## Binary Search

In computer science, 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 is faster than linear search except for small arrays. However, the array must be sorted first to be able to apply binary search.

## Algorithm –

- **Step 1:** START
- **Step 2:** Initialize low = 0, high = size-1, flag = 0
- **Step 3:** Repeat Steps 3 and 4 while low <= high
- **Step 4:** SET mid = (low + high)/2
- **Step 5:**
  - if value == arr[mid] set flag = 1 and stop
  - else if value > arr[mid] set low = mid+1
  - else if value < arr[mid] set high = mid - 1
- **Step 6:** if flag==1 value found else not found
- **Step 7:** STOP

## Program

```
#include<stdio.h>
int main()
{
        int num = 0,value,low,high,mid,flag=0;
        printf("Enter how many element you want = ");
        scanf("%d",&num);
```

```
//Filter
if(num<=0)
{
        printf("Invalid Size\n");
        return -1;
}
int arr[num];  //array creation

//Accept Value
for(int i=0;i<num;i++)
{
        printf("Enter Number = ");
        scanf("%d",&arr[i]);
}
//Display Array
printf("Array = ");
for(int i=0;i<num;i++)
{
        printf("%d ",arr[i]);
}
//Accepting Value to Search
printf("\nEnter Value to search = ");
scanf("%d",&value);

//initilization of low, high and mid
low = 0,high = num-1, mid = (low+high)/2;

//binary search logic start from here
while(low<=high)
{
        mid = (low+high)/2;
```

```
            if(value == arr[mid])
            {
                flag = 1;
                break;
            }
            else if(value > arr[mid])
            {
                low = mid+1;
            }
            else if(value < arr[mid])
            {
                high = mid-1;
            }
        }
        if(flag == 1)
        {
            printf("Value found at %d index",mid);
        }
        else
        {
            printf("Value not found");
        }
        //binary search logic end here
        return 0;
}
```