SNS COLLEGE OF TECHNOLOGY
Coimbatore-37.
An Autonomous Institution

COURSE NAME : 23CAT602 - DATA STRUCTURES & ALGORITHMS

I YEAR/ I SEMESTER

UNIT – III SORTING & SEARCHING

Topic: Exchange Sort

Ms.B.Sumathi
Assistant Professor
Department of Computer Science and Engineering

EXCHANGE SORT

Exchange sort is an algorithm used to sort in **ascending** as well as **descending** order. It compares the **first** element with every element if any element seems out of order it **swaps**.

EXCHANGE SORT

Steps Involved in Implementation for ascending order sorting:

✓First, we will iterate over the array from **arr[1]** to **n – 2** in the outer loop to compare every single element with every other element in an array, inner loops will take of comparing that single element in the outer loop with all the other elements in an array.

✓The inner loop will start from **i + 1st** index where i is the index of the outer loop

✓We compare if the ith element is bigger than the jth element we swap in case of ascending order

✓To sort in descending order we swap array elements if the jth element is bigger than the ith element

✓If there is no case where the condition doesn't meet that means it is already in desired order so we won't perform any operations

✓Here both if and the inner loop end and so does the outer loop after that we didn't take the last element in the outer loop since the inner loop's current index is i+1th so eventually, when the current index of the outer loop is n-2 it will automatically take care of last element due to i+1th index of the inner loop. this case can also be considered a corner case for this algorithm

EXCHANGE SORT

Example:
Input: arr[] = {5, 1, 4, 2, 8}
Output: {1, 2, 4, 5, 8}
Explanation: Working of exchange sort:
1st Pass:
Exchange sort starts with the very first elements, comparing with other elements to check which one is greater.
(**5 1 4 2 8**) –> (**1 5 4 2 8**).
Here, the algorithm compares the first two elements and swaps since **5 > 1**.
No swap since none of the elements is smaller than 1 so after 1st iteration (**1 5 4 2 8**)
2nd Pass:
(**1 5 4 2 8**) –> (**1 4 5 2 8**), since **4 < 5**
(**1 4 5 2 8**) –> (**1 2 5 4 8**), since **2 < 4**
(**1 2 5 4 8**) No change since in this there is no other element smaller than 2
3rd Pass:
(**1 2 5 4 8**) -> (**1 2 4 5 8**), since **4 < 5**
after completion of the iteration, we found array is sorted
After completing the iteration it will come out of the loop, Therefore array is sorted.

EXCAHNGE SORT

Time Complexity: O(N^2)
Auxiliary Space : O(1)
Advantages of using **Exchange sort** over other sorting methods:
There are some situations where exchange sort may be preferable over other algorithms. For example, exchange sort may be useful when sorting very small arrays or when sorting data that is already mostly sorted. In these cases, the overhead of implementing a more complex algorithm may not be worth the potential performance gains.
Another advantage of the exchange sort is that it is stable, meaning that it preserves the relative order of equal elements. This can be important in some applications, such as when sorting records that contain multiple fields.

THANK YOU