

# Theory: Comparisons

🕒 19 minutes    11 / 12 problems solved

Start practicing

17400 users solved this topic. Latest completion was 9 minutes ago.

Writing code without comparing any values in it will get you only so far. Now, it's time to master this skill.

## §1. Comparison operators

Comparison or relation operations let you compare two values and determine the relation between them. There are ten comparison operators in Python:

- `<` strictly less than
- `<=` less than or equal
- `>` strictly greater than
- `>=` greater than or equal
- `==` equal
- `!=` not equal
- `is` object identity
- `is not` negated object identity
- `in` membership
- `not in` negated membership.

The result of applying these operators is always `bool`. The following sections focus on the first six operators, but you can find more details about identity and membership testing in the next topics.

## §2. Comparing integers

In this topic, we will cover only integer comparison.

```
1 a = 5
2 b = -10
3 c = 15
4
5 result_1 = a < b    # False
6 result_2 = a == a   # True
7 result_3 = a != b   # True
8 result_4 = b >= c   # False
```

Any expression that returns integer is a valid comparison operand too:

```
1 calculated_result = a == b + c # True
```

Given the defined variables `a`, `b` and `c`, we basically check if `5` is equal to `-10 + 15`, which is true.

## §3. Comparison chaining

Since comparison operations return boolean values, you can join them using logical operators.

```
1 x = -5
2 y = 10
3 z = 12
4
5 result = x < y and y <= z # True
```

In Python, there is a fancier way to write complex comparisons. It is called **chaining**. For example, `x < y <= z` is almost equivalent to the expression you saw in the last example. The difference is that `y` is evaluated only once.

```
1 result = 10 < (100 * 100) <= 10000 # True, the multiplication is evaluated once
```

Current topic:

✓ [Comparisons](#) Stage 1 16★ ...

Topic depends on:

✓ [Program with numbers](#) Stage 1 17★ ...

✓ [Boolean logic](#) Stage 1 16★ ...

Topic is required for:

[Bytes basics](#) ...

✓ [Set](#) ...

✓ [Identity testing](#) ...

✓ [If statement](#) Stage 1 16★ ...

Table of contents:

- [1 Comparisons](#)
- [§1. Comparison operators](#)
- [§2. Comparing integers](#)
- [§3. Comparison chaining](#)
- [Feedback & Comments](#)

Please pay attention to the fact that tools for code quality often recommend *chaining* comparisons instead of *joining* them.

 Report a typo

1281 users liked this theory. 18 didn't like it. What about you?



Start practicing

[Comments \(7\)](#)[Hints \(0\)](#)[Useful links \(0\)](#)[Show discussion](#)