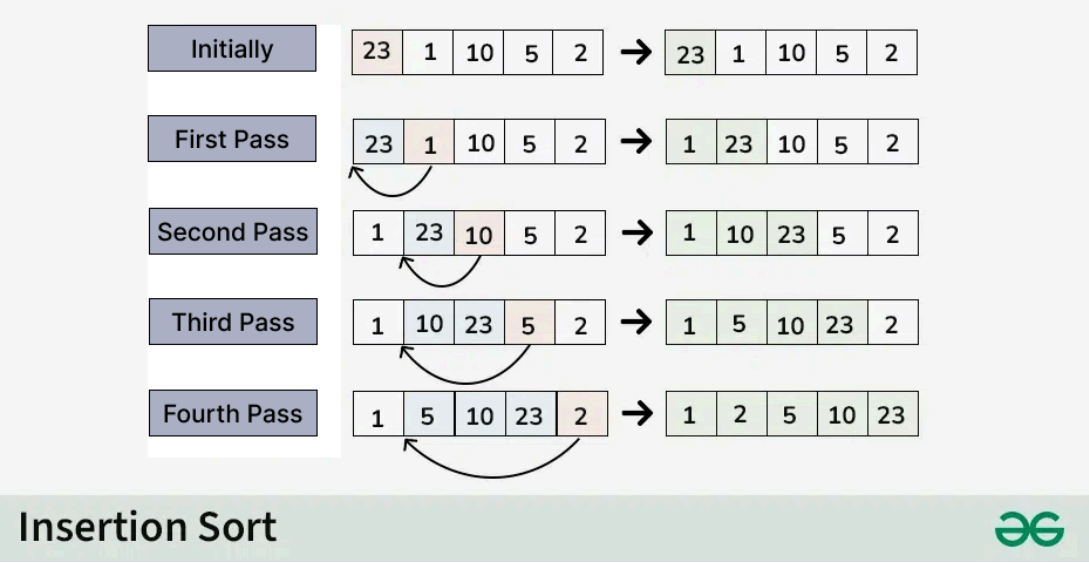
Insertion Sort Algorithm

Last Updated : 07 Oct, 2024

Insertion sort is a simple sorting algorithm that works by iteratively inserting each element of an unsorted list into its correct position in a sorted portion of the list. It is like sorting playing cards in your hands. You split the cards into two groups: the sorted cards and the unsorted cards. Then, you pick a card from the unsorted group and put it in the right place in the sorted group.

- We start with second element of the array as first element in the array is assumed to be sorted.
- Compare second element with the first element and check if the second element is smaller then swap them.
- Move to the third element and compare it with the first two elements and put at its correct position
- Repeat until the entire array is sorted.



C++ C Java Python C# JavaScript PHP

```python
# Python program for implementation of Insertion Sort

# Function to sort array using insertion sort
```

```python
def insertionSort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i - 1

        # Move elements of arr[0..i-1], that are
        # greater than key, to one position ahead
        # of their current position
        while j >= 0 and key < arr[j]:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key

# A utility function to print array of size n
def printArray(arr):
    for i in range(len(arr)):
        print(arr[i], end=" ")
    print()

# Driver method
if __name__ == "__main__":
    arr = [12, 11, 13, 5, 6]
    insertionSort(arr)
    printArray(arr)

    # This code is contributed by Hritik Shah.
```

Output

```
5 6 11 12 13
```

Illustration

arr = {23, 1, 10, 5, 2}

Initial:

- *Current element is* ***23***
- *The first element in the array is assumed to be sorted.*
- *The sorted part until* ***0th*** *index is :* ***[23]***

First Pass:

- *Compare* ***1*** *with* ***23*** *(current element with the sorted part).*
- *Since* ***1*** *is smaller, insert* ***1*** *before* ***23*** *.*
- *The sorted part until* ***1st*** *index is:* ***[1, 23]***

Second Pass:

- *Compare* ***10*** *with* ***1*** *and* ***23*** *(current element with the sorted part).*
- *Since* ***10*** *is greater than* ***1*** *and smaller than* ***23*** *, insert* ***10*** *between* ***1*** *and* ***23*** *.*
- *The sorted part until* ***2nd*** *index is:* ***[1, 10, 23]***

Third Pass:

- *Compare* ***5*** *with* ***1*** *,* ***10*** *, and* ***23*** *(current element with the sorted part).*
- *Since* ***5*** *is greater than* ***1*** *and smaller than* ***10*** *, insert* ***5*** *between* ***1*** *and* ***10***
- *The sorted part until* ***3rd*** *index is* ***: [1, 5, 10, 23]***

Fourth Pass:

- *Compare* ***2*** *with* ***1, 5, 10*** *, and* ***23*** *(current element with the sorted part).*
- *Since* ***2*** *is greater than* ***1*** *and smaller than* ***5*** *insert* ***2*** *between* ***1*** *and* ***5*** *.*
- *The sorted part until* ***4th*** *index is:* ***[1, 2, 5, 10, 23]***

Final Array:

- *The sorted array is:* ***[1, 2, 5, 10, 23]***

Complexity Analysis of Insertion Sort :

Time Complexity of Insertion Sort

- **Best case: O(n)** , If the list is already sorted, where n is the number of elements in the list.
- **Average case: O(n^2)** , If the list is randomly ordered
- **Worst case: O(n^2)** , If the list is in reverse order

Space Complexity of Insertion Sort

- **Auxiliary Space:** O(1), Insertion sort requires **O(1)** additional space, making it a space-efficient sorting algorithm.

