Python Casting❮ Previous❯

Specify a Variable Type

There may be times when you want to specify a type on to a variable. This can be done with casting. Python is an object-orientated language, and as such it uses classes to define data types, including its primitive types.

Casting in python is therefore done using constructor functions:

* int() - constructs an integer number from an integer literal, a float literal (by rounding down to the previous whole number), or a string literal (providing the string represents a whole number)
* float() - constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)
* str() - constructs a string from a wide variety of data types, including strings, integer literals and float literals

Example

Integers:

x = int(1)   # x will be 1  
y = int(2.8) # y will be 2  
z = int("3") # z will be 3  
Try it Yourself »

Example

Floats:

x = float(1)     # x will be 1.0  
y = float(2.8)   # y will be 2.8  
z = float("3")   # z will be 3.0  
w = float("4.2") # w will be 4.2  
Try it Yourself »

Example

Strings:

x = str("s1") # x will be 's1'  
y = str(2)    # y will be '2'  
z = str(3.0)  # z will be '3.0'

Python Strings

String Literals

String literals in python are surrounded by either single quotation marks, or double quotation marks.

'hello' is the same as "hello".

You can display a string literal with the print() function:

Example

print("Hello")  
print('Hello')Try it Yourself »

Assign String to a Variable

Assigning a string to a variable is done with the variable name followed by an equal sign and the string:

Example

a = "Hello"  
print(a)Try it Yourself »

Multiline Strings

You can assign a multiline string to a variable by using three quotes:

Example

You can use three double quotes:

a = """Lorem ipsum dolor sit amet,  
consectetur adipiscing elit,  
sed do eiusmod tempor incididunt  
ut labore et dolore magna aliqua."""  
print(a)Try it Yourself »

Or three single quotes:

Example

a = '''Lorem ipsum dolor sit amet,  
consectetur adipiscing elit,  
sed do eiusmod tempor incididunt  
ut labore et dolore magna aliqua.'''  
print(a)

Strings are Arrays

Like many other popular programming languages, strings in Python are arrays of bytes representing unicode characters.

However, Python does not have a character data type, a single character is simply a string with a length of 1.

Square brackets can be used to access elements of the string.

Example

Get the character at position 1 (remember that the first character has the position 0):

a = "Hello, World!"  
print(a[1])

Slicing

You can return a range of characters by using the slice syntax.

Specify the start index and the end index, separated by a colon, to return a part of the string.

Example

Get the characters from position 2 to position 5 (not included):

b = "Hello, World!"  
print(b[2:5])

Negative Indexing

Use negative indexes to start the slice from the end of the string:

Example

Get the characters from position 5 to position 1 (not included), starting the count from the end of the string:

b = "Hello, World!"  
print(b[-5:-2])

String Length

To get the length of a string, use the len() function.

Example

The len() function returns the length of a string:

a = "Hello, World!"  
print(len(a))

String Methods

Python has a set of built-in methods that you can use on strings.

Example

The strip() method removes any whitespace from the beginning or the end:

a = " Hello, World! "  
print(a.strip()) # returns "Hello, World!"

Example

The lower() method returns the string in lower case:

a = "Hello, World!"  
print(a.lower())

Example

The upper() method returns the string in upper case:

a = "Hello, World!"  
print(a.upper())

Example

The replace() method replaces a string with another string:

a = "Hello, World!"  
print(a.replace("H", "J"))

Example

The split() method splits the string into substrings if it finds instances of the separator:

a = "Hello, World!"  
print(a.split(",")) # returns ['Hello', ' World!']

Check String

To check if a certain phrase or character is present in a string, we can use the keywords in or not in.

Example

Check if the phrase "ain" is present in the following text:

txt = "The rain in Spain stays mainly in the plain"  
x = "ain" in txt  
print(x)

Example

Check if the phrase "ain" is NOT present in the following text:

txt = "The rain in Spain stays mainly in the plain"  
x = "ain" not in txt  
print(x)

String Concatenation

To concatenate, or combine, two strings you can use the + operator.

Example

Merge variable a with variable b into variable c:

a = "Hello"  
b = "World"  
c = a + b  
print(c)

Example

To add a space between them, add a " ":

a = "Hello"  
b = "World"  
c = a + " " + b  
print(c)

String Format

As we learned in the Python Variables chapter, we cannot combine strings and numbers like this:

Example

age = 36  
txt = "My name is John, I am " + age  
print(txt)

But we can combine strings and numbers by using the format() method!

The format() method takes the passed arguments, formats them, and places them in the string where the placeholders {} are:

Example

Use the format() method to insert numbers into strings:

age = 36  
txt = "My name is John, and I am {}"  
print(txt.format(age))

The format() method takes unlimited number of arguments, and are placed into the respective placeholders:

Example

quantity = 3  
itemno = 567  
price = 49.95  
myorder = "I want {} pieces of item {} for {} dollars."  
print(myorder.format(quantity, itemno, price))

You can use index numbers {0} to be sure the arguments are placed in the correct placeholders:

Example

quantity = 3  
itemno = 567  
price = 49.95  
myorder = "I want to pay {2} dollars for {0} pieces of item {1}."  
print(myorder.format(quantity, itemno, price))

Escape Character

To insert characters that are illegal in a string, use an escape character.

An escape character is a backslash \ followed by the character you want to insert.

An example of an illegal character is a double quote inside a string that is surrounded by double quotes:

Example

You will get an error if you use double quotes inside a string that is surrounded by double quotes:

txt = "We are the so-called "Vikings" from the north."

To fix this problem, use the escape character \":

Example

The escape character allows you to use double quotes when you normally would not be allowed:

txt = "We are the so-called \"Vikings\" from the north."

Python Booleans

**Booleans represent one of two values: True or False.**

**Boolean Values**

**In programming you often need to know if an expression is True or False.**

**You can evaluate any expression in Python, and get one of two answers, True or False.**

**When you compare two values, the expression is evaluated and Python returns the Boolean answer:**

**Example**

**print(10 > 9)**

**print(10 == 9)**

**print(10 < 9)**

**When you run a condition in an if statement, Python returns True or False:**

**Example**

**Print a message based on whether the condition is True or False:**

**a = 200**

**b = 33**

**if b > a:**

**print("b is greater than a")**

**else:**

**print("b is not greater than a")**