**Objective –**

Write a program to implement Linear Search in array

**Description –**

# Binary search

* Binary search in C language to find an element in a sorted array. If the array isn't sorted, you must sort it using a sorting technique such as merge sort. If the element to search is present in the list, then we print its location. The program assumes that the input numbers are in **ascending** order.
* A Binary Search is a sorting algorithm, that is used to search an element in a sorted array. A binary search technique works only on a sorted array, so an array must be sorted to apply binary search on the array. It is a searching technique that is better then the liner search technique as the number of iterations decreases in the binary search.
* The logic behind the binary search is that there is a key. This key holds the value to be searched. The highest and the lowest value are added and divided by 2. Highest and lowest and the first and last element in the array. The mid value is then compared with the key. If mid is equal to the key, then we get the output directly. Else if the key is greater then mid then the mid+1 becomes the lowest value and the process is repeated on the shortened array. Else if the key value is less then mid, mid-1 becomes the highest value and the process is repeated on the shortened array. If it is not found anywhere, an error message is displayed.
* A simple approach is to do [**linear search**](http://quiz.geeksforgeeks.org/linear-search/)**.**The time complexity of above algorithm is O(n). Another approach to perform the same task is using Binary Search.
* **Binary Search:** Search a sorted array by repeatedly dividing the search interval in half. 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.

Algorithm

The steps of binary search algorithm:

1- Select the element in the middle of the array.

2- Compare the selected element to the searched element, if it is equal to the searched element, terminate.

3- If the searched element is larger than the selected element, repeat the search operation in the major part of the selected element.

4- If the searched element is smaller than the selected element, repeat the search in the smaller part of the selected element.

5- Repeat the steps until the smallest index in the search space is less than or equal to the largest index.

Program –

#include<stdio.h>

#include<conio.h> //header files

void bubble\_sort(int [],int);

void binary\_search(int [],int,int); //function prototype

/\* starting point of the program \*/

int main(void)

{

int arr[25],i,x,n;

clrscr();

printf("\n\t Enter the number of elements=");

scanf("%d", &n);

printf("\n\t Enter array elements=");

for(i=0;i<n;i++) // for loop

scanf("%d", &arr[i]);

printf("\n\t Enter the element to be searched=");

scanf("%d", &x);

bubble\_sort(arr,n); //function call

binary\_search(arr,n,x);

getch();

}

/\* utility function to sort elements in ascending order \*/

void bubble\_sort(int arr[],int n) //function definition

{

int i,j,t;

for(i= n-2; i>=0 ;i--)

for( j = 0; j<= i;j++)

{

if(arr[j] > arr[j+1])

{

t = arr[j];

arr[j] = arr[j+1];

arr[j+1] = t;

}

}

}

/\* utility function to search the element \*/

void binary\_search(int arr[],int n,int x)

{

int left,right,mid;

left=0;

right= n-1;

while(left <= right)

{

mid = (left+right)/2; //formula to find mid

if(x == arr[mid])

break;

if(x > arr[mid])

left = mid +1;

else

right = mid -1;

}

if(left>right)

printf("\n\t %d element not found ! ", x);

else

printf("\n\t %d element is found at %d position ", x,mid+1);

}

Output-

