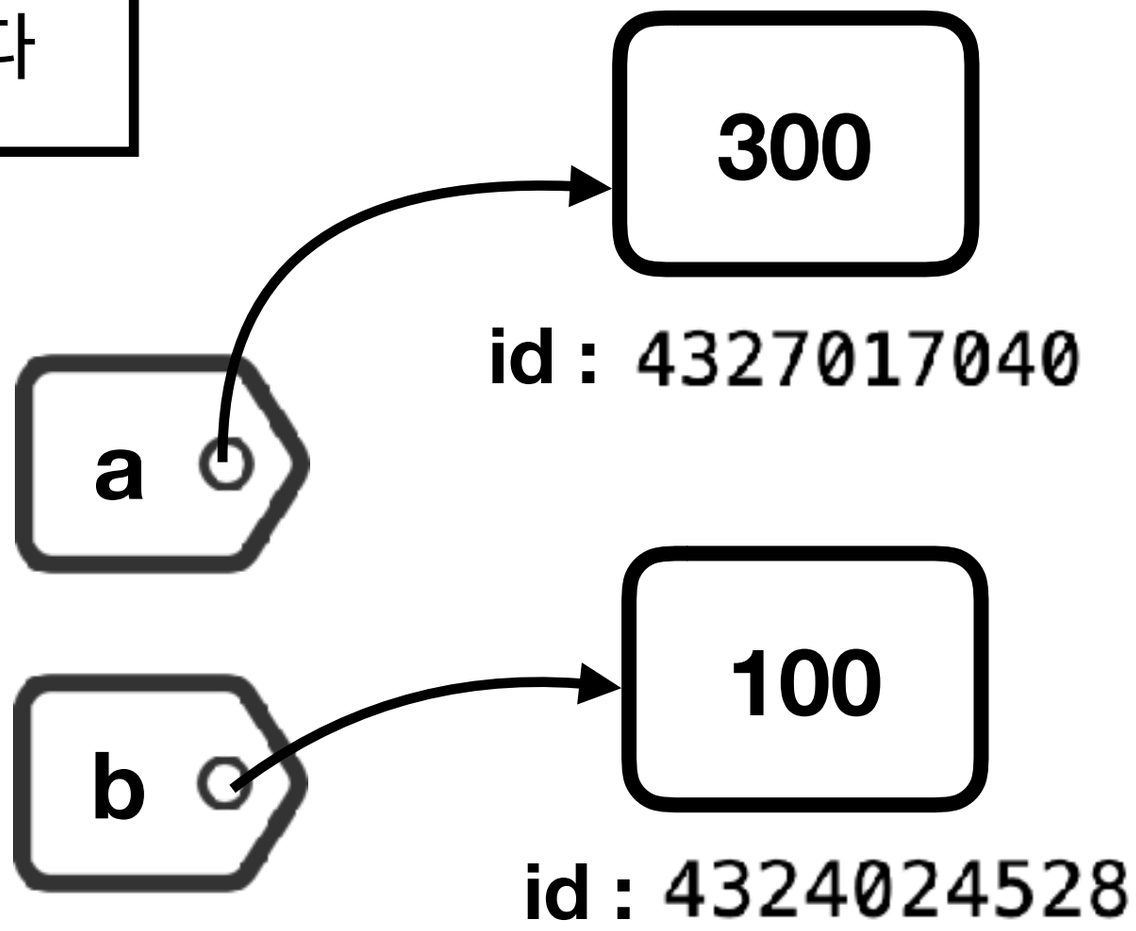
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



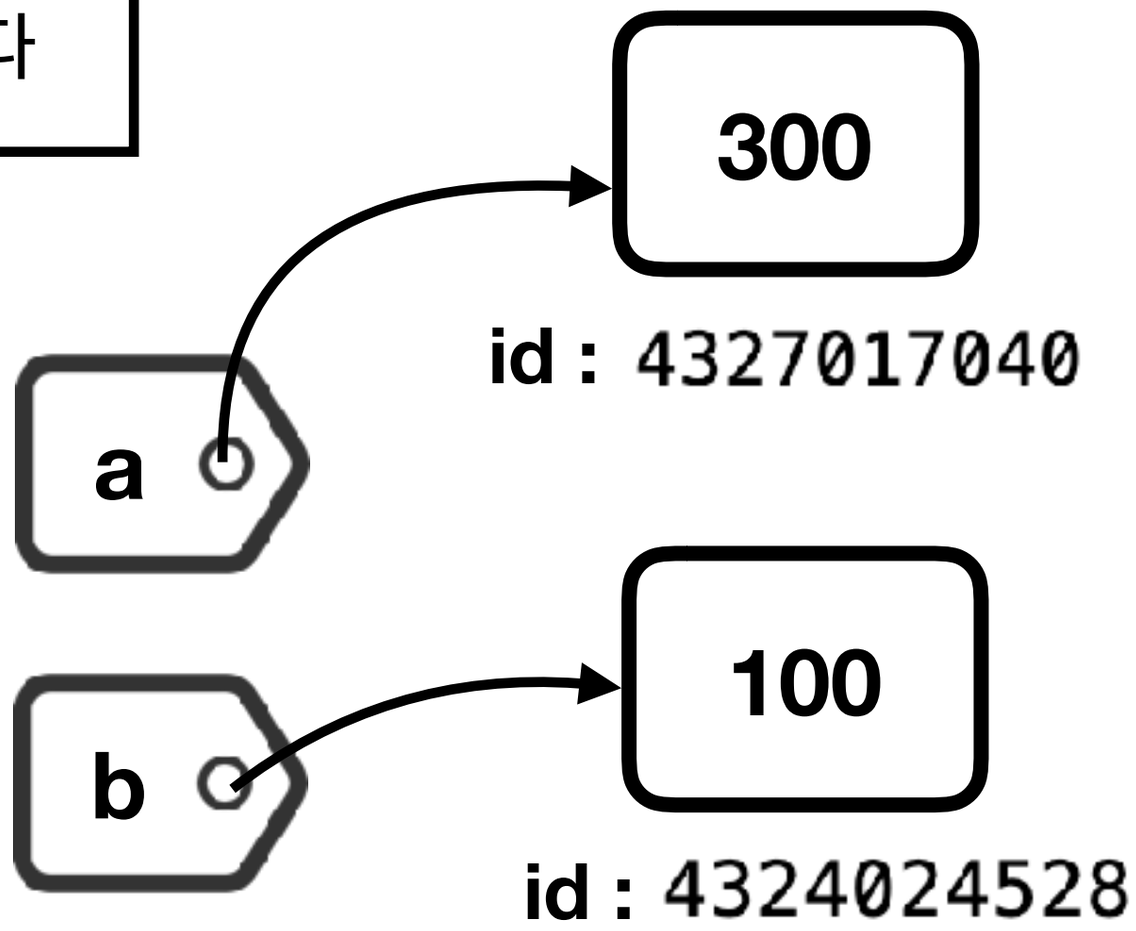
a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



a는 새로운 객체를
참조할 수 있다

```
>>> a = 100
>>> b = a
>>> a = 300
>>> print(id(a))
4327017040
>>> print(id(b))
4324024528
```



Lab

감사합니다.