Python: Sorting How To

Connect with me: Youtube | LinkedIn | WhatsApp Channel | Web | Facebook | Twitter

- Download PDF
- To access the updated handouts, please click on the following link: https://yasirbhutta.github.io/python/docs/sorting.html

Sorting Basics

list.sort()

Sorts the elements of a list in place, meaning it modifies the original list directly.

Syntax:

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

Parameters:

key (optional): A function that takes a single element from the list and returns a key to be used for sorting. This allows for custom sorting criteria. **reverse (optional):** A boolean value. If True, sorts the list in descending order. Defaults to False (ascending order).

Example:

- Video: List sort() Function

sorted Function

Creates a new sorted list from an iterable object (like a list, tuple, or string), leaving the original iterable unchanged.

Syntax:

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

Parameters:

iterable: The iterable object to be sorted (e.g., a list, tuple, string). **key (optional):** A function that takes a single element from the iterable and returns a key to be used for sorting. This allows for custom sorting criteria. **reverse (optional):** A boolean value. If True, sorts the iterable in descending order. Defaults to False (ascending order).

Examples:

- Video: sorted() Function

Key Functions

Both list.sort() and sorted() have a key parameter to specify a function (or other callable) to be called on each list element prior to making comparisons.

Examples:

- Video: Sorting a List of Colors by Length using Python's sorted() Function
- Video: How to Sort a List of Fruit Tuples
- Video: Sorting a List of Dictionaries by Age using Lambda Function

Operator Module Functions

Ascending and Descending

Both list.sort() and sorted() accept a reverse parameter with a boolean value. This is used to flag descending sorts.

Sort Stability and Complex Sorts

Decorate-Sort-Undecorate

Comparison Functions

True/False (Mark T for True and F for False)

- The list.sort() method modifies the original list. **True or Fasle**
- The sorted() function returns a new sorted list. **True or False**
- The key parameter is used to specify a custom sorting function. **True or False**
- The reverse parameter is used to reverse the sorting order. **True or False**
- The sorted() function can only sort lists. **True or False**

Multiple Choice (Select the best answer)

> Which of the following statements sorts a list in descending order?

1. ☐ list.sort(reverse=True)
2. ☐ sorted(list, reverse=True)
3. ☐ list.reverse()
4. ☐ sorted(list, ascending=False)

> What is the output of the following code?

```
numbers = [3, 1, 4, 2]
numbers.sort(key=lambda x: -x)
print(numbers)
```

1. [ ] [1, 2, 3, 4]
2. [ ] [4, 3, 2, 1]
3. [ ] [1, 4, 2, 3]
4. [ ] [3, 4, 1, 2]

> How would you sort a list of tuples by the second element of each tuple?

1. [ ] list.sort(key=lambda x: x[1])
2. [ ] sorted(list, key=lambda x: x[1])
3. [ ] list.sort(key=operator.itemgetter(1))
4. [ ] All of the above

Fill in the Blanks

1. The ______ method sorts a list in place.
2. The ______ function returns a new sorted list.
3. The ______ parameter is used to specify a custom sorting function.
4. The ______ parameter is used to reverse the sorting order.

Exercises

1. Sorting Numbers:

- Create a list of numbers and sort them in ascending order using both `list.sort()` and `sorted()`.
- Sort a list of numbers in descending order.
- Sort a list of numbers in ascending order, but keep the original list intact.
- Sort a list of numbers based on their absolute values.

2. Sorting Strings:

- Sort a list of strings alphabetically.
- Sort a list of strings in reverse alphabetical order.
- Sort a list of strings by their length. solution
- Sort a list of strings alphabetically, but case-insensitively.

3. Sorting Tuples:

- Sort a list of tuples by their first element.
- Sort a list of tuples by their second element.
- Sort a list of tuples by the length of their second element.

4. Sorting Dictionaries:

- Sort a list of dictionaries by their keys.
- Sort a list of dictionaries by their values.
- Sorting a List of Dictionaries by Age using Lambda Function Solution

5. Custom Sorting:

- Sort a list of objects based on a custom attribute.
- Sort a list of words based on the number of vowels in each word.

- Sort a list of employees based on their salary, then by their name if salaries are equal.

6. Sorting Algorithms:

- Implement the bubble sort algorithm.
- Implement the insertion sort algorithm.
- Implement the selection sort algorithm.
- Implement the merge sort algorithm.
- Implement the quick sort algorithm.

7. Advanced Sorting:

- Sort a large list of numbers efficiently using a suitable algorithm.
- Sort a list of items based on multiple criteria.
- Handle sorting of mixed data types (e.g., numbers, strings, tuples).
- Implement a stable sorting algorithm.

Review Questions

- Explain the difference between list.sort() and sorted().
- What is the purpose of the key parameter in sorting functions?

References and Bibliography

- Sorting HOW TO: Python - Documentation