JAVA Logic Placement Preparation Test 3

PRN : 030

Q.1) Segregate positive and negative integers in linear time Given an array of positive and negative integers, . The output should print all negative numbers, followed by all positive numbers. For example, Input: [19, -13, 15, -12, -18, -16, 1, 3] Output: [-13, -12, -18, -16, 15, 19, 1, 3

=>

```
package java_logic;
import java.util.Arrays;
public class Negative_Positive_Series {
    public static void main(String[] args) {
        int[] arr = {19,-13,15,-12,-18,-16,1,3};
        int n = arr.length;
        int[] ans = negPos(arr,n);
        Arrays.sort(ans);
        for(int num:ans) {
            System.out.print(num + " ");
        }
    }
    static int[] negPos(int[] arr, int n) {
        int[] res = new int[n];
        int left=0;
        int right=n-1;
        for(int i=0;i<n-1;i++) {
            if(arr[i]<0) {     res[left++]=arr[i]; }else {
                res[right--]=arr[i];        }        }
        return res;
    }
}
```

Q.2) Accept 5 number in an array, accept a number from user and check if given number is there in an array or not

=>

```
package java_logic;

import java.util.*;

public class Linear_Search_1 {

    public static void main(String[] args) {

        System.out.println("Enter 5 numbers in an array : ");

        Scanner sc = new Scanner(System.in);

        int[] arr = new int[5];

        for(int i=0;i<arr.length;i++) {

            arr[i]=sc.nextInt();

        }

        System.out.println("Enter number to be search : ");

        int key = sc.nextInt();

        linearSearch(arr,key);    }

    static void linearSearch(int[] arr, int key) {

        boolean found = false;

        int i;

        for(i=0;i<arr.length-1;i++) {

            if(arr[i]==key) {

                found=true;

                break;        }            }

        if(found) {System.out.println("number found at index : "+i);}else {

            System.out.println("Number not found!");

        }

    }

}
```

}

Q.1) Accept 10 number in an array. Display all even number at the beginning and all Odd at the end. Use only one loop

=>

```
package java_logic;

import java.util.Scanner;

public class Even_Odd_inSeries {

    public static void main(String[] args) {

        System.out.println("Enter 10 numbers (mix of even and odd) :");

        Scanner sc = new Scanner(System.in);

        int[] nums = new int[10];

        for(int i=0;i<nums.length;i++) {

            nums[i]=sc.nextInt();

        }

        int[] ans = evenOdd(nums);

        for(int num:ans) {System.out.print(num+" ");}    }

    static int[] evenOdd(int[] arr) {

        int[] res = new int[arr.length];

        int left=0, right=arr.length-1;

        for(int num:arr) {

            if(num%2==0)

                res[left++]=num;

            else

                res[right--]=num;

        }

        return res;

    }


}
```

Q.2) Accept 5 number in an array and sort it. Accept a number from user and check if it is there in an array or not use binary search.

=>

```
package java_logic;
import java.util.Arrays;
import java.util.Scanner;
public class Binary_Search_2 {
    public static void main(String[] args) {
        System.out.println("Enter 5 numbers :");
        Scanner sc = new Scanner(System.in);
        int[] arr = new int[5];
        for(int i=0;i<arr.length;i++) {
            arr[i]=sc.nextInt();
        }
        System.out.println("Enter a number to search: ");
        int key = sc.nextInt();
        binarySearch(arr,key);   }
    static void binarySearch(int[] arr, int key) {
        Arrays.sort(arr); // 1 2 3 4 5
        boolean found = false;
        int mid=0;
        int left=0;
        int right=arr.length-1;
        while(left<right) {
            mid = (right+left)/2;

            if(arr[mid]==key) {
                found = true;
                break;
            }else if(arr[mid]<key) {
                left=mid+1;
            }else {
                right=mid-1;
            }
        }


        if(found) {
            System.out.println("number "+key+" found at index : "+mid);
        }else {
            System.out.println("number not found !");
        }
    }

}
```