Data structures and Algo in Java - Day 20

import java.util.Arrays;

/\*

\* We are going to find first and last occurrences of x in a given array

\* there are 2 types of solutions for this

\* find lower bound and find upper bound

\* create an array of lower bound and upper bound - 1

\* or if the interviewer asks not to use lower upper bounds

\* we create a simple binary search in this condition

\* finding first and last occurrences

\* The second question is to count the number of occurrences of a number including duplicates

\* for this question we just have to find the last occurrence minus first occurrence plus 1

\*/

public class day20

{

// let us recap binary search algorithm

public static void main(String [] args)

{

int arr [] = {3,4,6,8,8,8,8,8,9,11,13};

int target = 8;

countTheOccurences(arr, target);

}

public static void binarySearch(int arr[],int tar)

{

int low = 0;

int high = arr.length -1 ;

int ans = search(arr,tar,low,high);

if(ans==-1)

{

System.out.println("Target Element Not Found");

}

else

{

System.out.println("Target Element Found at Index "+ans);

}

}

public static int search(int arr[],int tar,int low,int high)

{

if(low>high)

{

return -1;

}

int mid = low+(high-low)/2;

if(arr[mid]==tar)

{

return mid;

}

else if(arr[mid]>tar)

{

return search(arr,tar,low,mid-1);

}

else

return search(arr,tar,mid+1,high);

}

// okay this is binary search and it works well , im so proud of myself because i didnt look into the answer

// now lets solve the first question of occurrences using lower and upper bound

public static void firstAndLastOccurrencesUsingLBandUB(int arr[],int x)

{

int low = 0;

int n = arr.length;

int high = n-1;

int lb = lowerBound(arr,x,low,high);

int ub = upperBound(arr, x, low, high);

int ans [];

if (lb == n || arr[lb]!=x)

{

ans = new int []{-1,-1};

System.out.println(Arrays.toString(ans));

}

else

{

ans = new int []{lb,ub-1};

System.out.println(Arrays.toString(ans));

}

}

public static int lowerBound(int arr[],int target,int low,int high)

{

int lb = arr.length;

while(low<=high)

{

int mid = (low+high)/2;

if(arr[mid]>=target)

{

lb = mid;

high = mid -1;

}

else

{

low = mid+1;

}

}

return lb;

}

public static int upperBound(int arr[],int target,int low,int high)

{

int ub = arr.length;

while(low<=high)

{

int mid = (low+high)/2;

if(arr[mid]>target)

{

ub= mid;

high = mid -1;

}

else

{

low = mid+1;

}

}

return ub;

}

// that works perfectly fine lets dive into basic binary search without using lb and ub

public static void firstAndLastOccurrencesUsingBS(int arr[],int x)

{

int first = first(arr, x);

int last = last(arr, x);

int ans [];

ans = new int [] {first,last};

System.out.println(Arrays.toString(ans));

}

public static int first(int arr[], int x)

{

int low = 0;

int high = arr.length -1;

int first = -1;

while(low<=high)

{

int mid = (low+high)/2;

if(arr[mid]==x)

{

first = mid;

high = mid -1;

}

else if (arr[mid]<x)

{

low = mid + 1;

}

else

{

high = mid -1;

}

}

return first;

}

public static int last(int arr[],int x)

{

int low = 0;

int high = arr.length -1;

int last = -1;

while(low<=high)

{

int mid = (low+high)/2;

if(arr[mid]==x)

{

last = mid;

low = mid +1;

}

else if (arr[mid]>x)

{

high = mid -1;

}

else

{

low = mid + 1;

}

}

return last;

}

// and its done , we have did it . completed both types of solutions . now to the couting one

public static void countTheOccurences(int arr[], int x)

{

// the hint for this question is to find the last occurrence minus first occurrence plus one

int first = firstOccurrence(arr, x);

int last = lastOccurrence(arr, x);

if (first == -1 || last == -1) {

System.out.println(x + " is not present in the array.");

return;

}

int count = last - first + 1;

System.out.println("There are "+count+" "+x+"'s in the Array");

}

public static int firstOccurrence(int arr[],int x)

{

int low = 0;

int high = arr.length -1 ;

int first = -1 ;

while(low<=high)

{

int mid = (low+high)/2;

if(arr[mid]==x)

{

first = mid;

high = mid -1;

}

else if (arr[mid]<x)

{

low = mid+1;

}

else

{

high = mid -1;

}

}

return first;

}

public static int lastOccurrence(int arr[],int x)

{

int low = 0;

int high = arr.length -1 ;

int last = -1 ;

while(low<=high)

{

int mid = (low+high)/2;

if(arr[mid]==x)

{

last = mid;

low = mid+1;

}

else if (arr[mid]>x)

{

high = mid -1;

}

else

{

low = mid+1;

}

}

return last;

}

}