Data Structures and Algorithms
Department of Computer Science
University of Engineering and Technology, Lahore

Topic: Sorting Algorithms (Bubble, Selection, Insertion) **CLO1, CLO2**

Objectives of the Lab

By the end of this lab, students will be able to:

- **Understand the concept of sorting** and why it is important in real life (attendance lists, exam results, library books, contacts in mobile, etc.).
- **Learn how data can be arranged** in ascending and descending orders using simple algorithms.
- **Explain and apply Bubble Sort, Selection Sort, and Insertion Sort** step by step.
- **Trace the execution (dry run)** of each algorithm to see how elements move during sorting.
- **Write and execute C++ programs** to implement Bubble Sort, Selection Sort, and Insertion Sort.
- **Differentiate between integer sorting and string sorting** (numbers vs. words).
- **Extend sorting to real-world problems** such as arranging names, sorting marks, and ordering words of a paragraph alphabetically.
- **Analyze the efficiency (basic performance)** of sorting algorithms in terms of number of comparisons and swaps.
- **Practice problem-solving** through coding exercises related to everyday life scenarios.

Introduction to Sorting

Sorting means **arranging data in particular order** (ascending or descending).

- Ascending → Small to Large (1, 2, 3, … or A, B, C, …)
- Descending → Large to Small (10, 9, 8, … or Z, Y, X, …)

Bubble Sort

1. Concept

Bubble Sort compares **two side-by-side elements** and swaps them if the left one is greater.

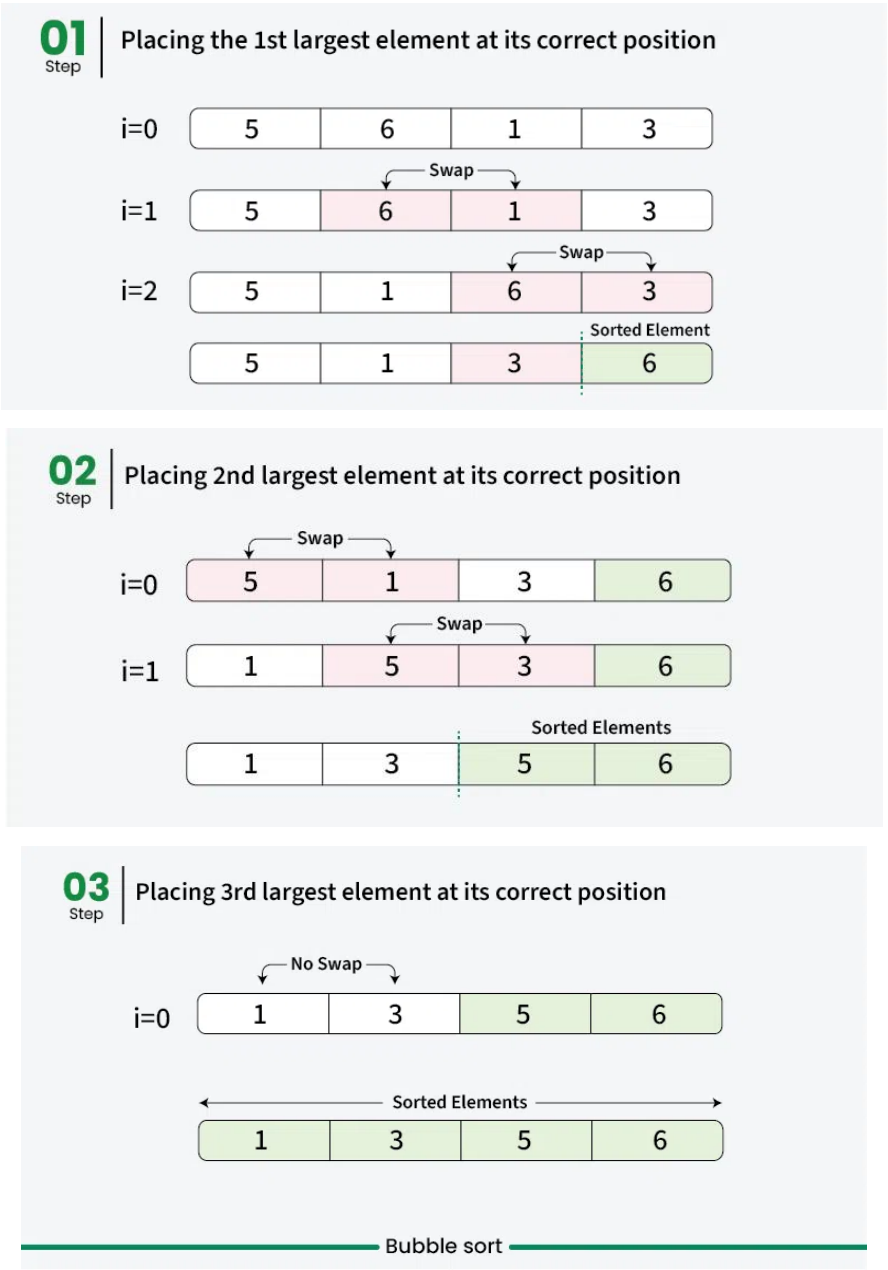
Largest elements move to the right side (like bubbles rising in water).

