Binary Search

Binary Search is a searching algorithm for searching an element in a sorted list or array. Binary Search is efficient than Linear Search algorithm and performs the search operation in logarithmic time complexity for sorted arrays or lists.

Binary Search performs the search operation by repeatedly dividing the search interval in half. The idea is to begin with an interval covering the whole array. If the value of the search key is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise narrow it to the upper half. Repeatedly check until the value is found or the interval is empty.

Problem: Given a sorted array arr[] of N elements, write a function to search a given element X in arr[] using *Binary Search Algorithm*.

Algorithm: We basically ignore half of the elements just after one comparison.

- Compare X with the middle element of the array.
- If X matches with middle element, we return the mid index.
- Else If X is greater than the mid element, then X can only lie in right half subarray after the mid element. So we will now look for X in only the right half ignoring the complete left half.
- Else if X is smaller, search for X in the left half ignoring the right half.

Implementation: The Binary Search algorithm can be implemented both recursively and iteratively.

- **Recursive Function**:

```c
// A recursive binary search function. It returns
// location of x in given array arr[l..r] if present,
// otherwise -1
// Initially,
// l = 0, first index of arr[].
// r = N-1, last index of arr[].
int binarySearch(int arr[], int l, int r, int x)
{
    if (r >= l) {
        int mid = l + (r - l) / 2;
        // If the element is present at the middle
        // itself
        if (arr[mid] == x)
            return mid;
        // If element is smaller than mid, then
        // it can only be present in left subarray
        if (arr[mid] > x)
            return binarySearch(arr, l, mid - 1, x);
        // Else the element can only be present
        // in right subarray
        return binarySearch(arr, mid + 1, r, x);
    }
    // We reach here when element is not
    // present in array
    return -1;
}
```

- Iterative Function:

```
// A iterative binary search function. It returns
// location of x in given array arr[l..r] if present,
// otherwise -1
// Initially,
// l = 0, first index of arr[].
// r = N-1, last index of arr[].
int binarySearch(int arr[], int l, int r, int x)
{
    while (l <= r) {
        int m = l + (r - l) / 2;
        // Check if x is present at mid
        if (arr[m] == x)
            return m;
        // If x greater, ignore left half
        if (arr[m] < x)
            l = m + 1;
        // If x is smaller, ignore right half
        else
            r = m - 1;
    }
    // if we reach here, then element was
    // not present
    return -1;
}

```

Time Complexity: O(Log N), where N is the number of elements in the array.