Softnerve Tech Assessment

Name: Akhil Dhiman

Ph.no: +91 93502 88980

E-mail: dhimanakhil75@gmail.com

Problem 1 : **Leader in the Array**

Language : **Java**

Code:

```
import java.util.*;

import java.lang.*;


public class Problem1{


   //Function to find the leaders in the array.

   public static void leaders(int arr[], int n){


   //ArrayList to print leader elements in order of their appearance

      ArrayList<Integer> a = new ArrayList<Integer>();


      a.add(0,arr[n-1]);
```

```
        }


        int max=arr[n-1];


        /*Traversing the array from right to left to chech if

        all elements to the right of current element are small


        and if this condition met, we store it in our ArrayList*/


        for(int i=n-2;i>=0;i--){
            if(arr[i] > max)
            {
                max = arr[i];
                a.add(arr[i]);
            }
        }


    /*Order of elements is reverse to the order of their appearance in the original
array
    so has to reverse the ArrayList to print the elements in order of their
appearance*/


       for (int i = 0; i < a.size() / 2; i++) {
            Integer temp = a.get(i);
            a.set(i, a.get(a.size() - i - 1));
            a.set(a.size() - i - 1, temp);
        }
        //print the elements in order of their appearance
        for(int i = 0;i<a.size();i++){
           System.out.print(a.get(i)+" ");
        }
        System.out.println(" ");
    }


public static void main(String[] args) {

   int arr[] = {7, 10, 4, 10, 6, 5, 2};
   int n=arr.length;
   leaders(arr,n);
}
}
```

Time Complexity : O(n)

Space Complexity : O(n)

OR

Solution with O(1) Space Complexity but the order of leaders is in reverse order

Code:

```
import java.util.*;


import java.lang.*;


public class Problem1{
   //Function to find the leaders in the array.


   public static void leaders(int arr[], int n){





      //variable to keep track of maximum element
      int max=Integer.MIN_VALUE;



      /*Traversing the array from right to left to chech if


      all elements to the right of current element are small
```

```
        and if this condition met, then we print our value and assign
        it to max*/



        for(int i=n-1;i>=0;i--){

            if(arr[i] > max)

            {
                max = arr[i];
                System.out.print(max + " ");

            }

        }
       System.out.println(" ");

    }

  //Driver method
public static void main(String[] args) {

   int arr[] = {7, 10, 4, 10, 6, 5, 2};
  int n=arr.length;

     //calling the function
  leaders(arr,n);

}
}
```

Time Complexity : O(n)

Space Complexity : O(1)

Problem 2 : **<u>Best time to buy and sell stocks</u>**

Language : **<u>Java</u>**

Code:

```
public class Problem2{

//function to calculate maximum pfoit
public static int stockProfit(int[] arr)
{

   int n = arr.length;
```

```
/*variables to keep track of the maximum and minimum price
 for each day*/
int min = arr[0];
int max = arr[0];



   int max_prf = 0;
   int diff=max-min;

   /*Calculating profit on each day using the pair of
   least buying price and highest sale price*/


   if(diff>max_prf)
   {
      max_prf = diff;
   }

 for(int i=1;i<n;i++)
 {
   if(arr[i] < arr[i-1]){
      if(arr[i] < min){
           min = arr[i]; }

        max = 0;

   }
   else if(arr[i] > arr[i-1]){

      max = arr[i];

   }
   if(max-min > max_prf){
      max_prf=(max-min);
   }

 }
 // Returning the maximum profit
   return max_prf;
}



//Driver Code
public static void main(String[] args){
   int[] arr = {7,1,5,3,6,4};
   int max_profit = stockProfit(arr);
```

```
        System.out.println(max_profit);
    }
}
```

Time Complexity : O(n)

Space Complexity : O(1)

Problem 3 : **<u>Sum of All Subset XOR Totals</u>**

Language : **<u>Java</u>**

Code:

```
public class Problem3{

    //function to calculate XOR sum of all the elements of array
    static void subset(String s,int idx,String newstring,int[] sum){

        /* Base Case: If subset is found,then we will find the XOR of all
        its elements and Will add it to our final sum*/

        if(idx==s.length()){


            //if suset is empty, we will print the sum as zero
```

```
if(newstring  == null){

   sum[0]+=0;
   return;

}



//if subset is a single element,then its XOR is its value itself//
if(newstring.length()==1){

   sum[0]+=(int)(newstring.charAt(0)-'0');
   return;

}

/*else we will find XOR of all elements of subset,and will
add it into our final value*/
else{

   int XOR=0;

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

            XOR = XOR^((int)(newstring.charAt(i)-'0'));

        }

        sum[0]+=XOR;

      }

      return;

    }

    /*  At every stage,every character of the array has the choice to either
    be included in subset,or not*/

    char a = s.charAt(idx);

    //To be included
    subset(s,idx+1,newstring+a,sum);

    //To be not included
    subset(s,idx+1,newstring,sum);

}
```

```
//Helping Function to convert array to String
public static int subsetXORSum(int[] nums) {




    String s="";


    String nl ="";



    for(int i = 0; i<nums.length; i++){
        s=s+(char)(nums[i] + '0');


    }


    String cpy =s;


    int[] sum={0};
    int sum2=0;


    subset(s,0,nl,sum);
```

```
        sum2=sum[0];

        return sum2;

    }

    //Driver Code
    public static void main(String[] args){
        int[] arr = {5,1,6};
        int XORsum = subsetXORSum(arr);
        System.out.println(XORsum);
    }
}
```

Time Complexity : O(n)

Space Complexity : O(1)   (O(n) if String is counted)