Numbers and Arithmetic

Python for Absolute Beginners · Lesson 5

What you'll learn

- The two kinds of numbers: integers and floats
- The arithmetic operators: add, subtract, multiply, divide
- Whole-number division and remainders
- Powers, and the order Python does operations in
- What happens when you mix text and numbers

Integers and floats

- An integer (called int) is a whole number: 3, 42, −7
- A float is a number with a decimal point: 3.5, 0.25, −1.75
- The name comes from “floating point”, which is just how computers store decimal numbers
- Python tells them apart by the decimal point: 7 is an integer, 7.0 is a float, even though they represent the same amount



The basic operators

- + adds: 4 + 3 = 7
- – subtracts: 4 – 3 = 1
- * multiplies: 4 * 3 = 12
- / divides: 7 / 2 = 3.5



The basic operators in code

```
print(4 + 3)
print(10 - 4)
print(6 * 7)
print(7 / 2)

Output:
7
6
42
3.5
```

Note that 7 / 2 gives 3.5, not 3. Division in Python always produces a float, even when the answer looks whole: 8 / 2 gives 4.0, not 4.

Whole-number division and remainders

- Sometimes you want the whole-number part of a division, or the leftover
- // is floor division: how many whole times one number fits in another
- % is modulo, the remainder operator
- Modulo is surprisingly useful: x % 2 tells you whether a number is even (remainder 0) or odd (remainder 1)

Floor division and remainders in code

```
print(7 // 2)
print(7 % 2)
print(8 % 2)

Output:
3
1
0
```

7 // 2 is 3 (how many whole 2s fit in 7), 7 % 2 is 1 (what is left over), and 8 % 2 is 0 (nothing left over).

Powers, and the order of operations

- ** raises a number to a power
- When several operators appear in one line, Python follows the same order you learned in maths: powers first, then multiplication and division, then addition and subtraction
- Brackets override everything
- When in doubt, add brackets — they make your intention obvious to anyone reading the code, including future you

Powers and brackets in code

```
print(2 ** 3)
print(5 ** 2)
print(2 + 3 * 4)
print((2 + 3) * 4)

Output:
8
25
14
20
```

The third line is 2 + 12 = 14, because 3 * 4 happens first. The fourth is 5 * 4 = 20, because the brackets force the addition first.

Text and numbers do not mix

- This is the most common arithmetic surprise for beginners
- “5” + “3” joins two pieces of text into “53” — that is called concatenation, and you will meet it properly in Lesson 6
- 5 + 3 adds two numbers and gives 8
- Same symbols, different result, because quotes change what a value is
- If your arithmetic ever gives a strange-looking answer, check whether one of your values is secretly text in quotes

Text joins, numbers add

```
print("5" + "3")
print(5 + 3)

Output:
53
8
```

The quoted values are text, so + joins them; the unquoted values are numbers, so + adds them.

Hands-on exercise

- In the shell, work out: 15 + 27, 100 − 38, 9 * 8, 20 / 4 — write down each result and note which ones are floats
- Use // and % to split 17 cookies between 5 friends: how many whole cookies each, and how many left over?
- Predict the result of (10 + 2) * 3 before running it, then check
- Try print("4" * 3) in the shell — what happens, and why do you think it happens?

Key takeaways

- Integers are whole numbers; floats have decimal points
- +, −, *, / work as expected, but / always gives a float
- // gives the whole-number part of a division; % gives the remainder
- ** is power; brackets override the normal order of operations
- Quotes change a value's kind: text joins, numbers add



Take the quiz

Take the Lesson 5 quiz — 5 quick questions to check what you learned.

Next lesson

Strings: Text, Quotes, and Length