Advantages of Insertion Sort:

- Simple and easy to implement.
- **Stable** sorting algorithm.
- Efficient for small lists and nearly sorted lists.
- Space-efficient as it is an in-place algorithm.
- Adoptive. the number of inversions is directly proportional to number of swaps. For example, no swapping happens for a sorted array and it takes O(n) time only.

Disadvantages of Insertion Sort:

- Inefficient for large lists.
- Not as efficient as other sorting algorithms (e.g., merge sort, quick sort) for most cases.

Applications of Insertion Sort:

Insertion sort is commonly used in situations where:

- The list is small or nearly sorted.
- Simplicity and stability are important.
- Used as a subroutine in Bucket Sort
- Can be useful when array is already almost sorted (very few inversions)
- Since Insertion sort is suitable for small sized arrays, it is used in Hybrid Sorting algorithms along with other efficient algorithms like Quick Sort and Merge Sort. When the subarray size becomes small, we switch to insertion

sort in these recursive algorithms. For example IntroSort and TimSort use insertions sort.

Frequently Asked Questions on Insertion Sort

Q1. What are the Boundary Cases of the Insertion Sort algorithm?

Insertion sort takes the maximum time to sort if elements are sorted in reverse order. And it takes minimum time (Order of n) when elements are already sorted.

Q2. What is the Algorithmic Paradigm of the Insertion Sort algorithm?

The Insertion Sort algorithm follows an incremental approach.

Q3. Is Insertion Sort an in-place sorting algorithm?

Yes, insertion sort is an in-place sorting algorithm.

Q4. Is Insertion Sort a stable algorithm?

Yes, insertion sort is a stable sorting algorithm.

Q5. When is the Insertion Sort algorithm used?

Insertion sort is used when number of elements is small. It can also be useful when the input array is almost sorted, and only a few elements are misplaced in a complete big array.

GeeksforGeeks 801

Next Article

C Program For Insertion Sort

Similar Reads

Comparison among Bubble Sort, Selection Sort and Insertion Sort

Bubble Sort, Selection Sort, and Insertion Sort are simple sorting algorithms that are commonly used to sort small datasets or as building blocks for more comple...

15 min read

Insertion sort to sort even and odd positioned elements in different orders

We are given an array. We need to sort the even positioned elements in the ascending order and the odd positioned elements in the descending order. We...

7 min read

Sorting by combining Insertion Sort and Merge Sort algorithms

Insertion sort: The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in th...

2 min read

Merge Sort vs. Insertion Sort

Pre-requisite: Merge Sort, Insertion Sort Merge Sort: is an external algorithm based on divide and conquer strategy. In this sorting: The elements are split int...

14 min read

Count swaps required to sort an array using Insertion Sort

Given an array A[] of size N (1 ≤ N ≤ 105), the task is to calculate the number of swaps required to sort the array using insertion sort algorithm. Examples: Input:...

15 min read

Insertion Sort by Swapping Elements

Insertion Sort is suitable for arrays of small size. It also achieves the best-case complexity of O(n) if the arrays are already sorted. We have discussed both...

11 min read

Insertion Sort Visualization using JavaScript

Insertion sort is a simple sorting algorithm in which values from the unsorted part are picked and placed at the correct position in the sorted part. In order to know...

5 min read

Sorting an Array in Bash using Insertion Sort

Given an array, arr[] of size N, the task is to sort the array in ascending order using Insertion Sort in bash scripting. Examples: Input: arr[] = {9, 7, 2, 5}Output: 2 5 7...

2 min read

Binary Insertion Sort

Binary insertion sort is a sorting algorithm which is similar to the insertion sort, but instead of using linear search to find the location where an element should ...

15+ min read

Time complexity of insertion sort when there are O(n) inversions?

What is an inversion? Given an array arr[], a pair arr[i] and arr[j] forms an inversion if arr[i] j. For example, the array {1, 3, 2, 5} has one inversion (3, 2) and...

1 min read

Article Tags : DSA Sorting Accenture Cisco +5 More

Practice Tags : Accenture Cisco Dell Grofers +4 More

Tower, Sector- 136, Noida, Uttar Pradesh (201305) | Registered Address:- K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305