SORT VS SORTED

In Python

By
Harshit Kumar Singh
harshitsingh097@gmail.com

CODE_U_BABY

WhatsApp Channel

Chart Beast Portal

Telegram Channel

In Python, sorting sequences is straightforward with built-in methods like sort() and sorted().

Both methods are used for sorting but differ in their approach and usage.

sorted() Method

The sorted() function sorts any given sequence (like lists, tuples, or dictionaries) and returns a new sorted list. It does not modify the original sequence.

Syntax:

```
sorted(iterable, key=None, reverse=False)
```

sort() Method

The sort() method is specific to lists and sorts the list in place, modifying the original list. It does not return a new list

Syntax:

```
list.sort(key=None, reverse=False)
```

Key Differences

1. **Return Type**:

   sorted(): Returns a new sorted list.

   sort(): Returns None and sorts the list in place.

2. **Original Sequence**:

   sorted(): Does not modify the original sequence.

   sort(): Modifies the original list.

3. **Usage**:

   sorted(): Can be used with any iterable (lists, tuples, dictionaries).
   sort(): Can only be used with lists.

4. **Flexibility**:

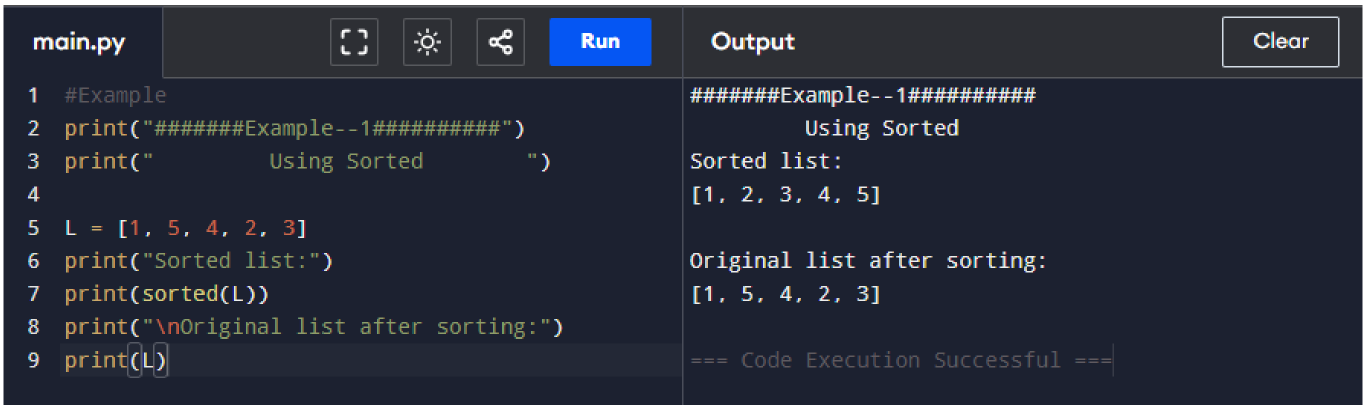
   sorted(): More flexible as it can handle different types of iterables.

   sort(): Limited to lists.

Conclusion

Choosing between sort() and sorted() depends on whether you need to preserve the original sequence or not. Use sorted() when you need a new sorted list and sort() when you want to sort the list in place.

EXAMPLES/ILLUSTRATIONS



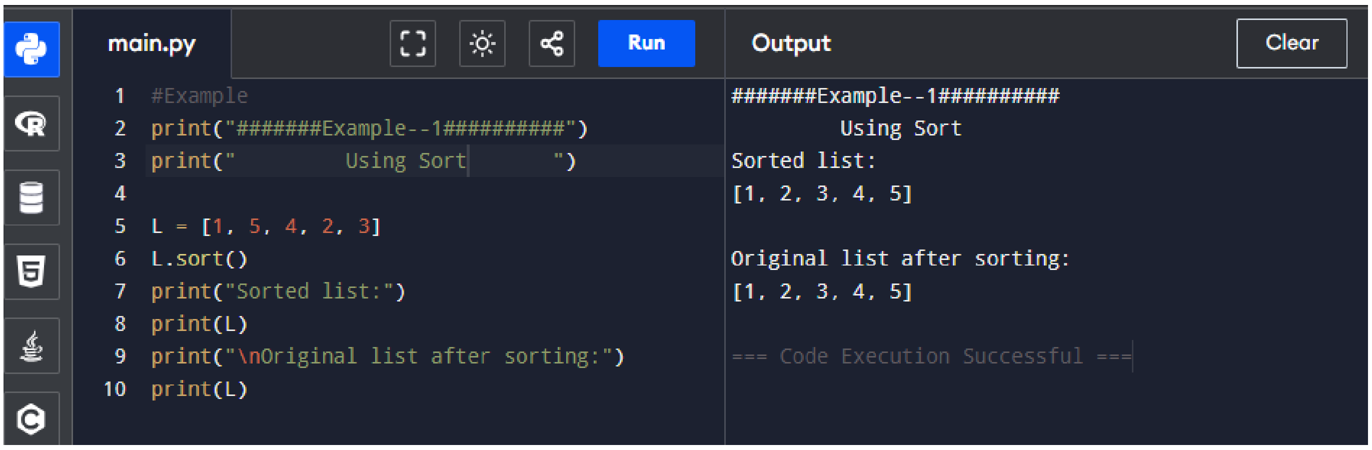
```python
#Example
print("#######Example--1##########")
print("          Using Sorted        ")

L = [1, 5, 4, 2, 3]
print("Sorted list:")
print(sorted(L))
print("\nOriginal list after sorting:")
print(L)
```

```
#######Example--1##########
          Using Sorted
Sorted list:
[1, 2, 3, 4, 5]

Original list after sorting:
[1, 5, 4, 2, 3]

=== Code Execution Successful ===
```



