



Arrays: Sort algorithm stability



Agenda

- To understand the meaning of a stable sort algorithm
- To explore the properties of a stable sort algorithm
- To analyze the stability of:
 - Bubble sort
 - Insertion sort
 - Merge sort
 - Quick sort

Stable sort algorithms

- Stable sorting algorithm
 - Multiple sort criteria: Often the case when objects are stored.
 - Based on the scenario when at least one “key” used for sorting has duplicates.
- Example scenario: eCommerce customers inventory
 - Object attributes:
 - First name
 - Last name
 - IP Address
 - Last login timestamp

Stable sort algorithms

- Example scenario: eCommerce customers inventory
 - First sort by last login timestamp.
 - You get a certain relative order of first names.
 - Save a copy of this sorted list.
 - Next, sort by first name of the users.
 - Compare this final sorted result, with the saved copy.
- Is the relative order of duplicate first names unchanged?

Stable sort: properties

- Property of Stable sort
 - Elements with equal keys, maintain their relative order after a sort.
 - When re-sorted by a different key, this order does not change.
- Where is stable sort useful?
 - When we sort objects with multiple attributes.
 - Sort according to different criteria - select a different attribute each time.

Stable sort: Analysis

- Let us examine the following sort algorithms for stability:
 - Bubble sort
 - Insertion sort
 - Mergesort
 - Quicksort

Is Bubble sort stable?

- A closer look at the pseudocode of Bubblesort:

Code For Bubble Sort

```
def swap(array, i, j):  
    temp = array[i]  
    array[i] = array[j]  
    array[j] = temp
```

```
def bubble_sort(array):  
    size = len(array)  
    for i in range(0, size):  
        for j in range(size-i-1):  
            if array[j] > array[j+1]:  
                swap(array, j, j+1)  
            else:  
                break
```

This file is meant for personal use by mayurbang5@gmail.com only.

Sharing or publishing the contents in part or full is liable for legal action.

Is Bubble sort stable?

- In the algorithm, the swap happens only when $\text{array}[j]$ is strictly greater than $\text{array}[j+1]$.
- Above condition ensures that if $\text{array}[j]$ and $\text{array}[j+1]$ have same value, then there will be no swapping and position of the elements will not change.
- Conclusion: Bubble sort is stable.

Is Insertion sort stable?

- A closer look at the pseudocode of Insertionsort():

Code For Insertion Sort

```
def swap(array, i, j):  
    temp = array[i]  
    array[i] = array[j]  
    array[j] = temp
```

```
def insertion_sort(array):  
    size = len(array)  
    for i in range(0, size):  
        for j in range(i, 0, -1):  
            if array[j] < array[j-1]:  
                swap(array, j, j-1)  
            else:  
                break
```

This file is meant for personal use by mayurbang5@gmail.com only.

Is Insertion sort stable?

- If we have duplicate (equal) elements in the array:
 - Equal elements never move past each other.
 - The relative order is maintained, even after the sort procedure completes.
- Conclusion: Insertion sort is stable.

Is Merge sort stable?

- A closer look at the pseudocode of Merge():

Pseudocode For Merge

```
i = lo
j = mid+1
for k in range(lo, hi+1):
    if i > mid:
        array[k] = aux[j]
        j+=1
    elif j > hi:
        array[k] = aux[i]
        i += 1
    elif aux[j] < aux[i]:
        array[k] = aux[j]
        j += 1
    else:
        array[k] = aux[i]
        i += 1
```

