note.nkmk.me

Multiple assignment in Python: Assign multiple values or the same value to multiple variables

Posted: 2019-06-24 / Modified: 2019-11-05 / Tags: Python

In Python, use the `=` operator to assign values to variables.

```
a = 100
b = 200

print(a)
# 100

print(b)
# 200
```

source: multi_variables_values.py

You can assign values to multiple variables on one line.

This article describes the following two cases.

- Assign multiple values to multiple variables
- Assign the same value to multiple variables

Assign multiple values to multiple variables

You can assign multiple values to multiple variables by separating variables and values with commas `,`.

```
a, b = 100, 200

print(a)
# 100

print(b)
# 200
```

source: multi_variables_values.py

You can assign to more than three variables. Moreover, it is also possible to assign to different types.

```
a, b, c = 0.1, 100, 'string'

print(a)
# 0.1

print(b)
# 100

print(c)
# string
```

source: multi_variables_values.py

If there is one variable on the left side, it is assigned as a tuple.

```
a = 100, 200

print(a)
print(type(a))
# (100, 200)
# <class 'tuple'>
```

source: multi_variables_values.py

If the number of variables on the left and the number of values on the right do not match, a `ValueError` will occur, but you can assign the rest as a list by appending `*` to the variable name.

```
# a, b = 100, 200, 300
# ValueError: too many values to unpack (expected 2)

# a, b, c = 100, 200
# ValueError: not enough values to unpack (expected 3, got 2)

a, *b = 100, 200, 300

print(a)
print(type(a))
# 100
# <class 'int'>

print(b)
print(type(b))
# [200, 300]
# <class 'list'>

*a, b = 100, 200, 300

print(a)
print(type(a))
# [100, 200]
# <class 'list'>

print(b)
print(type(b))
# 300
# <class 'int'>
```

source: multi_variables_values.py

For more information on `*` and how to assign elements of a tuple and list to multiple variables, see the following articles:

- **Related:** Unpack a tuple / list in Python

It is also possible to swap the values of multiple variables in the same way. See the article below.

- **Related:** Swap values in a list or values of variables in Python

Assign the same value to multiple variables

You can assign the same value to multiple variables by using `=` consecutively.

This is useful, for example, when initializing multiple variables to the same value.

```
a = b = 100

print(a)
# 100

print(b)
# 100
```

source: multi_variables_values.py

It is also possible to assign another value into one after assigning the same value. As described later, care must be taken when assigning mutable objects such as lists or dictionaries.

```
a = 200

print(a)
# 200

print(b)
# 100
```

source: multi_variables_values.py

Even three or more can be written in the same way.

```
a = b = c = 'string'

print(a)
# string

print(b)
# string

print(c)
# string
```

source: multi_variables_values.py

Rather than immutable objects such as `int`, `float` and `str`, mutable objects such as `list` and `dict` be careful when assigning.

If you use `=` consecutively, the same object is assigned to all variables, so if you change the value of element or add a new element, the other will also change.

```
a = b = [0, 1, 2]

print(a is b)
# True

a[0] = 100
print(a)
# [100, 1, 2]

print(b)
# [100, 1, 2]
```

source: multi_variables_values.py

Same as below.

```
b = [0, 1, 2]
a = b

print(a is b)
# True

a[0] = 100
print(a)
# [100, 1, 2]

print(b)
# [100, 1, 2]
```

source: multi_variables_values.py

If you want to process them separately, you need to assign them to each.

> after `c = []; d = []`, `c` and `d` are guaranteed to refer to two different, unique, newly created empty lists. (Note that `c = d = []` assigns the same object to both `c` and `d`.)
>
> *3. Data model — Python 3.8.0 documentation*

```
a = [0, 1, 2]
b = [0, 1, 2]

print(a is b)
# False

a[0] = 100
print(a)
# [100, 1, 2]

print(b)
# [0, 1, 2]
```

source: multi_variables_values.py

Related Categories

- Python

Related Articles

- Add padding to the image with Python, Pillow
- Crop a part of the image with Python, Pillow (trimming)

- Initialize a list with given size and values in Python
- Convert bool (True, False) and other types to each other in Python
- pandas: Assign existing column to the DataFrame index with set_index()
- Conditional expressions in Python
- NumPy: Count the number of elements satisfying the condition
- Check if a number is integer or decimal in Python
- Define and call functions in Python (def, return)
- Extract the file, dir, extension name from a path string in Python
- NumPy: Remove rows / columns with missing value (NaN) in ndarray
- pandas: Reset index of DataFrame, Series with reset_index()
- Alpha blending and masking of images with Python, OpenCV, NumPy
- Create transparent png image with Python, Pillow (putalpha)
- Shuffle a list, string, tuple in Python (random.shuffle, sample)