2. Real-Life Example

When you shake a cold drink bottle, the **big bubbles slowly rise to the top** one by one.

Similarly, Bubble Sort pushes the biggest numbers to the end.

3. Working of bubble Sorting

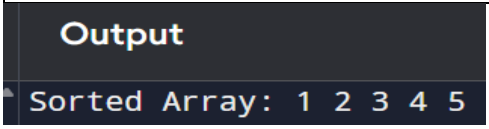


4. Pseudocode

for(int i = 0; i < n-1; i++) {
for(int j = 0; j < n-i-1; j++) {
if(arr[j] > arr[j+1]) {
int temp = arr[j];
arr[j] = arr[j+1];
arr[j+1] = temp;
}
}
	repeat n-1 times:
	for each pair of elements:
	if left > right:
	swap

Question: Write a C++ program to sort an array using Bubble Sort in ascending order. int arr[] = {5, 3, 4, 1, 2}; **Expected Output:** 1 2 3 4 5

```
#include <iostream>
using namespace std;
int main() {
    int arr[] = {5, 3, 4, 1, 2};
    int n = 5;
    for(int i = 0; i < n-1; i++) {
        for(int j = 0; j < n-i-1; j++) {
            if(arr[j] > arr[j+1]) {
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
    cout << "Sorted Array: ";
    for(int i = 0; i < n; i++)
        cout << arr[i] << " ";
}
```



Selection Sort

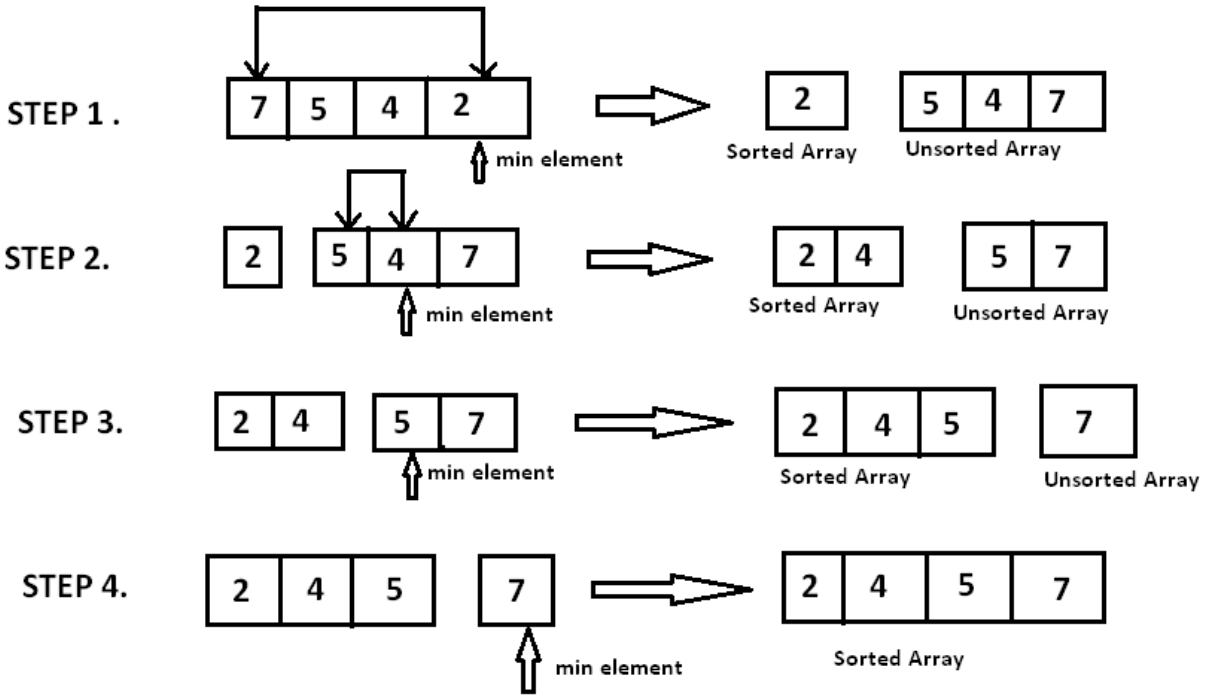
1. Concept

Selection Sort finds the **smallest element** from the unsorted part and places it at the correct position in the sorted part.

