Python Sort List

Summary: in this tutorial, you'll learn how to use the Python List `sort()` method to sort a list.

Introduction to the Python List sort() method

To sort a list (https://www.pythontutorial.net/python-basics/python-list/), you use the `sort()` method:

```
list.sort()
```

The `sort()` method sorts the original list in place. It means that the `sort()` method modifies the order of elements in the list.

By default, the `sort()` method sorts the elements of a list using the less-than operator (`<`). In other words, it places the lower elements before the higher ones.

To sort elements from higher to lower, you pass the `reverse=True` argument to the `sort()` method like this:

```
list.sort(reverse=True)
```

Python List sort() method examples

Let's take some examples of using the `sort()` method.

1) Using the Python List sort() method to sort a list of strings

If a list contains strings, the `sort()` method sorts the string elements alphabetically.

The following example uses the `sort()` method to sort the elements in the `guests` list alphabetically:

```
guests = ['James', 'Mary', 'John', 'Patricia', 'Robert', 'Jennifer']
guests.sort()

print(guests)
```

Output:

```
['James', 'Jennifer', 'John', 'Mary', 'Patricia', 'Robert']
```

And the following example uses the `sort()` method with the `reverse=True` argument to sort the elements in the `guests` list in the reverse alphabetical order:

```
guests = ['James', 'Mary', 'John', 'Patricia', 'Robert', 'Jennifer']
guests.sort(reverse=True)

print(guests)
```

Output:

```
['Robert', 'Patricia', 'Mary', 'John', 'Jennifer', 'James']
```

2) Using the Python List sort() method to sort a list of numbers

If a list contains numbers, the `sort()` method sorts the numbers from smallest to largest.

The following example uses the `sort()` method to sort numbers in the `scores` list from smallest to largest:

```
scores = [5, 7, 4, 6, 9, 8]
scores.sort()

print(scores)
```

Output:

```
[4, 5, 6, 7, 8, 9]
```

To sort numbers from the largest to smallest, you use the `sort(reverse=True)` like this:

```
scores = [5, 7, 4, 6, 9, 8]
scores.sort(reverse=True)

print(scores)
```

Output:

```
[9, 8, 7, 6, 5, 4]
```

3) Using the Python List sort() method to sort a list of tuples

Suppose that you have a list of tuples like this:

```
companies = [('Google', 2019, 134.81),
             ('Apple', 2019, 260.2),
```

```
              ('Facebook', 2019, 70.7)]
```

And you want to sort the companies list by revenue from highest to lowest. To do it:

First, specify a sort key and pass it to the `sort()` method. To define a sort key, you create a function that accepts a tuple and returns the element that you want to sort by:

```
def sort_key(company):
    return company[2]
```

This `sort_key()` function accepts a tuple called `company` and returns the third element.

Note that the company is a tuple e.g., `('Google', 2019, 134.81)` . And the `company[2]` references the revenue like `134.81` in this case.

Second, pass the sort_key function to the `sort()` method:

```
companies.sort(key=sort_key, reverse=True)
```

The `sort()` method will use the value returned by the `sort_key()` function for the comparisons.

Note that you just pass the function name without the parentheses to the `sort()` method:

```
companies = [('Google', 2019, 134.81),
             ('Apple', 2019, 260.2),
             ('Facebook', 2019, 70.7)]


# define a sort key
def sort_key(company):
    return company[2]



# sort the companies by revenue
```

```
companies.sort(key=sort_key, reverse=True)

# show the sorted companies
print(companies)
```

Output:

```
[('Apple', 2019, 260.2), ('Google', 2019, 134.81), ('Facebook', 2019, 70.7)]
```

Using lambda expression

To make it more concise, Python allows you to define a function without a name with the following syntax:

```
lambda arguments: expression
```

A function without a name is called an **anonymous function**. And this syntax is called a **lambda expression** (https://www.pythontutorial.net/python-basics/python-lambda-expressions/) .

Technically, it's equivalent to the following function:

```
def name(arguments):
    return expression
```

The following example uses the lambda expression to sort the companies by revenue from low to high:

```
companies = [('Google', 2019, 134.81),
             ('Apple', 2019, 260.2),
             ('Facebook', 2019, 70.7)]

# sort the companies by revenue
companies.sort(key=lambda company: company[2])
```

```
# show the sorted companies
print(companies)
```

Output:

```
[('Apple', 2019, 260.2), ('Google', 2019, 134.81), ('Facebook', 2019, 70.7)]
```

Summary

- Use the Python List `sort()` method to sort a list in place.
- The `sort()` method sorts the string elements in alphabetical order and sorts the numeric elements from smallest to largest.
- Use the `sort(reverse=True)` to reverse the default sort order.