Course title : **CSE2001**

Course title : **Data Structures and Algorithms**

Module : **4**

Topic : **1**

Introduction to Sorting Algorithms

Objectives

This session will give the knowledge about

- Introduction to Sorting algorithms
- Internal Sorting
- Bubble sort
- Selection sort
- Insertion sort

Introduction to Sorting Algorithms

Sorting refers to arranging data in a particular format.

Sorting algorithm specifies the way to arrange data in a particular order.

We can distinguish two types of sorting.

- If the number of objects is small enough to fits into the main memory, sorting is called internal sorting.
- If the number of objects is so large that some of them reside on external storage during the sort, it is called external sorting.

Types of Sorting algorithms

Some common internal sorting algorithms include:

- Bubble Sort
- Insertion Sort
- Quick Sort
- Heap Sort
- Radix Sort
- Selection sort

External Sorting

- Merge Sort

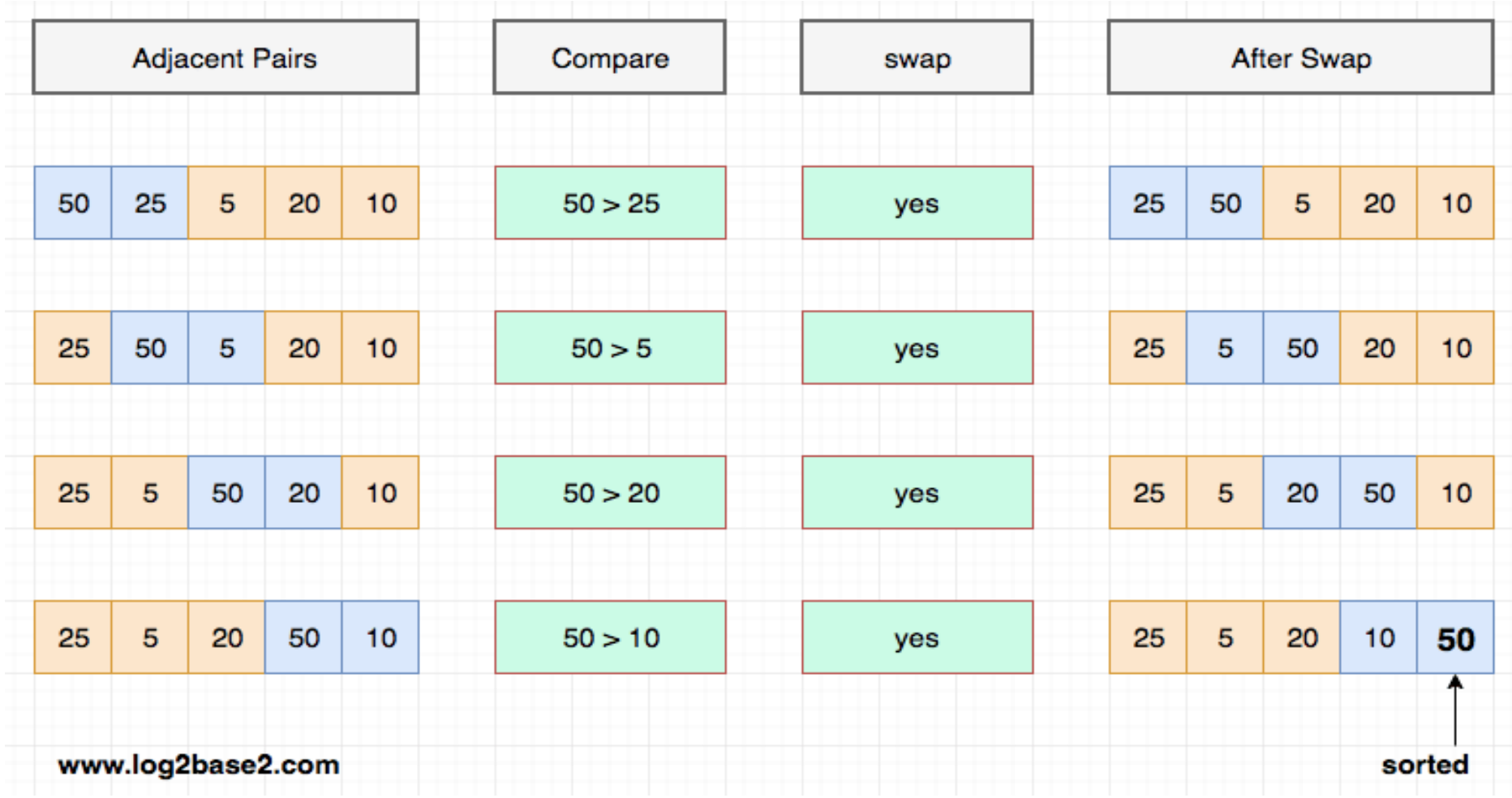
Bubble Sort

Bubble sort is a sorting algorithm that compares two adjacent elements and swaps them until they are in the intjed order.

The bubble sort works as follows:

- Starting from the first index, compare the first and the second elements.
- If the first element is greater than the second element, they are swapped.
- Now, compare the second and the third elements. Swap them if they are not in order.
- The above process goes on until the last element

Bubble Sort


Adjacent Pairs
Compare
swap
After Swap
50 25 5 20 10
50 > 25
yes
25 50 5 20 10
25 50 5 20 10
50 > 5
yes
25 5 50 20 10
25 5 50 20 10
50 > 20
yes
25 5 20 50 10
25 5 20 50 10
50 > 10
yes
25 5 20 10 50
sorted
www.log2base2.com

Bubble Sort

```
static void bubbleSort1(int a[]) {
    int n = a.length;
    boolean swapped = true;
    for (int i = 0; i < n - 1; ++i) {
        swapped = false;
        for (int j = 0; j < n - i - 1; ++j) {