2. Real-Life Example

A teacher wants to arrange exam papers in order of roll number. She **selects the smallest roll number** and puts it on top, then the next smallest, and so on.

3. Working of selection Sorting



4. Pseudocode

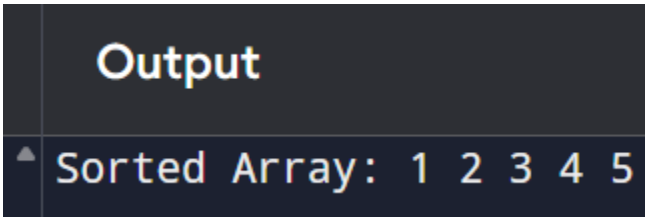
```
for i = 0 to n-1:
  find minimum element from i to n
  swap with element at i
```

Question: Write a C++ program to sort an array using **Selection Sort** in ascending order.

Input: int arr[] = {5, 3, 4, 1, 2}; **Expected Output:** 1 2 3 4 5

```
#include <iostream>
using namespace std;
int main() {
```

```
    int arr[] = {5, 3, 4, 1, 2};
    int n = 5;
    for(int i = 0; i < n-1; i++) {
        int minIndex = i;
        for(int j = i+1; j < n; j++) {
            if(arr[j] < arr[minIndex])
                minIndex = j;
        }
        int temp = arr[i];
        arr[i] = arr[minIndex];
        arr[minIndex] = temp;
    }
    cout << "Sorted Array: ";
    for(int i = 0; i < n; i++)
        cout << arr[i] << " ";
}
```



Insertion Sort

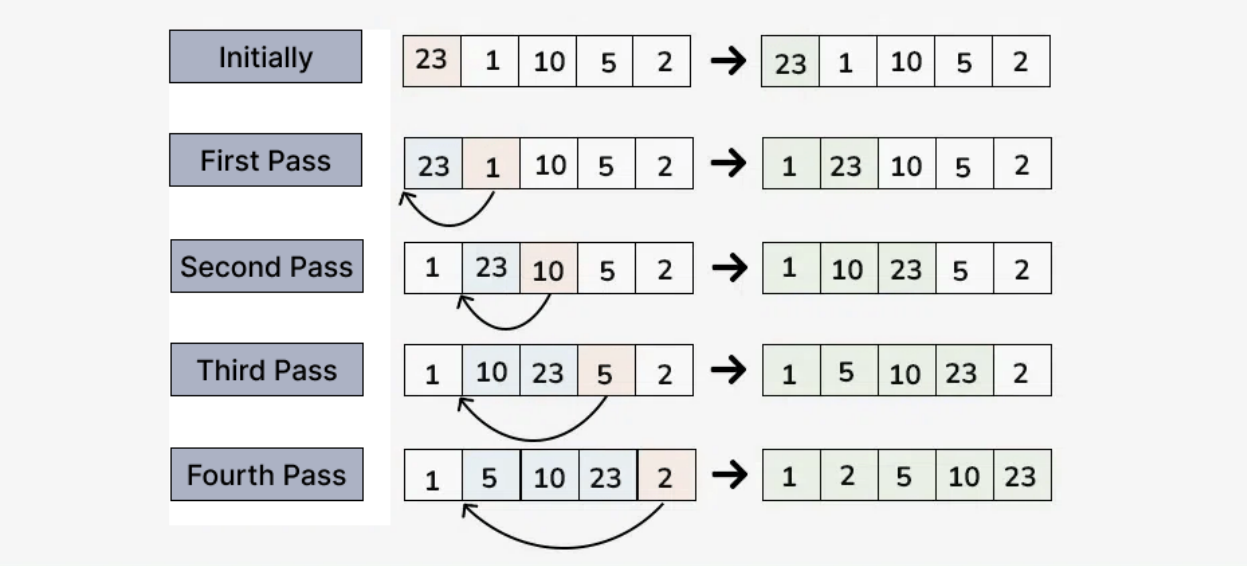
1. Concept

Insertion Sort takes one element at a time and places it in its correct position among already sorted elements.

2. Real-Life Example

When we arrange **playing cards**, we take a card and insert it in the correct position in our hand.