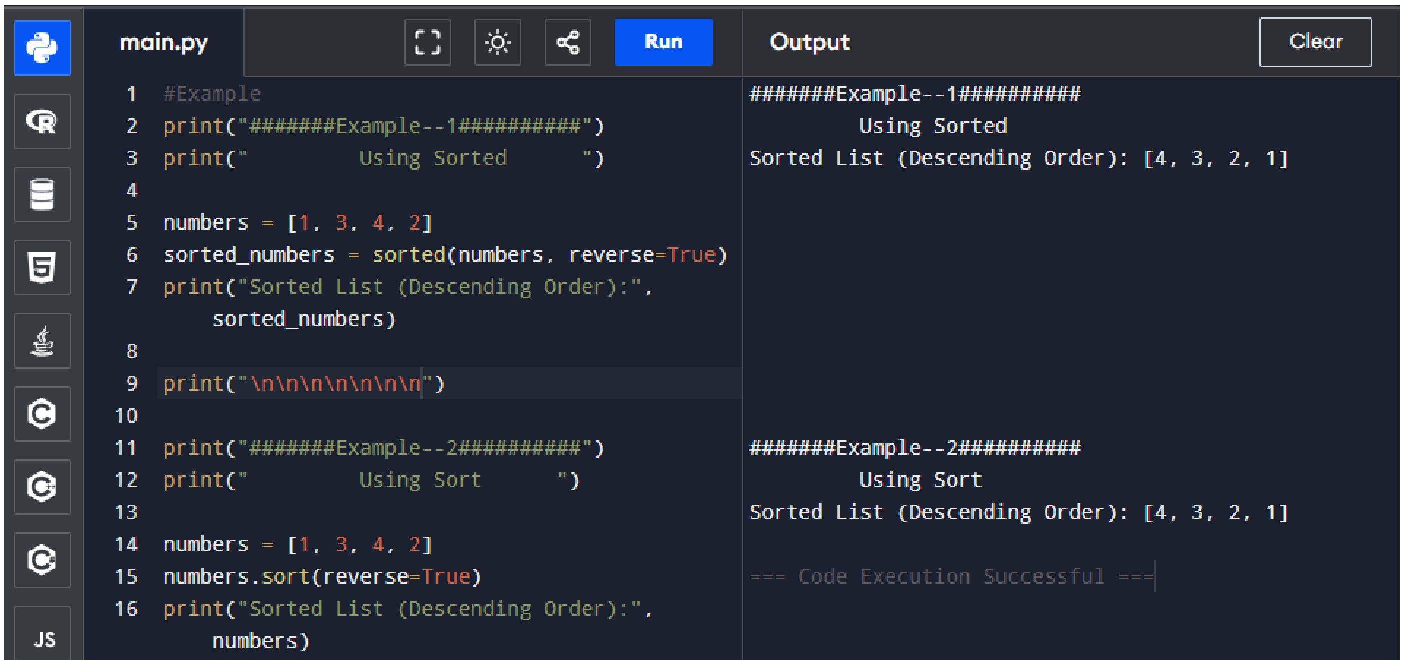
```python
#Example
print("#######Example--1##########")
print("          Using Sort       ")

L = [1, 5, 4, 2, 3]
L.sort()
print("Sorted list:")
print(L)
print("\nOriginal list after sorting:")
print(L)
```

```
#######Example--1##########
          Using Sort
Sorted list:
[1, 2, 3, 4, 5]

Original list after sorting:
[1, 2, 3, 4, 5]

=== Code Execution Successful ===
```



```python
#Example
print("#######Example--1##########")
print("          Using Sorted      ")

numbers = [1, 3, 4, 2]
sorted_numbers = sorted(numbers, reverse=True)
print("Sorted List (Descending Order):",
    sorted_numbers)

print("\n\n\n\n\n\n\n")

print("#######Example--2##########")
print("          Using Sort      ")

numbers = [1, 3, 4, 2]
numbers.sort(reverse=True)
print("Sorted List (Descending Order):",
    numbers)
```

```
#######Example--1##########
          Using Sorted
Sorted List (Descending Order): [4, 3, 2, 1]








#######Example--2##########
          Using Sort
Sorted List (Descending Order): [4, 3, 2, 1]

=== Code Execution Successful ===
```

THANK YOU

Contributed By:
Harshit Kumar Singh
harshitsingh097@gmail.com