Selection Sort

Selection Sort works by repeatedly selecting the smallest (or largest) element from the unsorted part of the array and placing it at the beginning of the sorted part.

It divides the array into two subarrays:

Sorted part: Left side (grows with each pass)

Unsorted part: Right side (shrinks with each pass)

Think of it like picking the shortest player from a line and moving him to the front one by one, until everyone’s arranged by height.

🔹 Algorithmic Steps (in pseudocode / algorithm format)

```
Algorithm: SelectionSort(A, n)
Input: Array A of n elements
Output: Sorted array A in ascending order

1. for i ← 0 to n-2 do
2.     min_index ← i
3.     for j ← i+1 to n-1 do
4.         if A[j] < A[min_index] then
5.             min_index ← j
6.     end for
7.     if min_index ≠ i then
8.         swap A[i] and A[min_index]
9. end for
10. return A
```

Implementation in C++

```cpp
#include <iostream>

using namespace std;


void selectionSort(int arr[], int n) {

    for (int i = 0; i < n - 1; i++) {

        int min_index = i; // assume the first element is the minimum


        // Find the index of the smallest element in the unsorted part

        for (int j = i + 1; j < n; j++) {

            if (arr[j] < arr[min_index]) {

                min_index = j;

            }

        }


        // Swap the found minimum element with the first unsorted element

        if (min_index != i) {

            int temp = arr[i];

            arr[i] = arr[min_index];

            arr[min_index] = temp;

        }

    }

}
```

```
int main() {

    int arr[] = {64, 25, 12, 22, 11};

    int n = sizeof(arr) / sizeof(arr[0]);


    selectionSort(arr, n);


    cout << "Sorted array: ";

    for (int i = 0; i < n; i++)

        cout << arr[i] << " ";

    cout << endl;


    return 0;

}
```

Time Complexity

Case	Comparisons	Swaps	Time
Best	$n^2/2$	n-1	$O(n^2)$
Average	$n^2/2$	n-1	$O(n^2)$
Worst	$n^2/2$	n-1	$O(n^2)$

Space complexity: O(1) — in-place sorting

Stable? ❌ No (it may swap equal elements)

Adaptive? ❌ No (still runs full passes even if sorted)

Summary

Don’t use this; we have better options. Just learn this, since this is one of the basic sorting algorithms