            if (a[j] > a[j + 1]) {
                int t = a[j];
                a[j] = a[j + 1];
                a[j + 1] = t;
                swapped = true;
            }
        }
        if (swapped == false)
            break;
    }
}
```

Bubble Sort

```
static void bubbleSort(int a[]) {
    int n = a.length;
    for(int i=0;i<n;i++) {
        for(int j=i+1;j<n;j++) {
            if(a[i]>a[j]) {
                int t=a[i];
                a[i]=a[j];
                a[j]=t;
            }
        }
    }
}
```

Selection Sort

The algorithm works by selecting the smallest unsorted item and then swapping it with the item in the next position to be filled.

The selection sort works as follows:

- Begin from left most element.
- Find smallest element, swap it with the first element of the array.
- Then find the next smallest element in the remaining array (an array without the first element) and swap it with the second element.

Selection Sort

29, 64, 73, 34, **20**,

20, **64**, 73, 34, **29**,

20, 29, **73**, **34**, 64

20, 29, 34, **73**, **64**

20, 29, 34, 64, 73

Selection Sort

```
static void selectionsort(int a[]) {
        int n = a.length;
        for (int i = 0; i < n; i++) {
                int min = i;
                for (int j = i + 1; j < n; j++) {
                        if (a[min] > a[j])
                        min = j;
                }
                int t = a[i];
                a[i] = a[min];
                a[min] = t;
        }
}
```

Insertion Sort

To sort unordered list of elements, we remove its entries one at a time and then insert each of them into a sorted part (initially empty).

The insertion sort works as follows:

- Begin from left most element.
- We take an element from unsorted part and compare it with elements in sorted part, moving from right to left.

Insertion Sort

We color a sorted part in green, and an unsorted part in black.

