22IT023 DHONI R

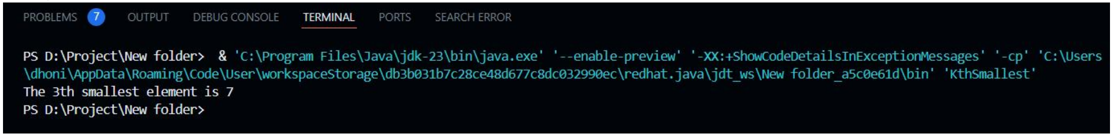
DSA PRACTICE SET 4

1) Kth smallest element

Code:

```
import java.util.PriorityQueue;
public class KthSmallest {
public static int kthSmallest(int[] arr,int k) {
PriorityQueue<Integer> minheap=new PriorityQueue<>();
for(int num:arr) {
minheap.add(num);
}
for(int i=0;i<k-1;i++) {
minheap.poll();
}
return minheap.poll();
}
public static void main(String[] args) {
int[] arr = {7, 10, 4, 3, 20, 15};
int k = 3;
System.out.println("The " + k + "th smallest element is " + kthSmallest(arr, k));
}
}
```

Output:



```
PS D:\Project\New folder>  & 'C:\Program Files\Java\jdk-23\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users
\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'KthSmallest'
The 3th smallest element is 7
PS D:\Project\New folder>
```

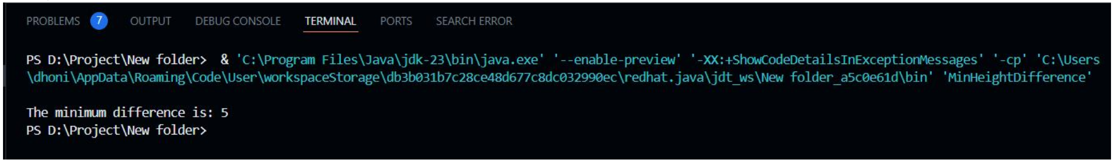
Time Complexity: O(Nlogn)

2) Minimize the height ll

Code:

```
import java.util.Arrays;
public class MinHeightDifference {
public static int getMinDiff(int[] arr, int k) {
int n = arr.length;
if (n == 1) {
return 0;
}
Arrays.sort(arr);
int mindiff = arr[n - 1] - arr[0];
for (int i = 1; i < n; i++) {
if (arr[i] - k < 0) continue;
int min = Math.min(arr[0] + k, arr[i] - k);
int max = Math.max(arr[i - 1] + k, arr[n - 1] - k);
mindiff = Math.min(mindiff, max - min);}
return mindiff;
}
public static void main(String[] args) {
int[] arr = {1, 5, 8, 10};
int k = 2;
System.out.println("The minimum difference is: " + getMinDiff(arr, k));
}
}
```

Output:



```
PS D:\Project\New folder>  & 'C:\Program Files\Java\jdk-23\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users
\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'MinHeightDifference'

The minimum difference is: 5
PS D:\Project\New folder>
```

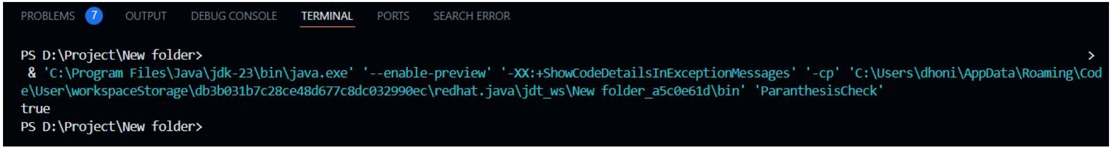
Time Complexity:O(Nlogn)

3) Parenthesis Checker

Code:

```
import java.util.Stack;class ParanthesisCheck{
static boolean isParenthesisBalanced(String s) {
Stack<Character> st=new Stack<>();
for(char it:s.toCharArray()){
if(it=='(' || it=='[' || it=='{'){
st.push(it);
}
else{
if(st.isEmpty()){
return false;
}
char ch=st.pop();
if(ch=='(' && it==')' ||
ch=='[' && it==']' ||
ch=='{' && it=='}'){
continue;
}
return false;
}
}
return st.isEmpty();
}
public static void main (String args[]) {
String s="(){}[]";
System.out.println(isParenthesisBalanced(s));
}
}
```

Output:



```
PS D:\Project\New folder>                                                                                                  >
 & 'C:\Program Files\Java\jdk-23\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Cod
e\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'ParanthesisCheck'
true
PS D:\Project\New folder>
```

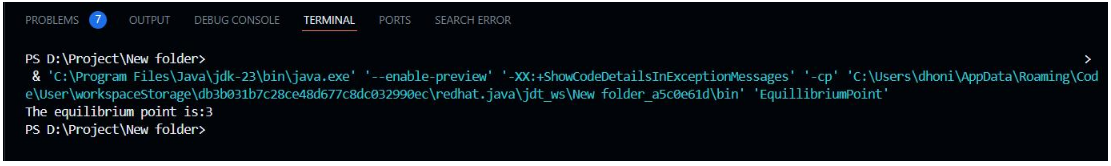
Time Complexity: O(N)

4) Equlibrium Point

Code:

```
class EquillibriumPoint {
public static int equilibriumPoint(int arr[]) {
int n=arr.length;
if(n==1){
return 1;
}
int totalsum=0;
for(int num:arr){
totalsum+=num;}
int left=0;
for(int i=0;i<n;i++){
int right=totalsum-left-arr[i];
if(left==right){
return i+1;
}
left+=arr[i];
}
return -1;
}
public static void main(String args[]) {
int[] arr= {1,3,5,2,2};
System.out.println("The equilibrium point is:"+equilibriumPoint(arr));
}
}
```

