산술 연산자

* 사칙연산: 덧셈(+), 뺄셈(-), 곱셈(*), 나눗셈(/)

다음 코드를 작성해 결과를 확인해 보자

```
print(3 + 9, 2 - 5, 9 * 7, 16 / 3)
```

* 그 외의 연산:

- 나눗셈의 몫(//)
- 나눗셈의 나머지(%)
- 제곱(**)

다음 코드를 작성해 결과를 확인해 보자

```
print(10 // 3, 11 % 2, 8**3)
```

관계/논리 연산자

❁ 관계연산: 비교하여 그 결과를 불리언(True/False)으로 리턴

같다(==), 다르다(!=)
크다(>), 작다(<), 크거나 같다(>=), 작거나 같다(<=)
다음 코드를 작성해 결과를 확인해 보자

```
print(1 == 2, 1 != 2, 1 > 2, 1 < 2, 1 >= 2, 1 <= 2)
```

❁ 논리연산: 관계연산과 함께 사용하는 경우 많음

and: 논리곱, 둘 다 참 이어야 참
or: 논리합, 둘 중 하나가 참 이면 참
not: 참/거짓의 역전
다음 코드를 작성해 결과를 확인해 보자

```
print(True and False, True or False, not True)
```

단항/ 비트 연산자

* 단항연산: 양, 음 부호 표시(+/-)
  숫자에 붙여서 쓴다 (-3)

* 비트연산: 산술연산과 달리 비트 단위로 연산

&(AND): 두 비트가 모두 1일 때 1
|(OR): 두 비트 중 하나라도 1이면 1
^(XOR): 두 비트가 서로 다르면 1, 같으면 0
~(Complement): 0은 1, 1은 0
<<, >>(Shift): 지정한 수만큼 왼쪽, 오른쪽 자리 이동

할당/ 복합 연산자

* 할당(대입)연산(=): 오른쪽 값을 왼쪽에 할당한다

  my_num = 10

* 복합연산: 산술연산과 할당연산의 결합

+=:	a = a + b	->	a += b
-=:	a = a - b	->	a -= b
*=:	a = a * b	->	a *= b
/=:	a = a / b	->	a /= b
%=:	a = a % b	->	a %= b
**=:	a = a ** b	->	a **= b
//=:	a = a // b	->	a //= b

알아 두면 좋은 기호

- **괄호:**
  - 소괄호( ): 함수 인자 묶음, 튜플 자료형
  - 중괄호{ }: 딕셔너리, 집합 자료형에 사용
  - 대괄호[ ]: 리스트 자료형에 사용
- **콜론(:)**
  - 조건문, 반복문, 함수 정의의 시작
  - 리스트, 튜플의 슬라이싱에 사용
- **세미콜론(;)**
  - 한 줄에 여러 개의 코드문을 적고 싶을 때 사용
- **들여쓰기(스페이스*4)**
  - 조건문, 반복문, 함수 정의 시 블록 단위로 구분지어 줌

변수 만들기

변수이름 = 초기값

- 영문자, 숫자, 언더스코어(_)
- 시작 문자에 숫자 사용 못 함
- 공백 허용 안 됨
- 띄어쓰기 할 곳에 언더스코어
- 대소문자 구분함
- 파이썬 예약어 사용 불가

```
import keyword
print(keyword.kwlist)
```

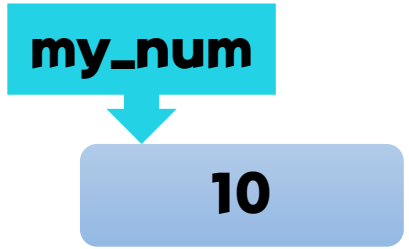
할당 연산자:
오른쪽 값을
왼쪽과 연결

- 초기값이 자료형 결정
- 초기값 없이 선언 불가
- None: 타입 지정 없이 초기화

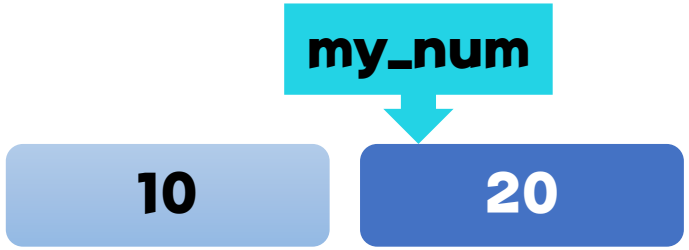
* 변경 필요 없는 값(ex. 원주율)을
  다른 언어는 상수로 구분하여 선언
* 파이썬은 따로 구분 x
  대신 구분 위해 대문자로 선언

