Selection Sort Algorithm

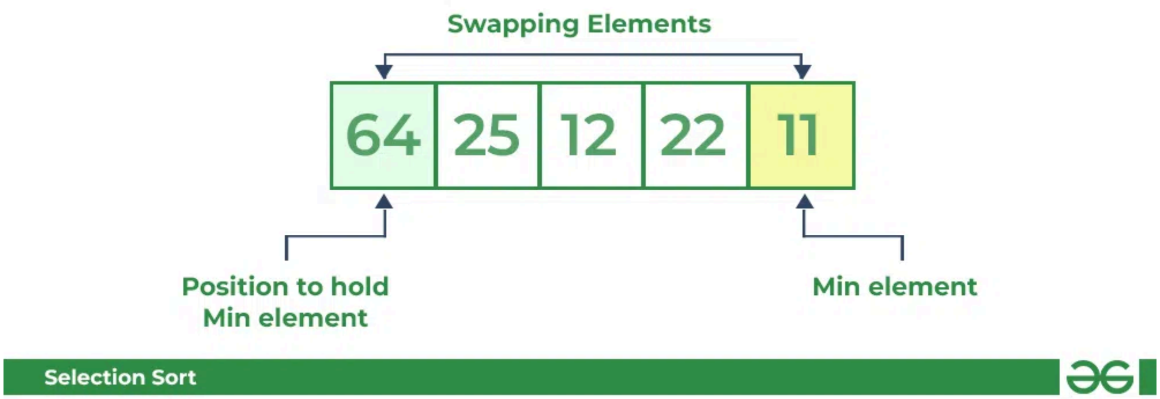
Last Updated : 06 Aug, 2024

> ***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.

How does Selection Sort Algorithm work?

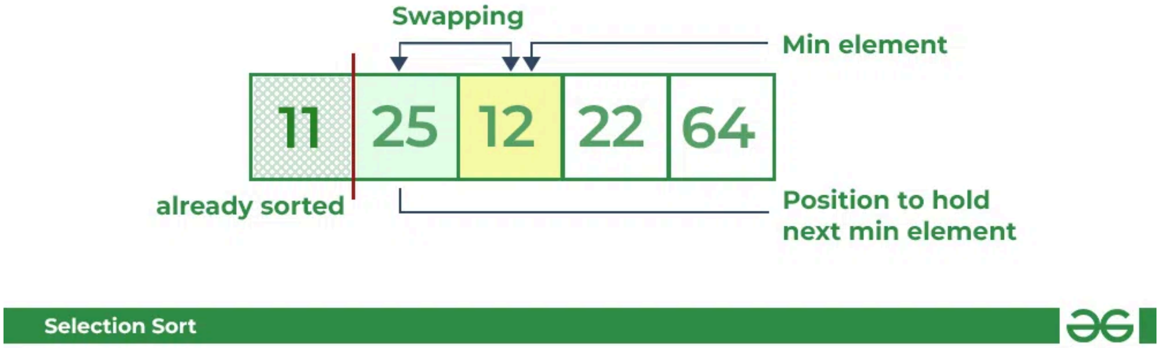
> *Lets consider the following array as an example:* ***arr[] = {64, 25, 12, 22, 11}***
>
> ***First pass:***
>
> - *For the first position in the sorted array, the whole array is traversed from index 0 to 4 sequentially. The first position where* ***64*** *is stored presently, after traversing whole 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.*



Selection Sort Algorithm | Swapping 1st element with the minimum in array

Second Pass:

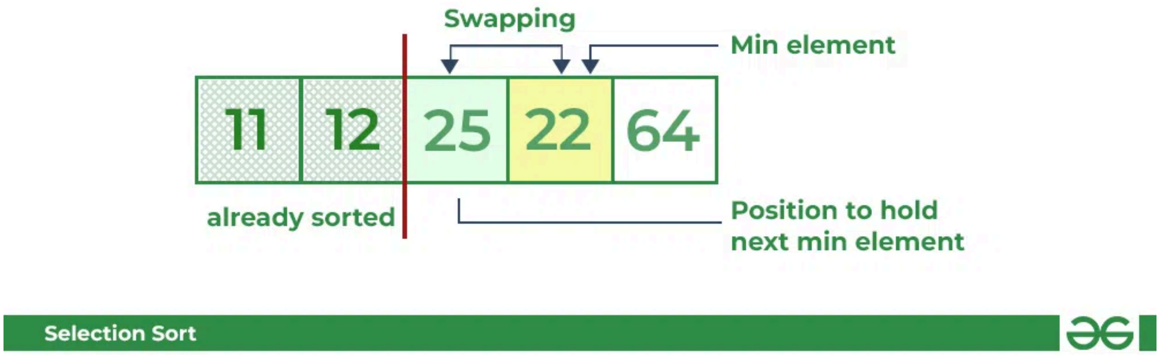
- *For the second position, where 25 is present, again traverse the rest of the array in a sequential manner.*
- *After traversing, we found that **12** is the second lowest value in the array and it should appear at the second place in the array, thus swap these values.*



