Python literals

Literals - data in itself

Literal

**Is data whose values are determined by the literal itself**

123

Can you guess what value it represents? Of course you can ‒ it's one hundred twenty three.

But what about this:

c

Does it represent any value? Maybe. It can be the symbol of the speed of light, for example. It can also be the constant of integration. Or even the length of a hypotenuse in the sense of a Pythagorean theorem. There are many possibilities.



Through this example, you encounter two different types of literals:

**a string**, which you already know,

**and an integer number**, something completely new.

**these two values are stored in completely different ways ‒   
 the string exists as just a string ‒ a series of letters.**

Integers

**1, 100, 5, 86, 4645, …**

the numbers handled by modern computers are of two types:

integers, **that is, those which are devoid of the fractional part;**

floating-point numbers (or simply floats), **that contain (or are able to contain) the fractional part**.

Octal and hexadecimal numbers

If an integer number is preceded by an **0O or 0o prefix** (zero-o), it will be treated as an octal value.

**This means that the number must contain digits taken from the [0..7] range only.**

**0o123 is an octal number with a (decimal) value equal to 83.**

**3 x 1 = 3, 2 x 8 = 16, 1 x 64 = 6.**

**The second convention allows us to use** hexadecimal numbers. Such numbers should be preceded by the **prefix 0x or 0X** (zero-x).

0x123 is a hexadecimal number with a (decimal) value equal to 291.

**3 x 16^0 = 3, 2 x 16^1 = 32, 1 x 16^2 = 256 --> 3 + 32 + 256 = 291**

Floats

**1.1, 5.3, 0.2, .9, 5., ….**

store the numbers that have (or may have) a fractional part after the decimal point

Ints vs floats

4

4.0

You may think that they are exactly the same, but Python sees them in a completely different way.

**4 is an integer number, whereas 4.0 is a floating-point number**.

Strings

**Strings are used when you need to process text** (like names of all kinds, addresses, novels, etc.), not numbers.

strings need quotes

However, there is a catch. The catch is how to encode a quote inside a string which is already delimited by quotes.

1. **Using \** --> print("I like \"Monty Python\"")  
   2. **using “ and ‘**  --> print('I like "Monty Python"')

Boolean values

They're not as obvious as any of the previous ones, as they're used to represent a very abstract value ‒ **truthfulness**.

**Each time you ask Python if one number is greater than another, the question results in the creation of some specific data ‒ a Boolean value**.

**use of only two distinct values: True and False, denoted as 1 and 0**.

Boolean values

are the two constant objects True and False used to represent truth values (**in numeric contexts 1 is True, while 0 is False**.

SECTION SUMMARY

Literals

notations for representing some fixed values in code. Python has various types of literals - for **example, a literal can be a number** (numeric literals, e.g., 123), **or a string** (string literals, e.g., "I am a literal.").

Octal and hexadecimal

binary system is a system of numbers that employs 2 as the base. Therefore, a binary number is made up of 0s and 1s only, e.g., 1010 is 10 in decimal.

Octal and hexadecimal numeration systems, similarly, employ 8 and 16 as their bases respectively. The hexadecimal system uses the decimal numbers and six extra letters.

Integers

(or simply ints) are **one of the numerical types supported by Python**. They are numbers written without a fractional component, e.g., 256, or -1 (negative integers).

Floating-point numbers

(or simply floats) are **another one of the numerical types supported by Python**. They are numbers that contain (or are able to contain) a fractional component, e.g., 1.27.

To encode an apostrophe or a quote inside a string

you can either

use the escape character, e.g., 'I\'m happy.', or

open and close the string using an opposite set of symbols to the ones you wish to encode, e.g., "I'm happy." to encode an apostrophe, and 'He said "Python", not "typhoon"' to encode a (double) quote.