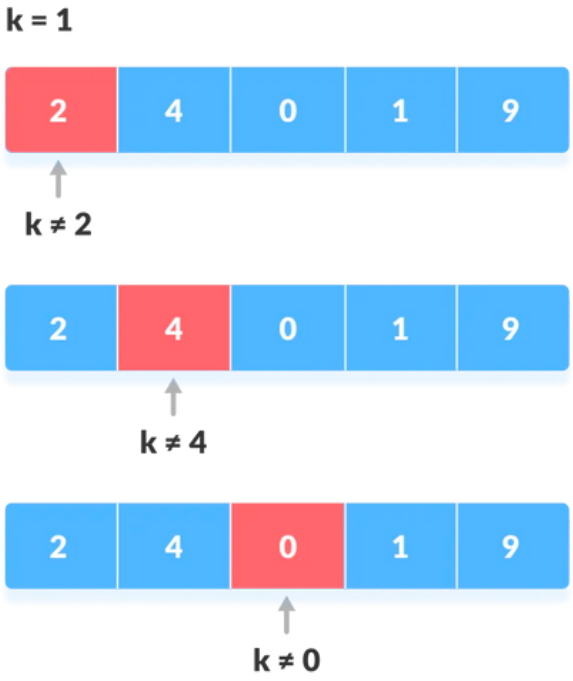
Data structures an Algorithms

linear search

- A linear search or sequential search is a method for finding an element within a list. It sequentially checks each element of the list until a match is found or the whole list has been searched.



Complexity Analysis of Linear Search:

- Time complexity for linear search is denoted by O(n) as every element in the array is compared only once. In linear search, best-case complexity is O(1) where the element is found at the first index. Worst-case complexity is O(n) where the element is found at the last index or element is not present in the array.
    -

Data structures an Algorithms

Advantages of Linear Search:

- Linear search is simple to implement and easy to understand.
- Linear search can be used irrespective of whether the array is sorted or not. It can be used on arrays of any data type.
- Does not require any additional memory.
- It is a well suited algorithm for small datasets.

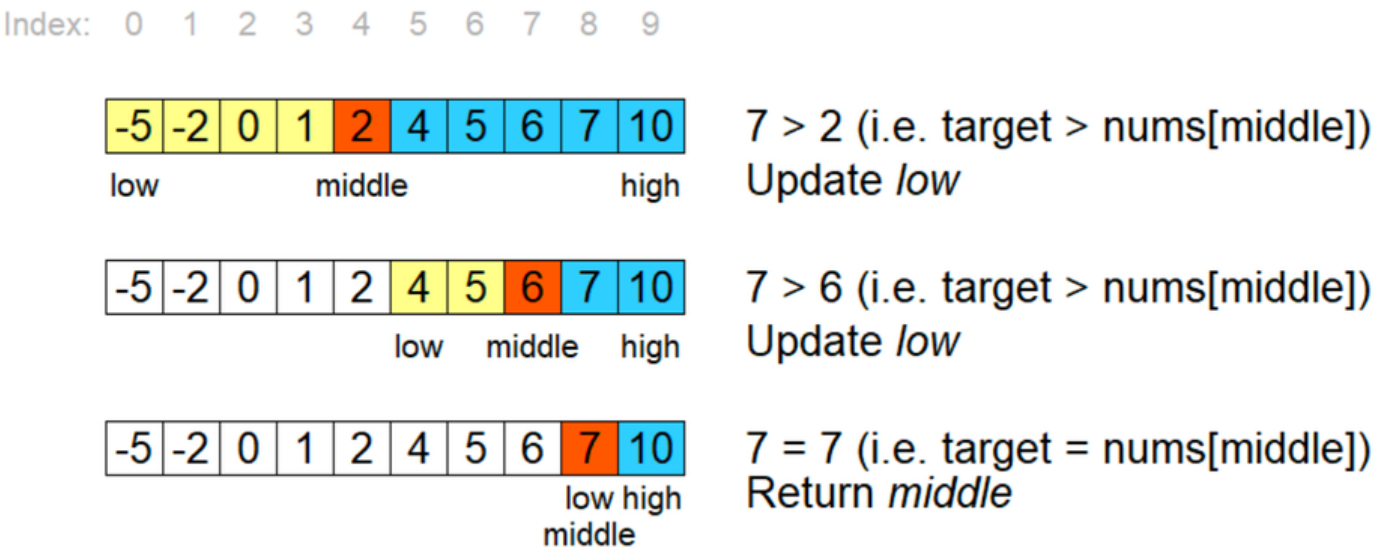
Drawbacks of Linear Search:

- Linear search has a time complexity of O(n), which in turn makes it slow for large datasets.
- Not suitable for large arrays.
- Linear search can be less efficient than other algorithms

Data structures an Algorithms

binary search

- binary search, also known as half-interval search, logarithmic search, or binary chop, is a search algorithm that finds the position of a target value within a sorted array. Binary search compares the target value to the middle element of the array.



