Selection Sort

The selection sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) from unsorted part and putting it at the beginning. The algorithm maintains two subarrays in a given array.

1. The subarray which is already sorted.
2. Remaining subarray which is unsorted.

In every iteration of selection sort, the minimum element (considering ascending order) from the unsorted subarray is picked and moved to the sorted subarray.

Following example explains the above steps:

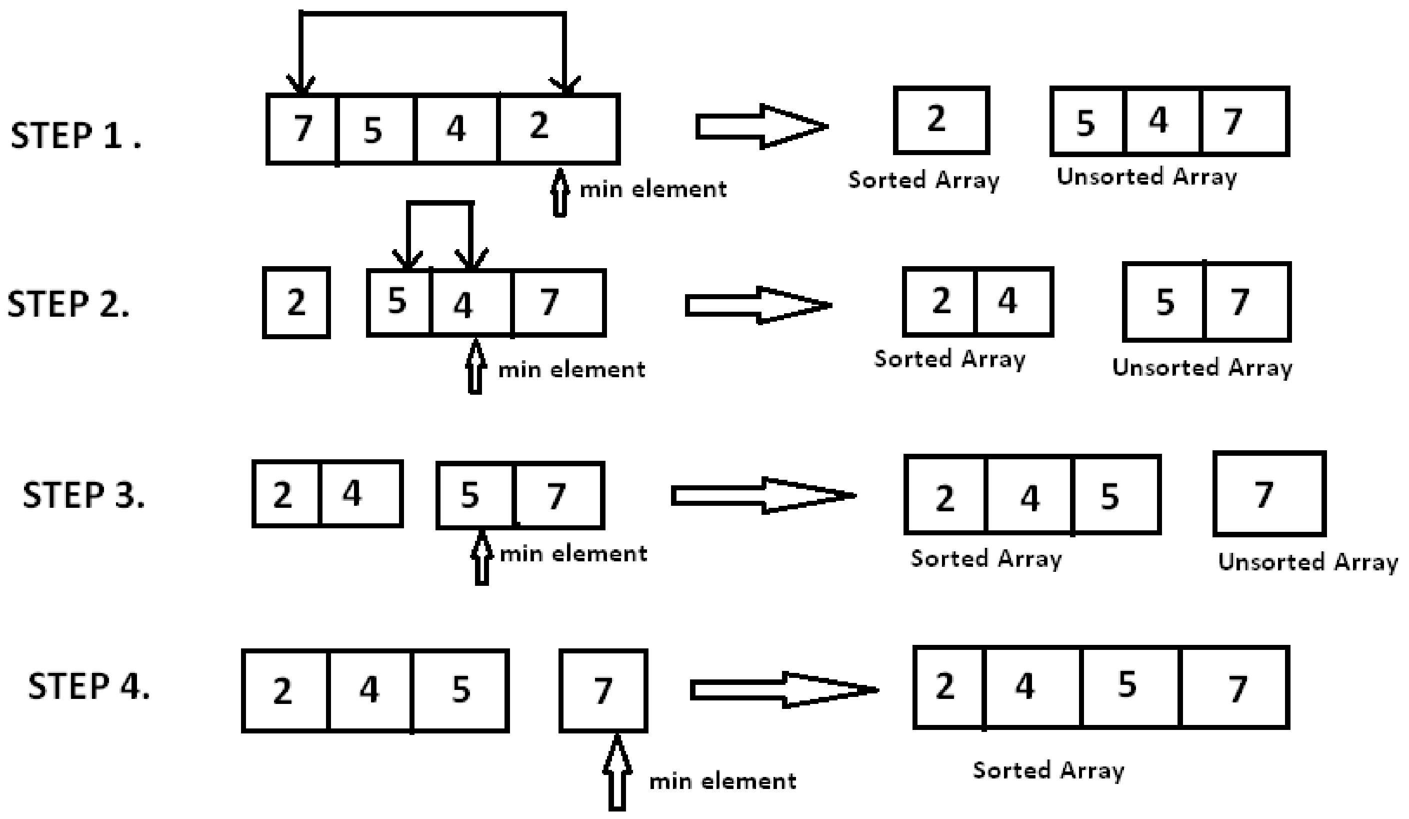
```
arr[] = 64 25 12 22 11

// Find the minimum element in arr[0...4]
// and place it at beginning
*11* 25 12 22 64

// Find the minimum element in arr[1...4]
// and place it at beginning of arr[1...4]
11 *12* 25 22 64

// Find the minimum element in arr[2...4]
// and place it at beginning of arr[2...4]
11 12 *22* 25 64

// Find the minimum element in arr[3...4]
// and place it at beginning of arr[3...4]
11 12 22 *25* 64
```



Time Complexity: O(n^2) as there are two nested loops.

Auxiliary Space: O(1)
The good thing about selection sort is it never makes more than O(n) swaps and can be useful when memory write is a costly operation.

Stability : The default implementation is not stable.
However it can be made stable. Please see stable selection sort for details.