2. Binary Search

Theory-

→ Here DATA is a sorted array with lower bound LB and upper bound UB, and ITEM is a given item of information.

→ The variables BEG, END and MID denote, respectively the beginning, end and middle location of a segment of elements of DATA.

Algorithm-

This algorithm finds the location LOC of ITEM in DATA or sets LOC=Null

1. [Initialize segment variables]
   Set BEG=LB, END=UB and MID=INT(BEG+END)/2
2. Repeat steps 3 and 4 while BEG≤END and DATA[MID]≠ITEM
3. If ITEM<DATA[MID] then:
   Set END=MID-1
   Else:
   Set BEG=MID+1
   [END of if structre]
4. Set MID=INT(BEG+END)/2
   [END of step 2 loop]
5. If DATA[MID]=ITEM then:
   Set LOC=MID
   Else:
   Set LOC=Null [END of If structre]
6. Exit

Process-

* Let us take an array of 5 elements, A={10,20,30,40,50} and Let K=40
* The user wants to search element 40 and
* Initializing L=0, h=n-1, m=(L+h)/2
* Since L<h loop runs

First operation -

[m=(0+4)/2 = 2]

a[m] == 40
i.e 30 == 40
Not true

Since 40 > a[m]
L=m+1
m=(L+h)/2

Second iteration -

[m=(3+4)/2 = 3]

a[m] == 40
i.e 40 == 40
True

* Hence we will get the element Rs found and loop breaks.

Output-

```
Enter search element: 8
Enter sorted array:
0
2
4
8
9
8 found at 4
```

Code-

```
#include<stdio.h>
int main()
{
  int se,A[],n=5;
  printf("enter search element:");
  scanf("%d",&se);
  printf("enter sorted array:");
  for(int i=0;i<n;i++)
     scanf("%d",&A[i]);
  int low=0,high=n-1,mid=(low+high)/2
  while(low<=high){
    if(A[mid]<se)
       low=mid+1
    else{if(A[mid]==se){
          printf("%d found at %d",se,mid+1);
          break;}
       else
          high=mid-1;}
       mid=(low+high)/2;
    }
    if(low>high)
       printf("Not found");
    return 0;
}
```