Complexity Analysis of Linear Search:

The time complexity of the binary search algorithm is O(log n). The best-case time complexity would be O(1) when the central index would directly match the desired value.

Data structures an Algorithms

Advantages of Binary Search:

- It eliminates half of the list from further searching by using the result of each comparison.
- It indicates whether the element being searched is before or after the current position in the list.
- This information is used to narrow the search.
- For large lists of data, it works significantly better than linear search.

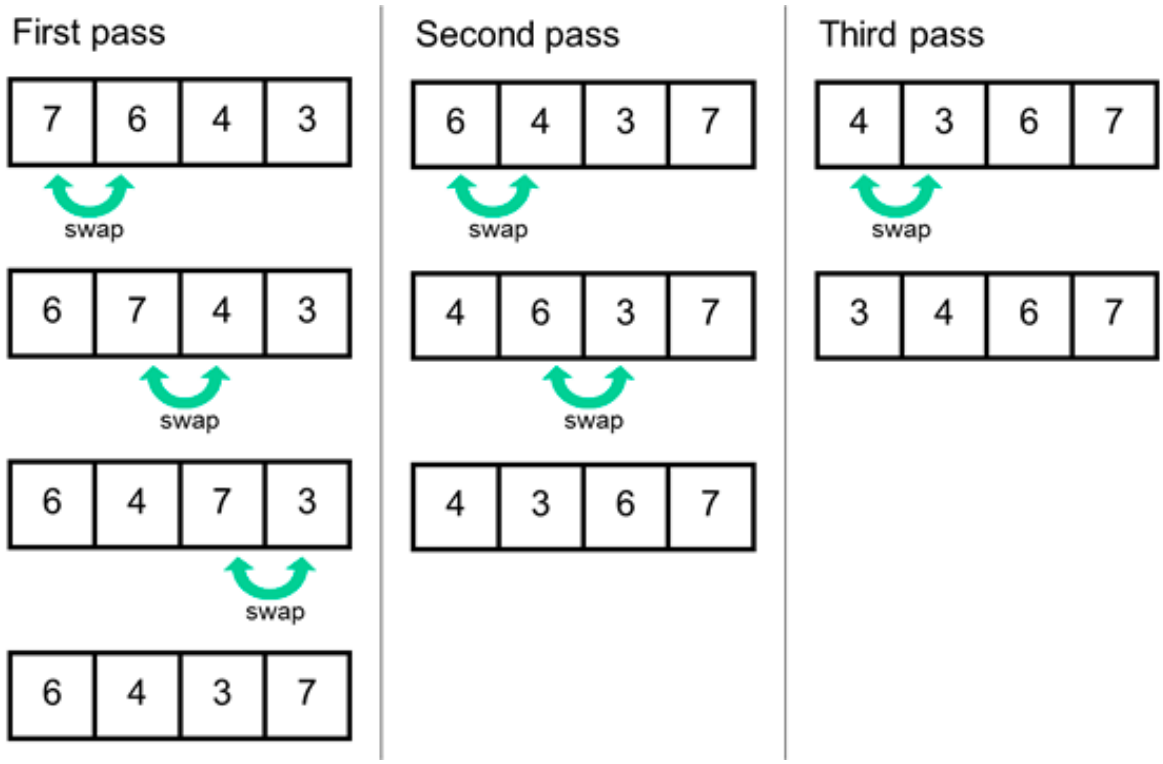
Drawbacks of Binary Search:

- It employs recursive approach which requires more stack space.
- Programming binary search algorithm is error prone and difficult.
- The interaction of binary search with memory hierarchy i.e. caching is poor.

Data structures an Algorithms

Bubble Sort

- Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in the wrong order.



Complexity Analysis of Bubble Sort

This algorithm has a worst-case time complexity of O(n2).
This algorithm is not suitable for large data sets as its average and worst-case time complexity is quite high.

Data structures an Algorithms

Advantages of Bubble sort

- Bubble sort is easy to understand and implement.
- It does not require any additional memory space.
- It's adaptability to different types of data.
- It is a stable sorting algorithm, meaning that elements with the same key value maintain their relative order in the sorted output.

