Insertion Sort

Insertion Sort

Insertion Sort is a simple and intuitive sorting algorithm that builds the final `sorted array one element` at a time.

It works by taking **each element** from the input and inserting it into its correct position in the `already sorted part of the array`.

`Starting from the second element`, it compares the current element with the previous ones, `shifting larger elements` one position ahead to make space for the **current element**.

This `process continues` until all elements are **sorted**.

`Insertion Sort` is efficient for `small` or nearly `sorrted` datasets and operates in-place without requiring **extra memory**.

Approach:

`Start from the second element` (index 1) since the first element is trivially `"sorted"`.

Store the current element (curr) and compare it with all `previous elements`.

`Shift` the previous elements **one position** forward if they are greater than the current element.

`Insert` the **current element** (curr) at its correct sorted position.

`Repeat` until the whole `array is sorted`.

Time & Space Complexity:

Time Complexity: O(n) Best Case `Already Sorted`.

Average Case: $O(n^2)$

Worst Case: $O(n^2)$ **Every element has to be compared and shifted back to the start.**

Space Complexity: O(1) No extra array is used; sorting is done in-place.

Dry Run

Input: `arr = [4, 5, 1, 3, 9]`

```
i = 1 → curr = 5, prev = 0
arr[0] = 4 ≤ 5 → no shifting
Insert curr → [4, 5, 1, 3, 9]

i = 2 → curr = 1, prev = 1
arr[1] = 5 > 1 → shift → [4, 5, 5, 3, 9]
arr[0] = 4 > 1 → shift → [4, 4, 5, 3, 9]
prev = -1 → stop
Insert curr → [1, 4, 5, 3, 9]

i = 3 → curr = 3, prev = 2
arr[2] = 5 > 3 → shift → [1, 4, 5, 5, 9]
arr[1] = 4 > 3 → shift → [1, 4, 4, 5, 9]
arr[0] = 1 ≤ 3 → stop
Insert curr → [1, 3, 4, 5, 9]

i = 4 → curr = 9, prev = 3
arr[3] = 5 ≤ 9 → no shifting
Insert curr → [1, 3, 4, 5, 9]
```

Final Sorted Array: `[1, 3, 4, 5, 9]`

JavaScript | Python | Java | C++ | C | C#

```
let arr = [4,5,1,3,9]
```

```
function insertionSort(arr){
  let n = arr.length;
  for(let i=1; i < n; i++) {
    let curr = arr[i];
    let prev = i - 1;
    while(arr[prev] > curr && prev >= 0) {
      arr[prev + 1] = arr[prev];
      prev--;
    }
    arr[prev + 1] = curr;
  }
  return arr;
}
let result = insertionSort(arr);
console.log("Sorted array", result);
```


Insertion Sort - DSA Notes

Insertion Sort - DSA Notes