# Theory: Basic data types



7 minutes 8 / 9 problems solved

Start practicing

Every *data object* (a variable or a constant) has a type that describes how to keep it in memory, which operations can be applied to this object and how to compute them.

A real-world analogy of types may be biological species or any other abstract attribute shared among specific objects. All the dogs you've seen have type *dog*, but each of them is an individual object. Thinking about a *dog* as a type you can assume some operations available, for example, a dog can bark.

In this topic, we will consider only a few of the simplest data types that are commonly used in programming practice.

## §1. Strings

Whenever you want to work with some kind of textual information in your program, you'll have to work with **strings**. The string type is called `str`. Strings are extremely common and useful in Python. String literals may be delimited using either single or double quotes.

- Examples of strings in double quotes:

```
print("")               # empty string
print("string")         # one word
print("Hello, world!")  # a sentence
```

- Examples of strings in single quotes:

```
print('a')                  # single character
print('1234')               # a sequence of digits
print('Bonjour, le monde!')  # a sentence
```

In a real program, a string can represent an email of a person or an organization.

```
print('hello@hyperskill.org')  # printing an email
```

As you can see, strings are very easy to use!

## §2. Numerical types

Numbers are the most important thing for any programmer. There is hardly any serious program you can write without using numbers, so let's discuss some basic **numerical types**:

- `int` **(signed integers)**. Called integers or ints, they are whole numbers (positive, negative, or zero), having no decimal point;
- `float` **(floating-point numbers)**. Called floats, they represent real numbers and have a decimal point.

You can start working with a number by just printing it out.

```
print(11)    # prints 11
print(11.0)  # prints 11.0
```

Even though `11` and `11.0` are the same number, the former is an integer, and the latter is a float. The simplest way to distinguish them is that floats have a **decimal point** and integers don't. Be attentive!

You can also use negative numbers as well as zeroes:

Current topic:
✓ Basic data types Stage 1 17★ ...

Topic depends on:
✓ Comments Stage 1 17★ ...

Topic is required for:
✓ Integer arithmetic Stage 1 17★ ...
Introduction to Python shell Stage 2 ...
✓ Quotes and multi-line strings Stage 1 12★ ...
✓ Variables Stage 1 17★ ...

```
print(0)      # prints 0
print(-5)     # prints -5
print(-1.03)  # prints -1.03
```

Integer numbers can be used to count things in the real world while floating-point numbers are a good choice for statistical and scientific calculations.

## §3. Printing types

We also have a way to clearly demonstrate types of different objects using the `type()` function which is a part of Python.

```
print(type('hello'))  # <class 'str'>
print(type("world"))  # <class 'str'>

print(type(100))      # <class 'int'>
print(type(-50))      # <class 'int'>

print(type(3.14))     # <class 'float'>
print(type(-0.5))     # <class 'float'>
```

As you can see from the examples above, the `type()` function indicates the data type of a passed value after the word *class*.

## §4. Conclusion

We hope that now you have some intuition about the concept of data types. You should remember the simplest types called `str`, `int` and `float` and how to write their literals. In the following topics, we will learn specific features of each of these types. If you need to know the type of an object, just print it using the `type()` function.