3.Working of Insertion sorting



4. Pseudocode

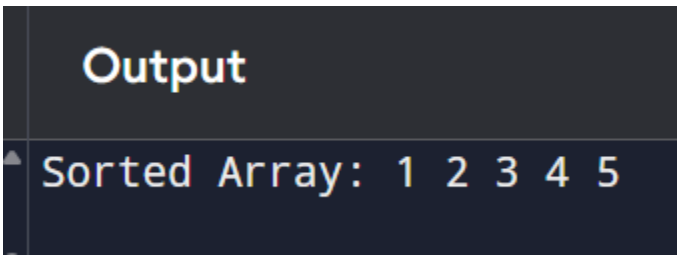
```
for(int i = 1; i < n; i++) {
    int key = arr[i];
    int j = i - 1;
    while(j >= 0 && arr[j] > key) {
        arr[j+1] = arr[j];
        j--;
    }
    arr[j+1] = key;
}
```

```
for i = 1 to n-1:
    key = arr[i]
    shift all bigger elements right
    insert key in correct position
```

Question: Write a C++ program to sort an array using **Insertion Sort** in ascending order. Input: int arr[] = {5, 3, 4, 1, 2}; Expected Output: 1 2 3 4 5

```
#include <iostream>
```

```
using namespace std;
int main() {
    int arr[] = {5, 3, 4, 1, 2};
    int n = 5;
    for(int i = 1; i < n; i++) {
        int key = arr[i];
        int j = i - 1;
        while(j >= 0 && arr[j] > key) {
            arr[j+1] = arr[j];
            j--;
        }
        arr[j+1] = key;
    }
    cout << "Sorted Array: ";
    for(int i = 0; i < n; i++)
        cout << arr[i] << " ";
}
```



Sorting on Strings

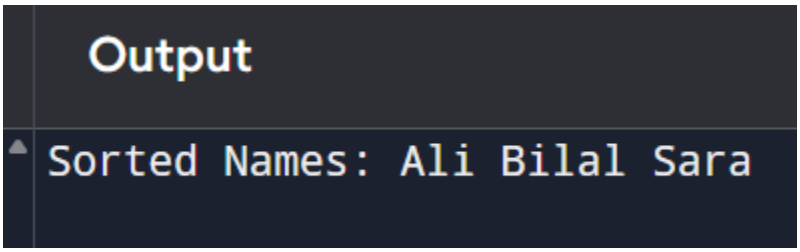
Sorting can also work on words instead of numbers.

- "Ali" < "Bilal" < "Sara"
- Dictionary order (A–Z).
- Computers compare ASCII values, but we usually do **case-insensitive sorting** for names.

Example: Bubble Sort on Strings

```
#include <iostream>
#include <string>
```

```
using namespace std;
int main() {
   string names[] = {"Sara", "Ali", "Bilal"};
   int n = 3;
   for(int i = 0; i < n-1; i++) {
      for(int j = 0; j < n-i-1; j++) {
         if(names[j] > names[j+1]) {
            string temp = names[j];
            names[j] = names[j+1];
            names[j+1] = temp;
         }
      }
   }
   cout << "Sorted Names: ";
   for(int i = 0; i < n; i++)
      cout << names[i] << " ";
}
```



Practice Questions

Q1: Write a C++ program to sort the following array using Bubble Sort:

int arr[5] = {12, 5, 7, 3, 9}; Expected Output: 3 5 7 9 12

Q2: You are given an array with duplicate numbers. Sort the array using Bubble Sort and **remove duplicates**.

int arr[7] = {4, 2, 7, 2, 4, 9, 1}; **Expected Output**: 1 2 4 7 9

Q3: You are given ages of people standing in line. Sort them in **descending order** using Selection Sort.

int ages[8] = {18, 25, 15, 30, 22, 40, 10, 28}; **Expected Output**: 40, 30, 28, 25, 22, 18, 15, 10

Q4: Write a program to sort **employee salaries** and then print the **2nd highest salary**.

int salaries[7] = {35000, 50000, 25000, 40000, 60000, 30000, 45000}; **Expected Output**: 2nd Highest Salary = 50000

Q5: Sort the following array using Insertion Sort and then find the **median**.(the middle number in a sorted list of numbers)

int arr[7] = {12, 4, 7, 9, 2, 15, 10}; Expected Output after sorting: 2 4 7 9 10 12 15 Median = 9

Q6: Sort an array of **student objects** (with name and marks) in **descending order of marks** using any sorting algorithm.

Input:	**Output:**
Ali 85	Sara 92
Sara 92	Hassan 90
Omar 75	Ali 85
Hassan 90	Omar 75

Q7: Sort a paragraph of words in **alphabetical order**.

Input:	**Output:**
"Ali goes to school and Sara goes to market"	Ali and goes goes market Sara school to

Think how we can sort words **grammatically** not **alphabetical order ?**