변수 이름과 메모리

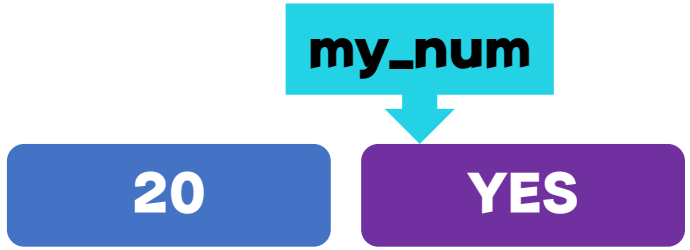
my_num = 10



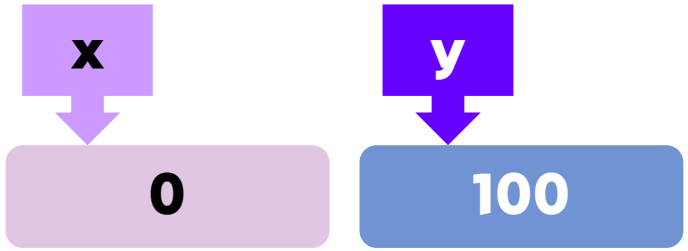
my_num = 20



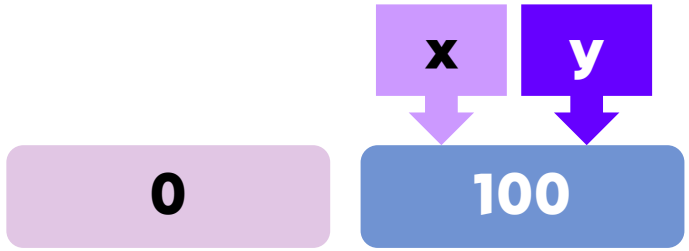
my_num = 'YES'



x, y = 0, 100



x = y



type 함수

- 반환값(함수 호출의 결과값): 인자의 type을 리턴

- type은 <class '자료형'>으로 표시됨

type(인자) -> <class '인자의 자료형'>

type(123) -> <class 'int'>
type('123') -> <class 'str'>
type("yes") -> <class 'str'>
type(3.14) -> <class 'float'>
type(True) -> <class 'bool'>

print 함수

- 출력함수

- 형식

  print(인자1, 인자2, ... , 인자n) -> 인자1 인자2 … 인자n

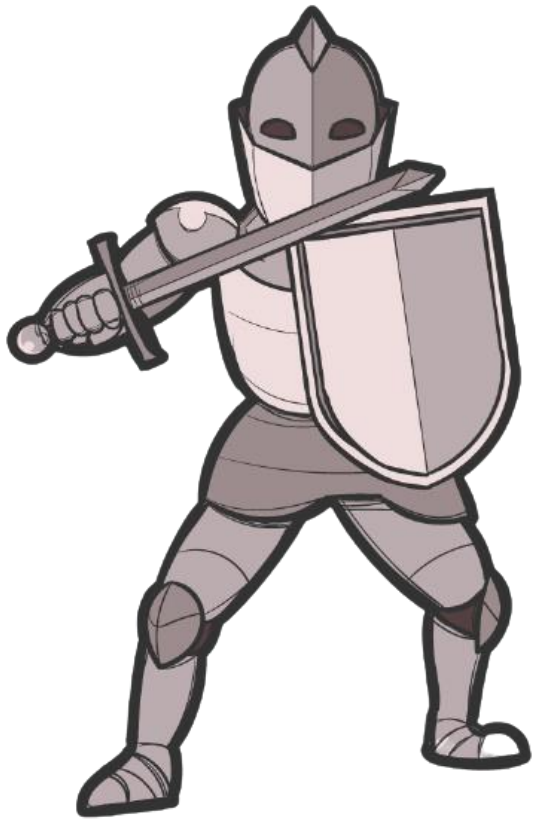
- 인자로 숫자의 연산식을 넣은 경우 연산 먼저 수행

  print(3 + 4 * 5) -> 23

- print함수의 인자로 함수를 호출할 수도 있다

  print(type(123)) -> <class 'int'>

변수 만들기 예시



변수 종류	변수 자료형 타입	예
이름	<class str> , 문자열	char_name = '소드마스터 '
HP, MP	<class int>, 정수형	HP, MP = 100, 100
위치	<class float>, 실수형	char_pos_x, char_pos_y = 0, 0
인벤토리	<class dict>, 딕셔너리형	inventory = {'HP포션': 10, 'MP포션':3}
보유 스킬	<class list>, 리스트형	skills = ['가로베기', '세로베기', '폭풍검']

변수 타입 바꾸기

- 자료형을 이름으로 하는 함수 사용

자료형(인자)

num = 123

str(num) -> <class 'str'> : '123'

int(num) -> <class 'int'> : 123

float(num) -> <class 'float'> : 123.0

주석 달기

- 한 줄 주석:

  # 주석문

- 여러 줄 주석:

  ''' 주석문1
  주석문2
  …
  주석문n'''

  """주석문1
  주석문2
  …
  주석문n"""

- 함수를 설명하기 위한 주석을 docstring이라 하며
  이 경우에는 큰 따옴표 3개를 사용한다

연습 문제 - 변수

- 이름 va_1, va_2, va_3, va_4, va_5인 변수를 선언하고
  각각 10, 45.495, "각자의 이름(한글)", False, '12345'를 할당한다

- va_4의 값과 자료형을 출력한다

- va_1 * 5의 값을 출력한다

- print함수를 한 번 써서 va_1, va_2, va_3를 출력한다

- 코드 한 줄로 va_5를 정수형으로 변환하고
  그 값에서 123을 뺀 값을 출력한다