SNS COLLEGE OF TECHNOLOGY
Coimbatore-37.
An Autonomous Institution

COURSE NAME : 23CAT602 - DATA STRUCTURES & ALGORITHMS

I YEAR/ I SEMESTER

UNIT – III SORTING & SEARCHING

Topic: Selection Sort

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

SELECTION SORT

In selection sort, the smallest value among the unsorted elements of the array is selected in every pass and inserted to its appropriate position into the array. It is also the simplest algorithm. It is an in-place comparison sorting algorithm. In this algorithm, the array is divided into two parts, first is sorted part, and another one is the unsorted part. Initially, the sorted part of the array is empty, and unsorted part is the given array. Sorted part is placed at the left, while the unsorted part is placed at the right.

In selection sort, the first smallest element is selected from the unsorted array and placed at the first position. After that second smallest element is selected and placed in the second position. The process continues until the array is entirely sorted.

SELCTION SORT

Selection sort is a simple and efficient sorting algorithm that works by repeatedly selecting the smallest (or largest) element from the unsorted portion of the list and moving it to the sorted portion of the list.

The algorithm repeatedly selects the smallest (or largest) element from the unsorted portion of the list and swaps it with the first element of the unsorted part. This process is repeated for the remaining unsorted portion until the entire list is sorted.

SELCTION SORT

Lets consider the following array as an example: **arr[] = {64, 25, 12, 22, 11}**

First pass:

For the first position in the sorted array, the whole array is traversed from index 0 to 4 sequentially. The first position where **64** is stored presently, after traversing whole array it is clear that **11** is the lowest value.
Thus, replace 64 with 11. After one iteration **11**, which happens to be the least value in the array, tends to appear in the first position of the sorted list.

SELCTION SORT

Second Pass:

For the second position, where 25 is present, again traverse the rest of the array in a sequential manner.
After traversing, we found that **12** is the second lowest value in the array and it should appear at the second place in the array, thus swap these values.

SELCTION SORT

Third Pass:

Now, for third place, where **25** is present again traverse the rest of the array and find the third least value present in the array.

While traversing, **22** came out to be the third least value and it should appear at the third place in the array, thus swap **22** with element present at third position.

SELCTION SORT

Fourth pass:

Similarly, for fourth position traverse the rest of the array and find the fourth least element in the array

As **25** is the 4th lowest value hence, it will place at the fourth position.

SELCTION SORT

Fifth Pass:

At last the largest value present in the array automatically get placed at the last position in the array
The resulted array is the sorted array.

Complexity Analysis of Selection Sort

Time Complexity: The time complexity of Selection Sort is **$O(N^2)$** as there are two nested loops:

One loop to select an element of Array one by one = O(N)

Another loop to compare that element with every other Array element = O(N)

Therefore overall complexity = O(N) * O(N) = O(N*N) = $O(N^2)$

Auxiliary Space: O(1) as the only extra memory used is for temporary variables while swapping two values in Array. The selection sort never makes more than O(N) swaps and can be useful when memory writing is costly.

SELECTION SORT

Advantages of Selection Sort Algorithm

Simple and easy to understand.
Works well with small datasets.

Disadvantages of the Selection Sort Algorithm

Selection sort has a time complexity of O(n^2) in the worst and average case.
Does not work well on large datasets.
Does not preserve the relative order of items with equal keys which means it is not stable.

THANK YOU