

BÀI 8: PHÉP TOÁN SO SÁNH VÀ CÁC PHÉP TOÁN LOGIC

Mục tiêu bài học:

Nắm vững các phép toán so sánh và các toán tử logic cơ bản trong Python.

Hiểu rõ cách hoạt động và bảng giá trị chân lý của `and`, `or`, `not`.

Biết cách kết hợp phép so sánh và phép toán logic để xây dựng biểu thức điều kiện.

Nội dung bài học:

1. Các phép toán so sánh (Comparison Operators)

Phép toán so sánh dùng để so sánh hai giá trị với nhau.

Kết quả của phép so sánh luôn trả về một giá trị kiểu Logic (`True` hoặc `False`).

Bảng các phép toán so sánh:

So sánh bằng `==`: Trả về `True` nếu hai giá trị bằng nhau.

So sánh khác `!=`: Trả về `True` nếu hai giá trị khác nhau.

So sánh lớn hơn `>`: Trả về `True` nếu giá trị bên trái lớn hơn giá trị bên phải.

So sánh nhỏ hơn `<`: Trả về `True` nếu giá trị bên trái nhỏ hơn giá trị bên phải.

So sánh lớn hơn hoặc bằng `>=`: Trả về `True` nếu giá trị bên trái lớn hơn hoặc bằng giá trị bên phải.

So sánh nhỏ hơn hoặc bằng `<=`: Trả về `True` nếu giá trị bên trái nhỏ hơn hoặc bằng giá trị bên phải.

Ví dụ minh họa phép so sánh:

```
a = 10
```

```
b = 5
```

```
print(a > b)
```

```
print(a == b)
```

```
print(a != b)
```

2. Các phép toán logic (Logical Operators)

Phép toán logic dùng để kết hợp nhiều biểu thức so sánh hoặc đảo ngược giá trị logic. Python hỗ trợ 3 toán tử logic chính: `and`, `or`, `not`.

Phép toán `and` (VÀ):

Trả về `True` khi và chỉ khi tất cả các điều kiện thành phần đều `True`.

Nếu có ít nhất một điều kiện `False`, kết quả sẽ là `False`.

Ví dụ: `(5 > 3) and (10 > 2)` cho kết quả `True`.

Phép toán `or` (HOẶC):

Trả về `True` nếu có ít nhất một điều kiện thành phần là `True`.

Chỉ trả về `False` khi tất cả các điều kiện đều `False`.

Ví dụ: `(5 > 3) or (10 < 2)` cho kết quả `True`.

Phép toán `not` (PHỦ ĐỊNH):

Dùng để đảo ngược giá trị logic của biểu thức.

Nếu biểu thức là `True` thì `not` sẽ chuyển thành `False`, và ngược lại.

Ví dụ: `not (5 > 3)` cho kết quả `False`.

Ví dụ minh họa phép toán logic:

```
x = 15
```

```
kiem_tra_and = (x > 10) and (x < 20)
```

```
kiem_tra_or = (x < 10) or (x == 15)
```

```
kiem_tra_not = not (x > 10)
```

```
print(kiem_tra_and)
```

```
print(kiem_tra_or)
```

```
print(kiem_tra_not)
```

3. Thứ tự ưu tiên thực hiện

Khi biểu thức kết hợp cả phép toán số học, so sánh và logic, Python ưu tiên thực hiện theo thứ tự:

Thứ nhất: Các phép toán số học (+, -, *, /, ...).

Thứ hai: Các phép toán so sánh (==, !=, >, <, >=, <=).

Thứ ba: Phép toán logic `not`.

Thứ tư: Phép toán logic `and`.

Thứ năm: Phép toán logic `or`.

Bài tập thực hành:

Khai báo biến tuổi `tuoi = 18` và điểm trung bình `dtb = 7.5`.

Viết biểu thức logic kiểm tra xem học sinh có đủ điều kiện nhận học bổng không (Điều kiện: `tuoi >= 18` và `dtb >= 8.0`).

Viết chương trình kiểm tra xem một số `n` có nằm trong khoảng từ 1 đến 100 hay không bằng toán tử `and`.