Max () function in Python

Python's built-in max () function returns the largest element in an iterable or the largest of the passed parameters

Python's built-in **max () function** returns the largest element in an iterable or the largest of the passed parameters.

If the values ??are strings, they will compare alphabetically.

Max () function syntax in Python

The max () function in Python has two forms:

```
max(iterable, *iterables[,key, default])
```

Or:

```
max(item1, item2, *item[, key])
```

Parameters of max function ()

The max () function works with two types of parameters corresponding to the two syntaxes mentioned above:

1. `max(iterable, *iterables[, key, default])`

1. `iterable` : Required. The tuples, string, set, dictionary or iterator objects that you need to find the largest element in.
2. `*iterables` : Optional. The largest Iterable will be returned.
3. `key` : Optional. Key function, where the iterable goes through. Comparisons are made based on the results returned after passing the key function.

4. `default` : Optional. The default value when iterable is empty.

2. `max(item1, item2, *item[, key])`

1. `item1` , `item2` : Required. Object to compare, can be number, string .
2. `*item` : Optional. Other objects for comparison.

Picture 2 of Max () function in Python

At least two objects are needed to make comparisons with the max () function.

3. `key` : Optional. Key function, where items pass. The comparison is performed on the result returned after passing the key function.

Value returned from max ()

The max function returns different results corresponding to the two types as above:

1. `max(iterable, *iterables[, key, default])`

Case
Key
Default
Return value
Iterable empty Yes or No No Exception Exception **ValueError** Iterable blank Yes Yes Returns Default value An iterable (not empty) No or No Returns the largest number in iterable An iterable (not empty) Yes or No Transmission each element in iterable for the key function, the result returned is the largest element based on the return value from the function key Multiple iterable (not empty) No or No Returns iterable with the largest Multiple iterable (not empty) Yes Yes or Don't pass each iterable to the key function. The result returned is the largest iterable based on the value returned from the key function

2. `max(item1, item2, *item[, key])`

Case
Key
Return value
2 item No Return parameter greater than 2 item Yes Pass each parameter for the key function, the result is returned as a larger element based on the return value from the key function Multiple item No Returns the largest parameter Many items Yes Pass each parameter to the key function, the result returned is the largest element based on the value returned from the key function

Example 1: Find the largest element in the number passed

```
# su dung max(item1, item2, *item) print('So lon nhat la:', max(1, 3, 2, 5, 4)) #
```

Run the program, the result is:

```
 So lon nhat la: 5 So lon nhat la: 10
```

Example 2: Find the number that has the largest total digits using the key function

```
def sumDigit(num): sum = 0 while(num): sum += num % 10 num = int(num / 10) retur
```

The return output is:

```
 Ket qua lon nhat la: 267 Ket qua lon nhat la: 821
```

In this example, the parameters or each element in the iterable parameter are passed in turn to `sumDigit()` to get the result that the number has the largest sum of digits.

Example 3: Find the list of the largest length using the key function

```
num = [15, 300, 2700, 821] num1 = [12, 2] num2 = [34, 567, 78] # su dung max(ite
```

Results returned:

```
 List dai nhat la: [15, 300, 2700, 821]
```

In the above program, iterable num, num1 and num2 passed into the key function are built-in len () functions in Python. The result is iterable and the max function will show iterable with the largest length.

See also: Built-in Python functions