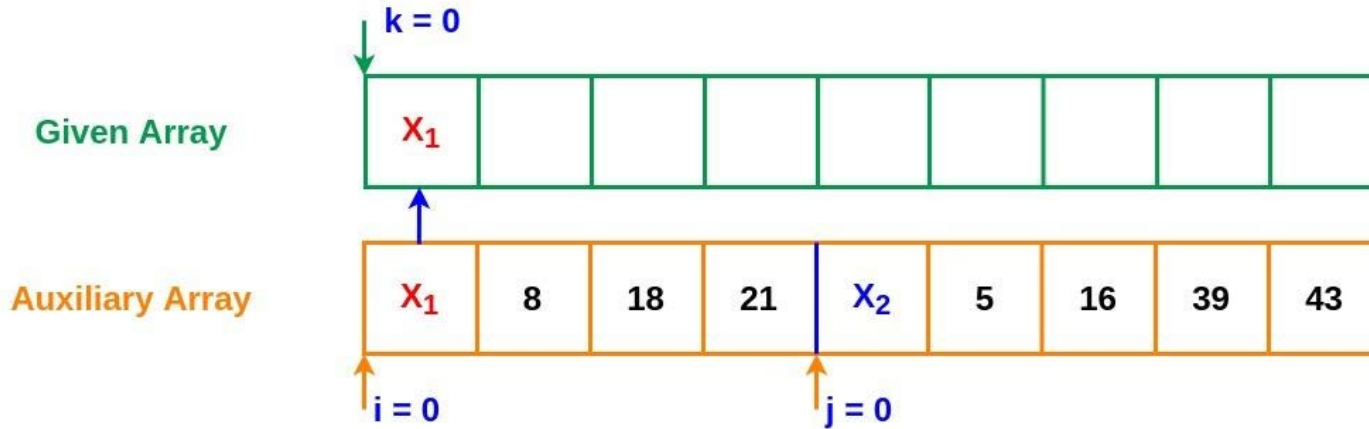
This file is meant for personal use by mayurbang5@gmail.com only.

Sharing or publishing the contents in part or full is liable for legal action.

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Is Merge sort stable?

Merge Procedure: Duplicate Keys



- **Conclusion: Mergesort is stable.**

Is Merge sort stable?

- Merge sort is stable.
 - Used in Java for object-based sort.
 - Used in other languages too:
 - Python
 - C++
 - Perl

Is Quick sort stable?

```
def partition(array, lo, hi):  
    i = lo  
    j = hi+1  
  
    while True:  
        i += 1  
        while array[i] < array[lo]:  
            if i == hi:  
                break  
  
        i += 1  
  
        j -= 1  
        while array[lo] < array[j]:  
            if j == lo:  
                break  
  
        j -= 1  
  
        if i >= j:  
            break  
  
        swap(array, i, j)  
  
    swap(array, lo, j)  
    return j
```

- There is an initial shuffling of Array elements.
- Also, the pseudocode for Partition()

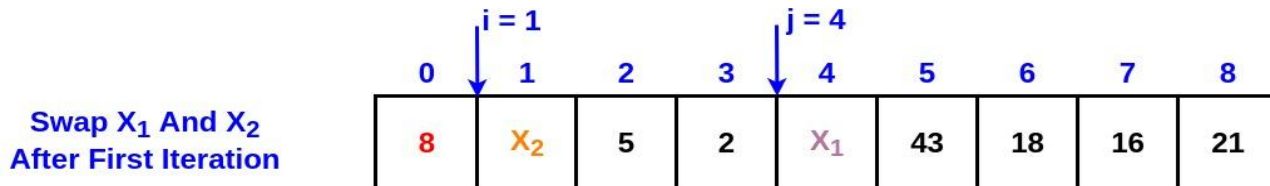
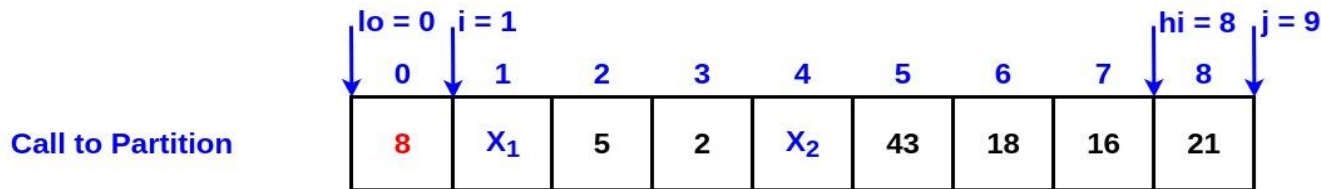
This file is meant for personal use by mayurbang5@gmail.com only.

Sharing or publishing the contents in part or full is liable for legal action.

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Is Quick sort stable?

Quicksort Partitioning: Duplicate values



- Conclusion: Quicksort is NOT stable.

Is Quick sort stable?

- Quick sort is NOT stable.
 - Used in Java for sorting primitive data types.
 - Used in other languages too:
 - Python
 - C qsort() function
 - Unix/Linux

Summary

- We understood the meaning of a stable sort algorithm.
- We explored the properties of a stable sort algorithm.
- We analyzed the following algorithms for stability:
 - Bubble sort
 - Insertion sort
 - Merge sort
 - Quick sort



Thank You