**IterativeBinarySearch**

**Dr. Rahul Das Gupta**

**Sequential Search**

#include <stdio.h>

#include <conio.h>

intSequentialSearch(int \*, int , int);

void main()

{

    intarr[] = { 2, 9, 4, 8,5, 40, 35, 10 };

    int n = sizeof(arr) / sizeof(arr[0]);

    int x = 10;

    int result = SequentialSearch(arr, n , x);

    if(result == -1)

printf("\n Element is not presentin the array.");

else

printf("\n Element %d is present at index %d", x,

                            result);

}

**int SequentialSearch(int \*p, intn, int key )**

**{**

inti;

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

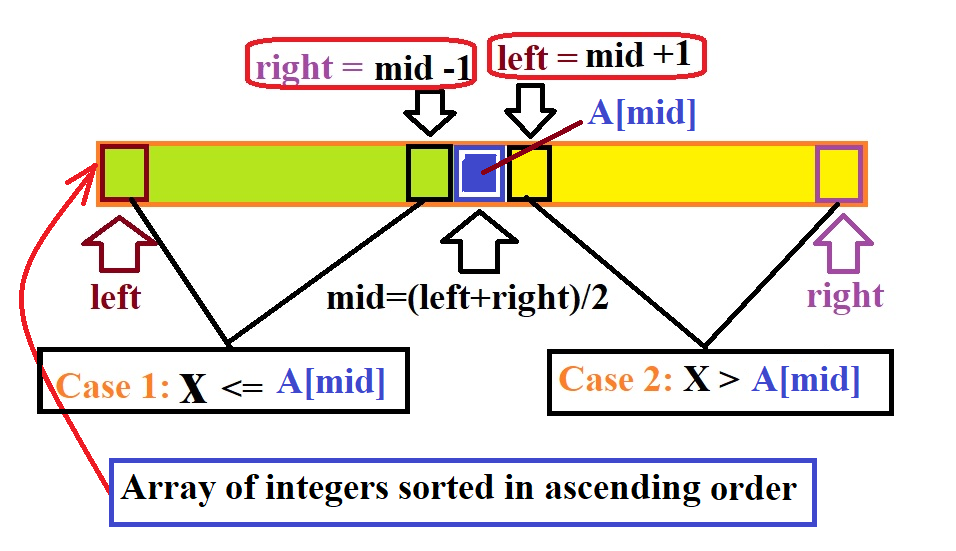
if(key==p[i])

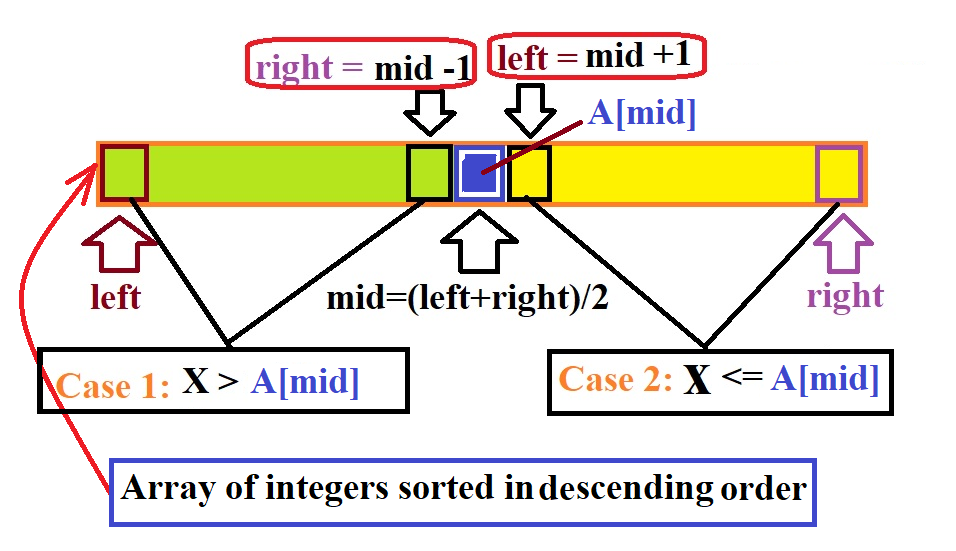
returni; **/\* Returning the location of the key element inside the array.\*/**

return -1; **/\*Here, key element is not present in the given array.\*/**

**}**

**Maximum number of comparisons required for SequentialSearch is n.**

****

****

**IterativeBinarySearch**

#include <stdio.h>

#include <conio.h>

#define ASCENDING 0

#define DESCENDING 1

**/\*Aniterative binary search functionreturns the**

**location of a key element x in a given sorted array**

**(sorted either in ascending order or descending order) arr[left…right] if it is present, otherwise -1.\*/**

intIterativeBinarySearch(int \*, int, int , int, int);

void main()

{

    intarr[] = { 2, 3, 4, 5,8, 10, 35, 40 };

    int n = sizeof(arr) / sizeof(arr[0]);

    int x = 10;

    int result

if (**arr[left]<arr[right]**)

/\***Case 1:Array is sorted in Ascending Order when**

**the condition(arr[left]<arr[right])is TRUE.\*/**

result= IterativeBinarySearch(arr, ASCENDING,

0, n - 1, x);

else

/\***Case 2:Array is sorted in Descending Order when**

**the condition(arr[left]<arr[right])is FALSE.\*/**

result= IterativeBinarySearch (arr, DESCENDING,

0, n - 1, x);

 if(result == -1)

printf("\n Element is not presentin the array.");

else

printf("\n Element %d is present at index %d",

x, result);

}

**intIterativeBinarySearch(int \*p, int type, int left, int right, intkey)**

{

    int mid;

if(type== ASCENDING)

{

**/\*Case 1:Array is sorted in Ascending Order\*/**

**while (left <= right)**

**{**

        mid = (left+right) / 2;

**/\* Check if x is present at mid \*/**

        if (p[mid] = = key)

            return mid;

**/\* When searching element is greater than themiddle element. \*/**

elseif (p[mid] <key)

      left = mid + 1;**/\*The element will be searched in right sub array from mid+1toright.\*/**

else

**/\* When searching element is smaller than themiddle element. \*/**

right = mid -1;**/\*The element will be searched in left sub array from leftto mid-1.\*/**

}**/\*End of while (left<=right) \*/**

return -1;**/\*If we reach here, then element was**

**not present. \*/**

}

else

{

**/\*Case 2:Array is sorted in Descending Order\*/**

**while (left <= right)**

**{**

        mid = (left+right) / 2;

**/\* Check if x is present at mid \*/**

        if (p[mid] = = key)

            return mid;

**/\* When searching element is smaller than themiddle element. \*/**

elseif (p[mid] > key)

      left = mid + 1;**/\*The element will be searched in right sub array from mid+1toright.\*/**

else

**/\* When searching element is greater than themiddle element. \*/**

right = mid-1;**/\*The element will be searched in right sub array from lefttomid-1.\*/**

**}/\*End of while (left<=right) \*/**

    return -1;**/\*If we reach here, then element was**

**not present. \*/**

}

}

**\*\* Binary Search can only be performed on an array which is sorted either in ascending or descending order.**

Complexity **Analysis of Binary Search**

**Go to next page and continue.**