Dash
All
Articles
Videos
Problems
Quiz
« Prev
Next »

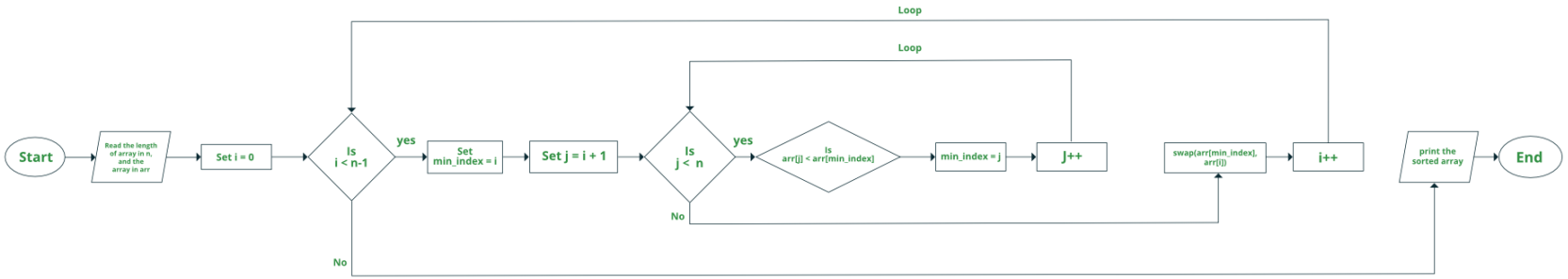
Selection Sort

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.

- The subarray which is already sorted.
- 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.

Flowchart of the Selection Sort:



Flowchart for Selection Sort

How selection sort works?

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.

64	25	12	22	11

- 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.

11	25	12	22	64

Second Pass:

- For the second position, where 25 is present, again traverse the rest of the array in a sequential manner.

11	**25**	12	22	64

- 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.

11	**12**	25	22	64

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.

11	12	**25**	22	64

- 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.

11	12	**22**	25	64

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.

11	12	22	**25**	64

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.

11 12 22 25 64

Approach:

- Initialize minimum value(min_idx) to location 0
- Traverse the array to find the minimum element in the array
- Repeat until array is sorted

Below is the implementation of the above approach:

C++ Java

```java
// Java program for implementation of Selection Sort
class SelectionSort
{
    void sort(int arr[])
    {
        int n = arr.length;

        // One by one move boundary of unsorted subarray
        for (int i = 0; i < n-1; i++)
        {
            // Find the minimum element in unsorted array
            int min_idx = i;
            for (int j = i+1; j < n; j++)
                if (arr[j] < arr[min_idx])
                    min_idx = j;

            // Swap the found minimum element with the first
            // element
            int temp = arr[min_idx];
            arr[min_idx] = arr[i];
            arr[i] = temp;
        }
    }

    // Prints the array
    void printArray(int arr[])
    {
```

```
        int n = arr.length;
        for (int i=0; i<n; ++i)
            System.out.print(arr[i]+" ");
        System.out.println();
    }

    // Driver code to test above
    public static void main(String args[])
    {
        SelectionSort ob = new SelectionSort();
        int arr[] = {64,25,12,22,11};
        ob.sort(arr);
        System.out.println("Sorted array");
        ob.printArray(arr);
    }
}
/* This code is contributed by Rajat Mishra*/
```

Dash
All
Articles
Videos
Problems
Quiz

Output

```
Sorted array:
11 12 22 25 64
```

Report An Issue

If you are facing any issue on this page. Please let us know.