Selection Sort Algorithm | swapping i=1 with the next minimum element

Third Pass:

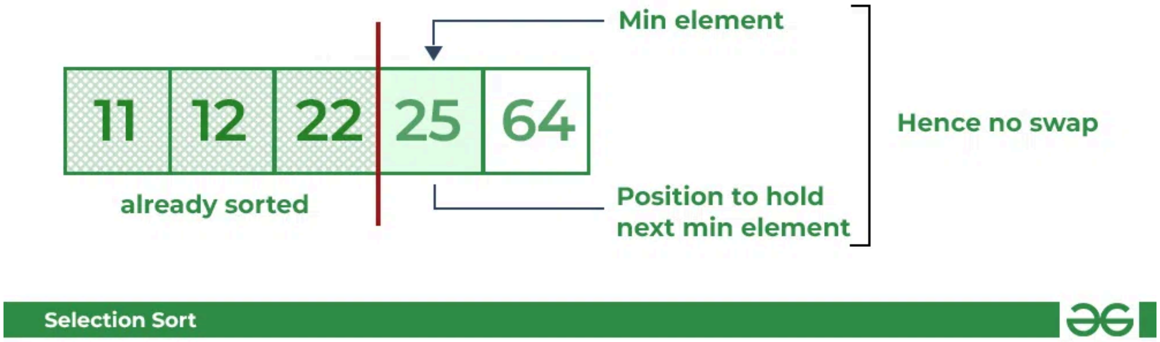
- *Now, for third place, where **25** is present again traverse the rest of the array and find the third least value present in the array.*
- *While traversing, **22** came out to be the third least value and it should appear at the third place in the array, thus swap **22** with element present at third position.*



Selection Sort Algorithm | swapping i=2 with the next minimum element

Fourth pass:

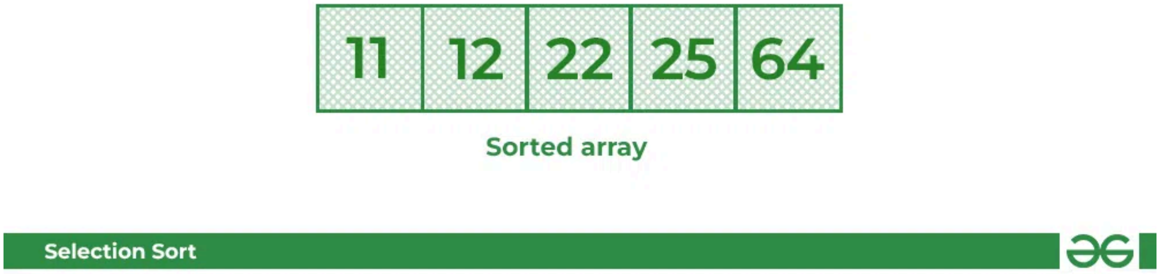
- *Similarly, for fourth position traverse the rest of the array and find the fourth least element in the array*
- *As **25** is the 4th lowest value hence, it will place at the fourth position.*



Selection Sort Algorithm | swapping i=3 with the next minimum element

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.*



Selection Sort Algorithm | Required sorted array

Below is the implementation of the above approach:

C++ C Java Python C# JavaScript PHP

```
// C# program for implementation
// of Selection Sort
using System;

class GFG
{
    static 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
    static void printArray(int []arr)
```

```
            Console.Write(arr[i]+" ");
        Console.WriteLine();
    }

    // Driver code
    public static void Main()
    {
        int []arr = {64,25,12,22,11};
        sort(arr);
        Console.WriteLine("Sorted array");
        printArray(arr);
    }

}
// This code is contributed by Sam007
```

Output

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

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.

Advantages of Selection Sort Algorithm

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

Disadvantages of the Selection Sort Algorithm

- Does not work well on large datasets.
- Does not preserve the relative order of items with equal keys which means it is not stable.

Applications of Selection Sort Algorithm

- Mainly works as a basis for some more efficient algorithms like Heap Sort. Heap Sort mainly uses Heap Data Structure along with the Selection Sort idea.
- Used when memory writes (or swaps) are costly for example EEPROM or Flash Memory. When compared to other popular sorting algorithms, it takes relatively less memory writes (or less swaps) for sorting. But Selection sort is not optimal in terms of memory writes, cycle sort even requires lesser memory writes than selection sort.
- Simple technique and used to introduce sorting in teaching.
- Used as a benchmark for comparison with other algorithms.

