Chapter 3
Operator

Operator là các hàm nội tại (special method)

Mỗi operator như `+` , , `in` , `is` , ... tương ứng với **phương thức đặc biệt của object** như `__add__` , `__contains__` , `__eq__` , `__is__` , v.v. Python gọi `a + b` thực chất là `a.__add__(b)` hoặc `b.__radd__(a)` nếu hàm trái không hỗ trợ. Đây là cách operator tương tác với kỹ thuật **operator overloading** (Python documentation).

Các nhóm Operator

Arithmetic Operators (số học)

Bao gồm: `+` , , , `/` , `//` , `%` , `*` , `@` (matrix multiplication)

Thực thi theo ngữ nghĩa toán học, kết quả trả về là object numeric tương ứng (`int` , `float` , `complex`)

`/` luôn trả float, `//` trả floor (làm tròn xuống), `%` là phép dư, `*` lũy thừa, `@` dành cho tính nhân ma trận (NumPy/array).

Comparison Operators

`==`, `!=`, `<`, `>`, `<=`, `>=`, hỗ trợ **chaining** như `a < b < c`, đánh giá từ trái sang phải, dừng sớm nếu false (short-circuit).

`is`, `is not`: so sánh **identity**, tức có cùng object reference hay không, không quan tâm value.

Logical Operators

`and` , `or` , `not` : dùng trong biểu thức logic, có **short-circuiting**: `and` dừng nếu False, `or` dừng nếu True. Trả về **operand cuối cùng được đánh giá**, không nhất thiết True/False nếu operand không phải bool.

Bitwise Operators

`&`, `|`, `^`, `~`, `<<`, `>>`: hoạt động trên từng bit của integer, thực hiện các phép logic bitwise/classic shift.

Assignment Operators

Ngoài `=` cơ bản, còn có: `+=` , `=` , `=` , `/=` , `//=` , `%=` etc. được gọi là **in-place operators**, thực chất là gọi các hàm như `operator.iadd(a, b)` tương đương `a = a + b` .

Membership Operators

`in`, `not in`: kiểm tra xem operand bên trái có là thành viên trong iterable hoặc container object (list, set, dict keys, string...).

Precedence & Associativity (Ưu tiên và kết hợp)

Python quy định thứ tự ưu tiên giữa các nhóm operator: ví dụ `*` > unary `+-` > `/ % //` > `+ -` > shifts > bitwise > comparison > logical.

Associativity: hầu hết binary operator là **left-associative**, trừ `*` là right-associative. Chaining comparison là đặc biệt (ở giữa) (scholarhat.com).

Unary vs Binary operators

Unary operators: hoạt động trên một operand đơn (`a` , `+a` , `not a` , `~a`). Negation, unary plus, logical not hoặc bitwise not (medium.com).
Binary operators: có hai operand (`a + b` , `a is b` , `a in b` , v.v.).

Bản chất runtime: dynamic dispatch và overloading

Python sử dụng dynamic typing: loại operand có thể thay đổi tại runtime, chọn implementation phù hợp dựa vào type object.

Nếu operand không hỗ trợ operator, Python sẽ gọi `__rxxx__` của operand còn lại hoặc raise `TypeError` nếu không hỗ trợ tồn tại (Python documentation, jakevdp.github.io).

Tóm tắt

Nhóm Operator	Cú pháp	Bản chất xử lý
Arithmetic	`a + b`, `a * b`, `a // b`,...	Gọi `__add__`, `__mul__`, `__floordiv__`,... tạo numeric object
Comparison	`==`, `<`, `is`, chaining	So sánh value hoặc identity, hỗ trợ chaining và short-circuit
Logical	`and`, `or`, `not`	Dùng truthiness, evaluate điều kiện ngắn, trả operand cuối
Bitwise	`&`, `|`, `^`, `~`, `<<`, `>>`	
Assignment	`=`, `+=`, `**=`, `//=`,...	Gán biến, in-place gọi hàm như `iadd`
Membership	`in`, `not in`	Dùng `__contains__`, kiểm tra chứa phần tử

Kết luận

Operator trong Python không chỉ là kí hiệu cú pháp, mà hoạt động như **phương thức đặc biệt (special method)** bên trong object.

Mọi phép toán đều dynamic dispatch, phụ thuộc type của operand.

Bản chất xử lý là delegate đến phương thức `__op__` tương ứng, tuỳ loại operator (arithmetic, comparison, bitwise, logic, membership...).

Python hỗ trợ short-circuit, chaining, operator overloading và evaluate các biểu thức đúng thứ tự ngữ nghĩa và ngữ cảnh.

cheers

cảm ơn

thank you!

muchas gracias

dziękuję

danke