**PEAK OF MOUNTAIN ARRAY**

A mountain array is an array divided in 2 parts.In one part elements are arranged in ascending order and other part has elements in descending order.There comes a point where a peak of these elements is achieved where peak is the greatest element among all elements present

EXAMPLE = 1 2 3 4 **5** 4 3 2 1

PEAK = 5

Here elements are ascending before 5 and then start decreasing after 5 following a mountain approach,due this formation this array is named as **mountain array**

**PROGRAM**

import java.util.Scanner;

class Binary\_search\_algorithm{

public static void main(String[] args){

Scanner scanf = new Scanner(System.in);

int[] arr = new int[100];

int i=0,j=0,l=0,t=0,b=1;

System.out.print("Enter Length\t");

l = scanf.nextInt();

System.out.println("");

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

{

System.out.print("ELEMENT["+(i+1)+"]\t");

arr[i] = scanf.nextInt();

}

System.out.println("");

int a = search(arr,l);

System.out.println("");

if(a!= 0) {System.out.println("PEAK NUMBER FOUND AT PLACE \t "+(a+1));} // **a+1 as index starts** **from 0**

else System.out.println("MISSING");

System.out.println("\nCLICK 1 TO CONTINUE\nENTER:");

b = scanf.nextInt();

scanf.close();

}

static int search(int arr[],int l){

int i = 0;

int mid = 0,start=0,end = l-1;

while(start<end)

{

mid = (start + end)/2;

if(arr[mid] < arr[mid+1]){

start = mid+1;  **//ASCENDING PART**

}

if(arr[mid]>arr[mid+1]){

end = mid;  **//DESCENDING PART**

}

}

return start; **// return start; valid as loop ends at single peak element so end = start**

}

}