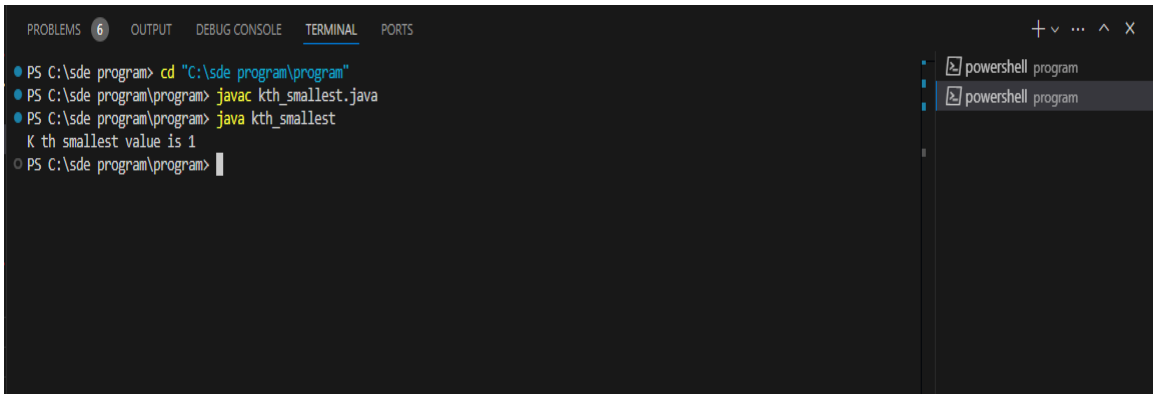
Coding practice Problems(13/11/2024)

Q 1) Kth Smallest Element

```
import java.util.Arrays;

import java.util.Collections;

class kth_smallest {

   public static int kth_smallest(Integer[] arr,int K)

   {

      Arrays.sort(arr);

      return arr[K-1];

   }

   public static void main(String[] args){

      Integer arr[]=new Integer[]{23,34,56,67,89,45,1,0,44,90,11,34,562,};

      int K=2;

      System.out.println("K th smallest value is "+ kth_smallest(arr,K));

   }

}
```

Time Complexity: O(n log n)

Output:



Q 2) Minimize the Height II

```
import java.util.Arrays;

class getMinDiff {
```

```
    int getMinDiff(int[] arr, int k){
        int n=arr.length;
        Arrays.sort(arr);
        int res=arr[n-1]-arr[0];
        for(int i=1;i<n;i++){
            if (arr[i]>=k){
                int max=Math.max(arr[i-1]+k,arr[n-1]-k);
                int min=Math.min(arr[0]+k,arr[i]-k);
                res=Math.min(res,max-min);
            }
        }
        return res;
    }
    public static void main(String[] arg){
        getMinDiff obj= new getMinDiff();
        int arr[]={34,56,77,86,56,76};
        int k=2;
        int result=obj.getMinDiff(arr,k);
        System.out.println("Minimum Difference is " + result);
    }
}
```

Time Complexity: O(n)

Output:

Q 3) Parenthesis Checker

```
import java.util.Stack;

public class parentheses_checker {

   public static boolean ispar(String s) {

      Stack<Character> stk = new Stack<>();

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

         if (s.charAt(i) == '(' || s.charAt(i) == '{' || s.charAt(i) == '[') {

            stk.push(s.charAt(i));

         }

         else {

            if (!stk.empty() &&

               ((stk.peek() == '(' && s.charAt(i) == ')') ||

                (stk.peek() == '{' && s.charAt(i) == '}') ||

                (stk.peek() == '[' && s.charAt(i) == ']'))) {

               stk.pop();

            }

            else {

               return false;

            }

         }

      }

      return stk.empty();
```

```
    }
    public static void main(String[] args) {
        String s = "{()}[]";
        if (ispar(s))
            System.out.println("true");
        else
            System.out.println("false");
    }
}
```

Time Complexity: O(n)

Output:

Q 4) Equilibrium Point

```
public class equilibrium_point {
    public static int findEquilibriumPoint(long[] arr) {
        int n = arr.length;
        long totalSum = 0;
        for (long num : arr) {
            totalSum += num;
        }
        long leftSum = 0;
        for (int i = 0; i < n; i++) {
            if (leftSum == totalSum - leftSum - arr[i]) {
                return i + 1;
```

```
        }
        leftSum += arr[i];
    }
    return -1;
  }
  public static void main(String[] args) {
    long[] arr = { 1, 3, 5, 2, 2 };
    System.out.println(findEquilibriumPoint(arr));
  }
}
```

Time Complexity: O(n)

Output:

Q 5) Check for binary

```
import java.util.*;
public class binary_checker {
        public static boolean checkrules(String s)
        {
                int n = s.length();
                int i = 0;
                if (s.charAt(i) != '1') {
                        return false;
                }
```

```
        i++;
        while (i < n) {
            if (s.charAt(i) == '1') {
                i++;
            }
            else if (i + 1 < n && s.charAt(i) == '0'
                    && s.charAt(i + 1) == '0') {
                i += 2;
                if (i < n && s.charAt(i) != '1') {
                    return false;
                }
            }
            else {
                return false;
            }
        }
        return true;
    }
    public static void main(String[] args)
    {
        String str = "1111";
        if (checkrules(str)) {
            System.out.println("Valid String");
        }
        else {
            System.out.println("Invalid String");
        }
```

```
        }
}
```

Time Complexity: O(n)

Output:

Q 6) Next Greater Element

```
import java.util.Arrays;

import java.util.Stack;

public class next_greater_element {

   public static int[] findNextGreaterElements(int[] arr) {

      int n = arr.length;

      int[] result = new int[n];

      Stack<Integer> stack = new Stack<>();

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

         while (!stack.isEmpty() && stack.peek() <= arr[i]) {

            stack.pop();

         }

         result[i] = stack.isEmpty() ? -1 : stack.peek();

         stack.push(arr[i]);

      }

      return result;

   }

   public static void main(String[] args) {
```

```
        int[] arr1 = {1, 3, 2, 4};

        System.out.println("Input: " + Arrays.toString(arr1));

        System.out.println("Output: " + Arrays.toString(findNextGreaterElements(arr1)));

    }

}
```

Time Complexity: O(n)

Output:

```
PROBLEMS 11   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\sde program> cd "C:\sde program\program"
PS C:\sde program\program> javac next_greater_element.java
PS C:\sde program\program> java next_greater_element
Input: [1, 3, 2, 4]
Output: [3, 4, 4, -1]
PS C:\sde program\program>
```