Output:



```
PS D:\Project\New folder>
 & 'C:\Program Files\Java\jdk-23\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Cod
e\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'EquillibriumPoint'
The equilibrium point is:3
PS D:\Project\New folder>
```

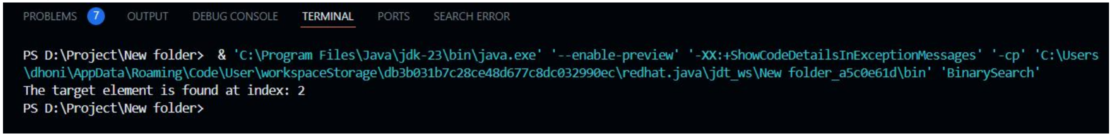
Time Complexity:O(N)

5) Binary Search

Code:

```
public class BinarySearch {
public static int binarySearch(int[] arr, int k) {
int left = 0;
int right = arr.length - 1;
while (left <= right) {
int mid = left + (right - left) / 2;
if (arr[mid] == k) {
return mid;
} else if (k < arr[mid]) {
right = mid - 1;
} else {
left = mid + 1;
}
}
return -1;
}
public static void main(String[] args) {
int[] arr = {2, 3, 5, 7, 8, 9};
int k = 5;
int result = binarySearch(arr, k);
if (result != -1) {
System.out.println("The target element is found at index: " + result);
} else {
System.out.println("The target element is not found in the array.");
}
}
}
```

Output:



```
PS D:\Project\New folder>  & 'C:\Program Files\Java\jdk-23\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users
\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'BinarySearch'
The target element is found at index: 2
PS D:\Project\New folder>
```

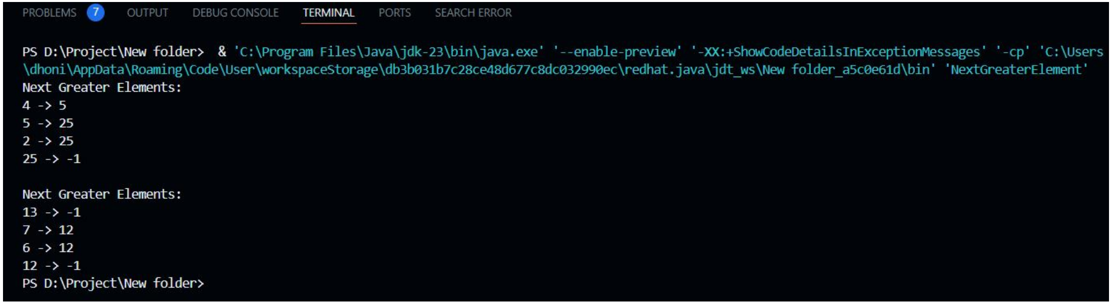
Time Complexity: O(logn)

6) Next Greater Element

Code:

```
import java.util.Scanner;
import java.util.Stack;
public class NextGreaterElement {
public static void nextGreater(int[] arr) {
Stack<Integer> stack=new Stack<>();
int[] nge=new int[arr.length];
for(int i=arr.length-1;i>=0;i--) {
while (!stack.isEmpty() && stack.peek() <= arr[i]) {
stack.pop();
}
nge[i] = stack.isEmpty() ? -1 : stack.peek();stack.push(arr[i]);
}
for (int i = 0; i < arr.length; i++) {
System.out.println(arr[i] + " --> " + nge[i]);
}
}
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Enter the elements of the array separated by spaces:");
String input = scanner.nextLine();
String[] elements = input.split(" ");
int[] arr = new int[elements.length];
for (int i = 0; i < elements.length; i++) {
arr[i] = Integer.parseInt(elements[i]);
}
System.out.println("Next Greater Elements:");
nextGreater(arr);
}
}
```

Output:



```
PS D:\Project\New folder>  & 'C:\Program Files\Java\jdk-23\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users
\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'NextGreaterElement'
Next Greater Elements:
4 -> 5
5 -> 25
2 -> 25
25 -> -1

Next Greater Elements:
13 -> -1
7 -> 12
6 -> 12
12 -> -1
PS D:\Project\New folder>
```

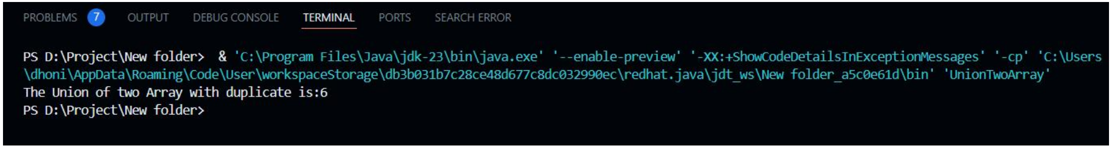
Time Complexity: O(N)

7) Union of Two Array with Duplicate

Code:

```
import java.util.HashSet;
public class UnionTwoArray {
public static int unionTwoArray(int[] a,int[] b) {
HashSet<Integer> hash=new HashSet<>();
for(int num:a) {
hash.add(num);
}
for(int num:b) {
hash.add(num);
}
return hash.size();
}public static void main(String[] args) {
int[] a= {1,2,3,4,5};
int[] b= {1,2,3,6};
System.out.println("The Union of two Array with duplicate is:"+unionTwoArray(a,b));
}
}
```

Output:



```
PS D:\Project\New folder>  & 'C:\Program Files\Java\jdk-23\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users
\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'UnionTwoArray'
The Union of two Array with duplicate is:6
PS D:\Project\New folder>
```

Time Complexity:O(N+M)