ASSIGNMENT- (SEARCHING)

Q1. Given an array. Find the number X in the array. If the element is present, return the index of the element, else print “Element not found in array”. Input the size of array, array from user and the element X from user. Use Linear Search to find the element .

Soln.

```
import java.util.Scanner;

public class Searching_Assg1 {

    static int linearSearch(int[] arr, int x) {
        int ans = -1;
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == x) {
                ans = i;
            }
        }
        return ans;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements you want to
add: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter the elements of the array");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        System.out.print("Enter the element to be searched in array:
");
        int x = sc.nextInt();
        int ans = linearSearch(arr, x);
        if (ans == -1) {
            System.out.println("Element not found in array");
        } else {
            System.out.println("Element found in array on the index:
" + ans);
        }
```

```
    }
}
```

Q2. Given an array and an integer “target”, return the last occurrence of “target” in the array. If the target is not present return -1.

Soln.

```
import java.util.*;

public class Searching_Assg2 {

    static int findOccurence(int[] arr, int target) {
        int start = 0, end = arr.length - 1, ans = -1;
        while (start <= end) {
            int mid = start + (end - start) / 2;
            if (arr[mid] == target) {
                ans = mid;
                start = mid + 1;
            } else if (arr[mid] < target) {
                start = mid + 1;
            } else {
                end = mid - 1;
            }
        }
        return ans;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements you want to
add: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter the elements of the array");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        System.out.print("Enter Target: ");
        int target = sc.nextInt();
        int ans = findOccurence(arr, target);
        if (ans == -1) {
            System.out.println("Element not found in array");
```

```
        } else {
            System.out.println("Last occurence of target in array
is: " + ans);
        }
    }
}
```

Q3. Given a sorted binary array, efficiently count the total number of 1's in it.

Soln.

```
import java.util.*;

public class Searching_Assg3 {

    static int countDigit(int[] arr) {
        int left = 0;
        int right = arr.length - 1;
        while (left <= right) {
            int mid = left + (right - left) / 2;
            if (arr[mid] == 1) {
                right = mid - 1;
            } else {
                left = mid + 1;
            }
        }
        return arr.length - left;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements you want to
add: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter the elements of the array");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        int ans = countDigit(arr);
        System.out.println("Total no. of 1s in the array is: " +
ans);
```

```
    }
}
```

Q4. Given a sorted integer array containing duplicates, count occurrences of a given number. If the element is not found in the array, report that as well.

Soln.

```
import java.io.*;
import java.util.*;

public class Searching_Assg4 {

    public static int lastOccurrence(int[] a, int low, int high, int
target) {
        int answer = -1;
        while (low <= high) {
            int mid = low + (high - low) / 2;

            if (a[mid] == target) {
                answer = mid;
                low = mid + 1; // if you found the target or if
target is greater than the current element, to
                               // find last occurrence move to right
half of the array
            } else if (a[mid] > target) {
                high = mid - 1;
            } else
                low = mid + 1;
        }
        return answer;
    }

    public static int firstOccurrence(int[] a, int low, int high,
int target) {

        int answer = -1;

        while (low <= high) {

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

```
            if (a[mid] == target) {
                answer = mid;
                high = mid - 1; // trying to find the minimum index
at which value x is present
            } else if (a[mid] > target) {
                high = mid - 1;
            }

            else
                low = mid + 1;
        }

        return answer;
    }

    public static void main(String args[]) {

        int m;
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements you want to
add : ");
        m = sc.nextInt();

        int[] arr = new int[m];

        System.out.print("Enter the elements of the array: ");

        for (int i = 0; i < m; i++) {
            arr[i] = sc.nextInt();
        }

        int target;
        Scanner sc1 = new Scanner(System.in);
        System.out.print("enter the target: ");
        target = sc1.nextInt();

        int first = firstOccurrence(arr, 0, m - 1, target);
        int last = lastOccurrence(arr, 0, m - 1, target);

        if (first == last && first == -1)
            System.out.println("The target does not exist in the
array.");
```

```
        else
            System.out.println("The frequency of target in the given
array is " + (last - first + 1) + " time/times");
    }
}
```

Q5. Given a positive integer num, return true if num is a perfect square or false otherwise. A perfect square is an integer that is the square of an integer. In other words, it is the product of some integer with itself.

Soln.

```
import java.util.Scanner;

public class Searching_Assg5 {

    static boolean perfectSquare(int num) {
        int start = 0, end = num / 2;
        while (start <= end) {
            int mid = start + (end - start) / 2;
            int val = mid * mid;
            if (val == num) {
                return true;
            } else if (val < num) {
                start = mid + 1;
            } else {
                end = mid - 1;
            }
        }
        return false;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number: ");
        int n = sc.nextInt();
        boolean ans = perfectSquare(n);
        System.out.println("Is " + n + " is a prefect Square? " +
ans);
    }
}
```