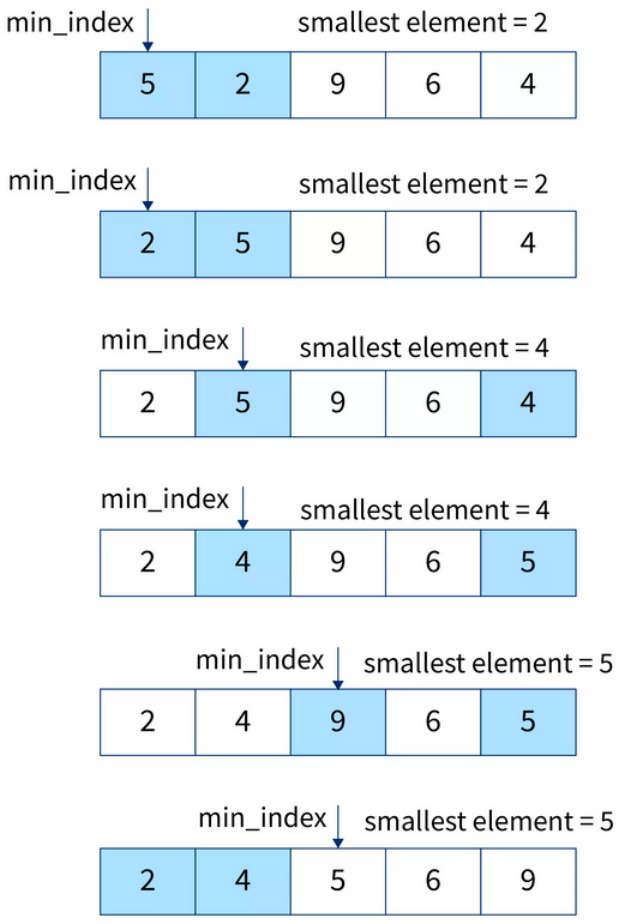
Drawbacks of bubble sort

- Bubble sort has a time complexity of O(n^2) which makes it very slow for large data sets.
- It is not efficient for large data sets, because it requires multiple passes through the data.

Data structures an Algorithms

selection Sort

- selection sort is an effective and efficient sort algorithm based on comparison operations. It adds one element in each iteration. You need to select the smallest element in the array and move it to the beginning of the array by swapping with the front element.



Complexity Analysis of selection Sort

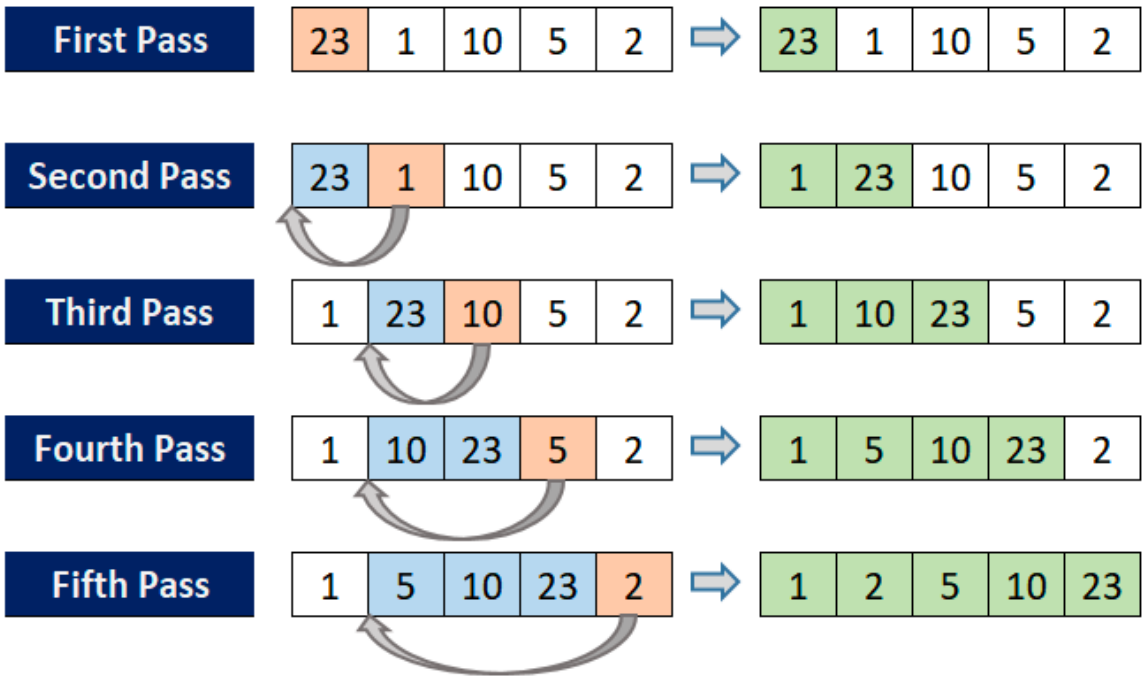
The time complexity of Selection Sort is O(N2) as there are two nested loops

- 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)

Data structures an Algorithms

Insertion Sort

- Insertion sort is a simple sorting algorithm that works similar to the way you sort playing cards in your hands. The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in the sorted part.



Complexity Analysis of Linear Search:

O(N^2)

KEEP LEARNING AND HAPPY CODING