

**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.

## Algorithm

1. SELECTION SORT(arr, n)
- 2.
3. Step 1: Repeat Steps 2 **and** 3 **for** i = 0 to n-1
4. Step 2: CALL SMALLEST(arr, i, n, pos)
5. Step 3: SWAP arr[i] with arr[pos]
6. [END OF LOOP]
7. Step 4: EXIT
- 8.
9. SMALLEST (arr, i, n, pos)
10. Step 1: [INITIALIZE] SET SMALL = arr[i]
11. Step 2: [INITIALIZE] SET pos = i
12. Step 3: Repeat **for** j = i+1 to n
13. **if** (SMALL > arr[j])
14.     SET SMALL = arr[j]
15. SET pos = j
16. [END OF **if**]
17. [END OF LOOP]
18. Step 4: RETURN pos

## Working of Selection sort Algorithm

Now, let's see the working of the Selection sort Algorithm.

To understand the working of the Selection sort algorithm, let's take an unsorted array. It will be easier to understand the Selection sort via an example.

Let the elements of array are -

12	29	25	8	32	17	40
----	----	----	---	----	----	----

Now, for the first position in the sorted array, the entire array is to be scanned sequentially.

At present, **12** is stored at the first position, after searching the entire array, it is found that **8** is the smallest value.

12	29	25	8	32	17	40
----	----	----	---	----	----	----

So, swap 12 with 8. After the first iteration, 8 will appear at the first position in the sorted array.

8	29	25	12	32	17	40
---	----	----	----	----	----	----

For the second position, where 29 is stored presently, we again sequentially scan the rest of the items of unsorted array. After scanning, we find that 12 is the second lowest element in the array that should be appeared at second position.

8	29	25	12	32	17	40
---	----	----	----	----	----	----

Now, swap 29 with 12. After the second iteration, 12 will appear at the second position in the sorted array. So, after two iterations, the two smallest values are placed at the beginning in a sorted way.

8	12	25	29	32	17	40
---	----	----	----	----	----	----

The same process is applied to the rest of the array elements. Now, we are showing a pictorial representation of the entire sorting process.

8	12	25	29	32	17	40
---	----	----	----	----	----	----

8	12	25	29	32	17	40
---	----	----	----	----	----	----

8	12	17	29	32	25	40
---	----	----	----	----	----	----

8	12	17	29	32	25	40
---	----	----	----	----	----	----

8	12	17	29	32	25	40
---	----	----	----	----	----	----

8	12	17	25	32	29	40
---	----	----	----	----	----	----

8	12	17	25	32	29	40
---	----	----	----	----	----	----

8	12	17	25	32	29	40
---	----	----	----	----	----	----

8	12	17	25	29	32	40
---	----	----	----	----	----	----

8	12	17	25	29	32	40
---	----	----	----	----	----	----