LINEAR SEARCH

This search method searches for an element by ~~its~~ visiting all the elements sequentially until the element is found or array finishes. It follows array traversal.

→ Array can be sorted or unsorted
WC complexity - O(n)

BINARY SEARCH

This search method searches for an element by breaking the search space into half each time it finds the wrong element. This method is limited to a sorted array. The ~~sear~~ search continues towards either side of the mid, based on whether the element to be searched is lesser or greater than the mid-element.

→ Only sorted array
→ WC complexity = O(log n)

Linear search	Binary search
1) Works on both sorted & unsorted array	1) Works only on sorted array
2) Equality operations	2) Inequality operation
3) O(n) WC Complexity	3) O(log n) WC Complexity

```
#include<stdio.h>

int linearSearch(int &arr[], int size,
                 int element)
{
for(int i=0; i<size; i++)
{
if(arr[i]== element)
{
return i;
}
}
```

```
        return -1;
}

int binary search(int arr[], int size,
                  int element)
    {
int low, mid, high;
  low=0;
high = size-1;
// Keep searching until low<=high

while(low<=high){
   mid=(low+high)/2;
  if(arr[mid]==element){
      return mid;
    }
  if(array[mid]<element){
      low=mid+1;
    }
  else{
     high=mid-1;
    }
}

return -1;
}

int main(){
//Unsorted array for linear search
// int arr[]={1,3,5,56,4,3,23,4,5,
              54634,56,34};
// int size= sizeof(arr)/sizeof(int);
```

```
// Sorted assay for binary search
int ass[] = {1, 3, 5, 56, 64, 73, 123, 225,
                                        444};
int size = sizeof(ass) / sizeof(int);
int element = 444;
int searchIndex = binarySearch
                    (ass, size, element);
printf("The element %d was found
        at index %d \n", element,
                        searchIndex);
    return 0;
}
```

O/P : The element ~~was~~ 444 was found
at index