Frequently Asked Questions on Selection Sort

Q1. Is Selection Sort Algorithm stable?

The default implementation of the Selection Sort Algorithm is **not stable**. However, it can be made stable. Please see the stable Selection Sort for details.

Q2. Is Selection Sort Algorithm in-place?

Yes, Selection Sort Algorithm is an in-place algorithm, as it does not require extra space.

"The DSA course helped me a lot in clearing the interview rounds. It was really very helpful in setting a strong foundation for my problem-solving skills. Really a great investment, the passion Sandeep sir has towards DSA/teaching is what made the huge difference." - **Gaurav | Placed at Amazon**

Before you move on to the world of development, **master the fundamentals of**

GeeksforGeeks 1.02k

Next Article

Recursive Selection Sort

Similar Reads

Comparison among Bubble Sort, Selection Sort and Insertion Sort

Bubble Sort, Selection Sort, and Insertion Sort are simple sorting algorithms that are commonly used to sort small datasets or as building blocks for more comple...

15 min read

A sorting algorithm that slightly improves on selection sort

As we know, selection sort algorithm takes the minimum on every pass on the array, and place it at its correct position.The idea is to take also the maximum on...

6 min read

Selection Sort VS Bubble Sort

Not a valid contributionIn this, we will cover the comparison between Selection Sort VS Bubble Sort. The resources required by Selection Sort &amp; Bubble So...

13 min read

Program to sort an array of strings using Selection Sort

Given an array of strings, sort the array using Selection Sort. Examples: Input : paper true soap floppy flower Output : floppy, flower, paper, soap, true...

7 min read

Selection sort is a simple sorting algorithm that works by repeatedly finding the minimum element from the unsorted portion of the list and swapping it with the...

2 min read

8086 program for selection sort

Problem - Write an assembly language program in 8086 microprocessor to sort a given array of n numbers using Selection Sort. Assumptions - The number of...

3 min read

Iterative selection sort for linked list

Given a linked list, the task is to sort the linked list in ascending order by using selection sort.Examples: Input : 1-&gt;4-&gt;2-&gt;2-&gt;3 Output : 1-&gt;2-...

15+ min read

Selection Sort Visualizer in JavaScript

Selection sort is the simplest sorting algorithm that works by repeatedly finding the minimum element (considering ascending order) from the unsorted part and...

5 min read

Javascript Program For Recursive Selection Sort For Singly Linked List -...

Given a singly linked list containing n nodes. The problem is to sort the list using the recursive selection sort technique. The approach should be such that it...

4 min read

Time and Space complexity analysis of Selection Sort

The Selection sort algorithm has a time complexity of O(n^2) and a space complexity of O(1) since it does not require any additional memory space apart...

2 min read

Article Tags : DSA Sorting Medlife

Practice Tags : Medlife Sorting

GeeksforGeeks
Sanchhaya Education Private Limited

Corporate & Communications Address:- A-143, 9th Floor, Sovereign Corporate Tower, Sector- 136, Noida, Uttar Pradesh (201305) | Registered Address:- K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305

Company

About Us
Legal
In Media
Contact Us
Advertise with us
GFG Corporate Solution
Placement Training Program
GeeksforGeeks Community

Languages

Python
Java
C++
PHP
GoLang
SQL
R Language
Android Tutorial
Tutorials Archive

DSA

Data Structures
Algorithms
DSA for Beginners
Basic DSA Problems
DSA Roadmap
Top 100 DSA Interview Problems
DSA Roadmap by Sandeep Jain
All Cheat Sheets

Data Science & ML

Data Science With Python
Data Science For Beginner
Machine Learning
ML Maths
Data Visualisation
Pandas
NumPy
NLP
Deep Learning

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap

Python Tutorial

Python Programming Examples
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview Question
Django

Operating Systems

Computer Network

Database Management System

Software Engineering

Digital Logic Design

Engineering Maths

Software Development

Software Testing

System Design

High Level Design

Low Level Design

UML Diagrams

Interview Guide

Design Patterns

OOAD

System Design Bootcamp

Interview Questions

School Subjects

Mathematics

Physics

Chemistry

Biology

Social Science

English Grammar

Commerce

World GK

Git

Linux

AWS

Docker

Kubernetes

Azure

GCP

DevOps Roadmap

Inteview Preparation

Competitive Programming

Top DS or Algo for CP

Company-Wise Recruitment Process

Company-Wise Preparation

Aptitude Preparation

Puzzles

GeeksforGeeks Videos

DSA

Python

Java

C++

Web Development

Data Science

CS Subjects