8 days:

**Day 4**

**Sorting and Searching Algorithms**

**Sorting:** Sorting is a process of arranging element or data of in type in acceding or descending order.

In JavaScript Array refence or linear reference data type hold present sort() method which help to do sorting using asci or Unicode code.

First we will see pre defined sort method present in Array with number or string or object type sort.

Then we will see few pre defined sorting technique

Bubble sort

Selection sort

Insertion sort

Heap sort

Merge sort

Timsort etc

Sort pre defined method by default sort one 1 digit number.

If we want to sort for more than 1 digit number then we need to pass callback function inside sort method.

num1.sort((a,b)=>a-b)

if result <0 🡪 a comes before b

if result >0🡪 b comes before a

if result ==0 🡪 a and b unchanged.

Using sort method

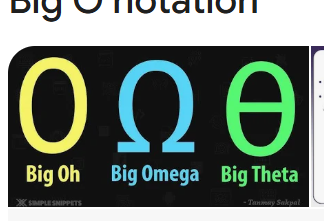
1. Number sort
2. String sort
3. Object sort with property

If we want to use sorting algorithms

1. Time complexity
2. Extra space
3. Data set

Time complexity

Time complexity (may be best, worst and average) based upon type of sort algorithms we use.



**Best case** (Omega notation): the minimum time the algorithms takes when the input is favourable

**Average case** (Theta notation): The expected run time for random number.

**Worst case** (Big O notation): The maximum time the algorithms takes when then input is unfavourable.

1. Bubble sort: this sort repeatedly compares adjacent element and swap if they are in wrong position. It bubbles the largest/smallest value to the top base upon order of the sort which we want to achieve.

Bubble sort : O(n2)

Best case : already sorted 🡪only one pass needed.

1,2,3,4,5

Average case : random input 🡪 multiple pass required depending upon the value.

5,3,1,2,4

Worst case : reversed sorted 🡪 maximum swaps required. If element already in sorted in ascending order to make descending order is worst case.

5,4,3,2,1

1. Selector sort : Selector sort is another simplest sorting technique. Which use comparison- based sorting algorithm.

It works by dividing input list into two parts.

A sorted sub-list and unsorted sub-list. Initially the sorted sub-list is empty. While then unsorted sub-list contains all elements of the input list or data. We find minimum or maximum element iterate through unsorted list or data. Swap with first unsorted elements. Then expand sorted sub-list.

Best case : n(n-1)/2 : if array already sorted. Selection sort still scan or check entire array to find the minimum for each position. 0(n2)

1,2,3,4,5

Average case : - 0(n2)

4,1,3,2,5

Worst case : - 0(n2)

5,4,3,2,1