29, 20, 73, 34, 64

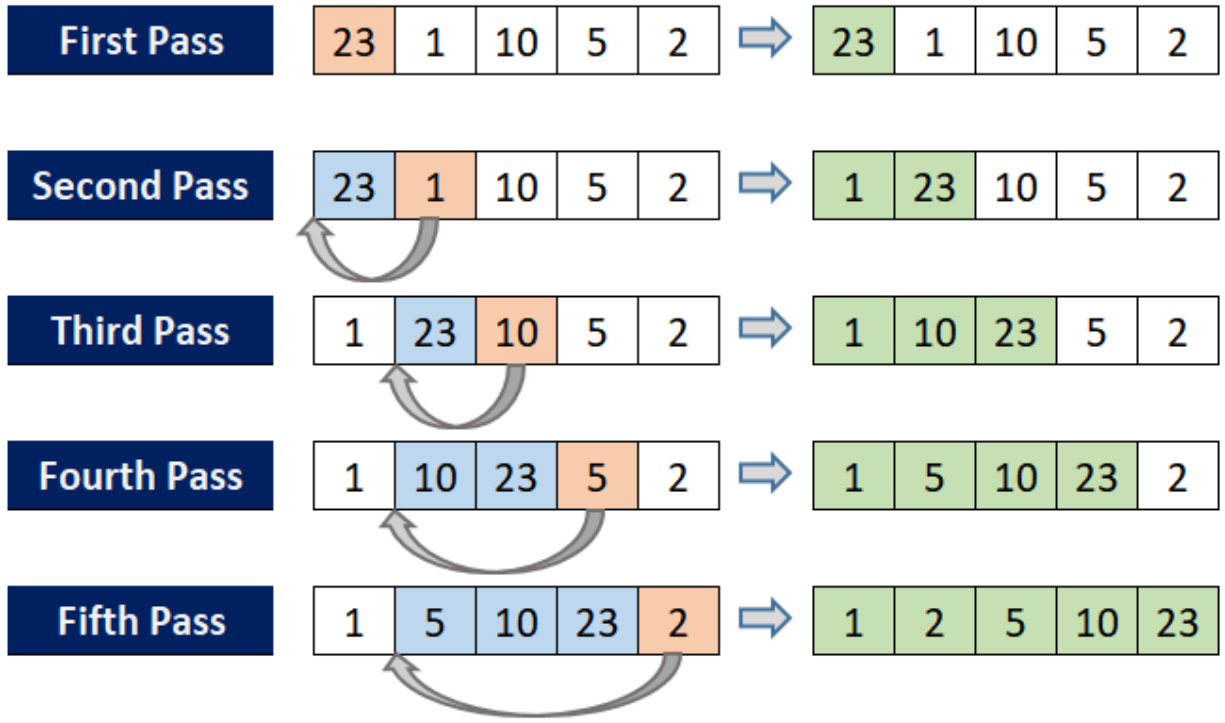
29, 20, 73, 34, 64

20, 29, 73, 34, 64

20, 29, 73, 34, 64

20, 29, 34, 73, 64

20, 29, 34, 64, 73



Insertion Sort

```
static void insertionsort(int a[]) {
    int n = a.length;
    for (int i = 1; i < n; i++) {
        j = i - 1;
        x = a[i];
        while(j >= 0 &&  a[j]>x)
        {
        a[j + 1] = a[j];
        j= j-1;
        }
        a[j + 1] = x;
```

Complexity Analysis

Algorithm	Time Complexity		
	Best	Average	Worst
Selection Sort	Ω(n^2)	θ(n^2)	O(n^2)
Bubble Sort	Ω(n)	θ(n^2)	O(n^2)
Insertion Sort	Ω(n)	θ(n^2)	O(n^2)
Heap Sort	Ω(n log(n))	θ(n log(n))	O(n log(n))
Quick Sort	Ω(n log(n))	θ(n log(n))	O(n^2)
Merge Sort	Ω(n log(n))	θ(n log(n))	O(n log(n))
Bucket Sort	Ω(n+k)	θ(n+k)	O(n^2)
Radix Sort	Ω(nk)	θ(nk)	O(nk)

Summary

At the j of this session we learned about

- Introduction to Sorting algorithms
- Internal Sorting
- Bubble sort
- Selection sort
- Insertion sort