Linear Search

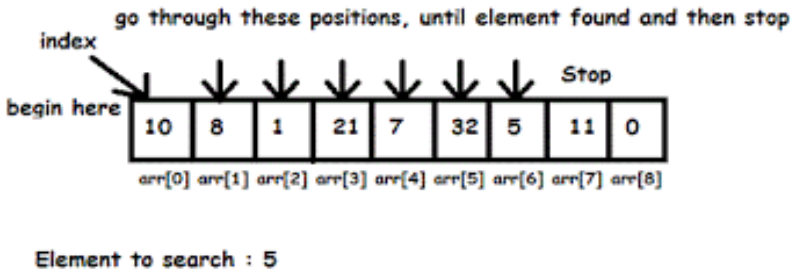
Linear search is the simplest searching algorithm. In this type of search, a sequential search is made over all items in a list (array or linked list in general) one by one. Every item is checked and if a match is found then that particular item is returned, otherwise the search continues till the end of the list.

Steps to do linear search in an array:

Step 1: Start from the leftmost element of array and one by one compare the element we are searching for with each element of the array.

Step 2: If there is a match between the element we are searching for and an element of the array, return the index.

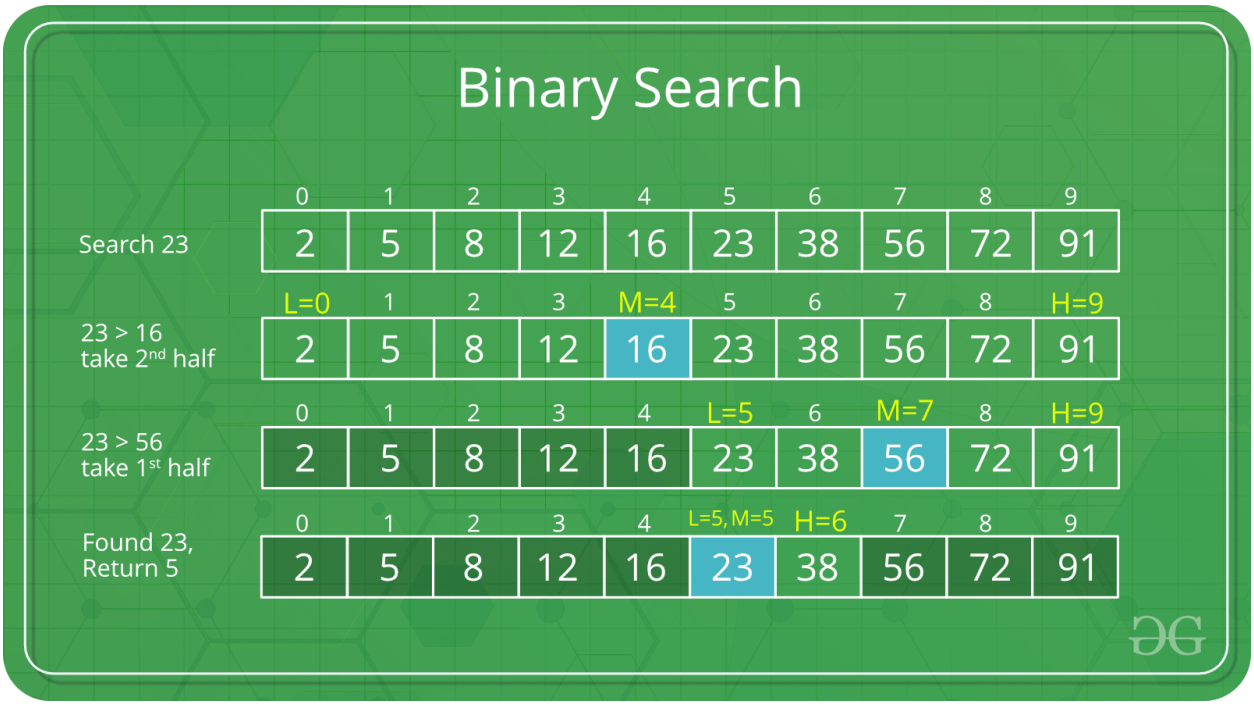
Step 3: If there is no match between the element we are searching for and an element of the array, return -1.



Abhishek Dey
Assistant Professor
Department of Computer Science
Bethune College, Kolkata

Binary Search

The prerequisite (or precondition) of binary search is the array must be sorted. Binary search looks for a particular item by comparing the middle most item of the array. If a match occurs, then the index of item is returned. If the middle item is greater than the item, then the item is searched in the sub-array to the left of the middle item. Otherwise, the item is searched for in the sub-array to the right of the middle item. This process continues on the sub-array as well until the size of the sub-array reduces to zero.



Pseudocode:

```
Procedure binary_search(A, n, T)
   L = 0 // lower bound
   R = n – 1 //upper bound
   while L ≤ R do
      m = (L + R) / 2 //mid index calculation
      if A[m] < T then
         L = m + 1
      else if A[m] > T then
         R = m - 1
      else
         return m
   return unsuccessful
```

Abhishek Dey
Assistant Professor
Department of Computer Science
Bethune College, Kolkata

Comparison of Linear Search and Binary Search

Basis for Comparison	Linear Search	Binary Search
Prerequisite / Precondition	Not required.	Array must be in sorted order.
Basic principle	Sequential search of elements one by one.	Based on Divide and Conquer approach.
Type of access	Strictly sequential.	Random.
Best case time complexity	When the first element is the element to be searched. Time complexity is O(1).	When the element to be searched is present in the middle of the array. Time complexity is O(1).
Worst Case time complexity for an array of n elements	Item is not present in the array or the last comparison is a match. n comparisons are needed. Time complexity is O(n).	Item is not present in the array or the last comparison is a match. In each iteration, the array is divided into half, so there will be maximum $\log_2 n$ such divisions. Time complexity is $O(\log_2 n)$.
Space Complexity	No extra space is required. So, space complexity is O(1).	No extra space is required. So, space complexity is O(1).
Average Case time complexity for an array of n elements	Time complexity is O(n).	Time complexity is $O(\log_2 n)$.
Whether suitable for linked list	Yes.	Not suitable because this method requires direct access to middle element.

Divide and Conquer

Divide and Conquer is an algorithmic paradigm. A typical Divide and Conquer algorithm solves a problem using following three steps.

1. **Divide**: Break the given problem into sub-problems of same type.
2. **Conquer**: Recursively or iteratively solve these sub-problems
3. **Combine**: Appropriately combine the solutions of sub-problems to obtain actual solution of the given problem.

Binary search follows Divide and Conquer because it divides a large array into two smaller sub-arrays and then discards one of the sub-arrays. This decision of discarding takes only one comparison. After that the same process is applied on the sub-array that was not discarded in the previous step. So, binary search reduces the search space by half at every step.

Abhishek Dey
Assistant Professor
Department of Computer Science
Bethune College, Kolkata

Reference books and websites:

1. Data Structures Using C by Reema Thareja
2. Data Structures Through C In Depth by S.K.Srivastava and Deepali Srivastava
3. https://www.geeksforgeeks.org/complexity-analysis-of-binary-search/
4. https://www.gatevidyalay.com/linear-search-searching-algorithms/

Abhishek Dey
Assistant Professor
Department of Computer Science
Bethune College, Kolkata