Name: Ramanarayanan k **13/11/2024**
22IT083

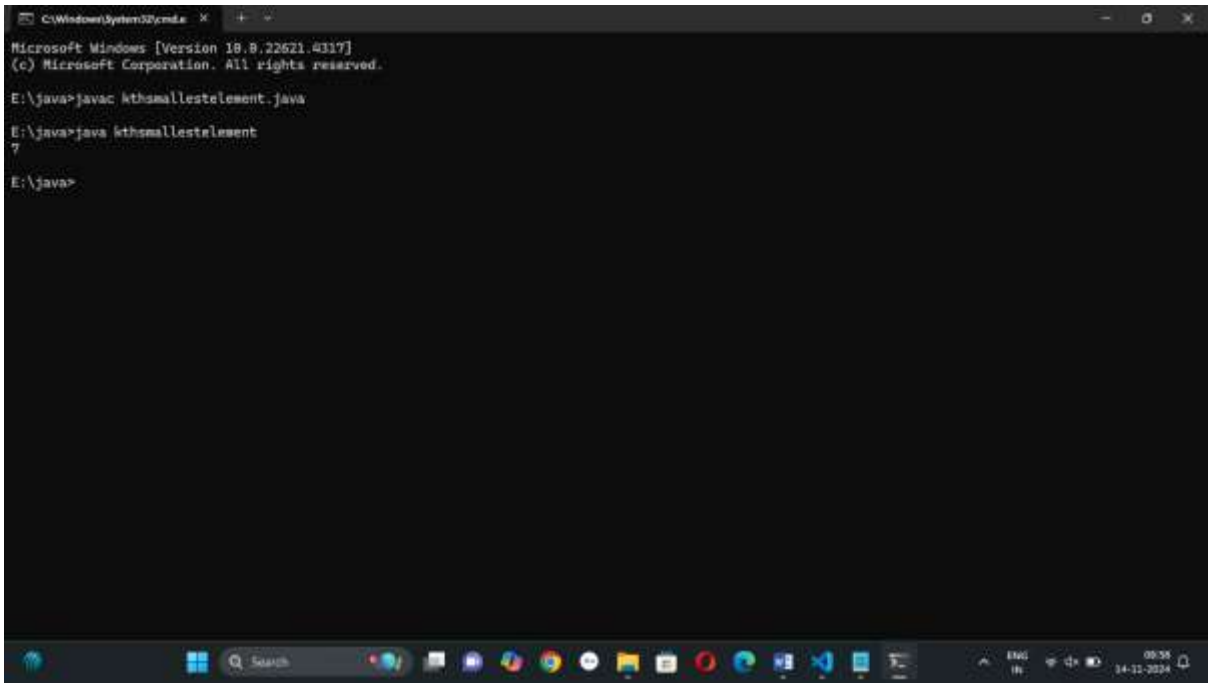
1 Kth smallest elements without sorting**:**

```
class kthsmallestelement{
    public static int kth(int num[], int k){
      //find the maxelement
      int n = num.length;
      int maxelement = num[0];
      for(int i=0;i<n;i++){
        if(num[i] > maxelement){
           maxelement = num[i];
        }
      }
      int[] freq = new int[maxelement+1];
      for(int i=0;i<n;i++){
        freq[num[i]]++;
      }

      int c=0;
      for(int i=0;i<maxelement;i++){
        if(freq[i] != 0){
           c+=freq[i];
           if(c >= k){
              return i;
           }
        }
```

```
        }
        return -1;
    }
    public  static void main(String[] arr){
        int num[] = {7, 10, 4, 3, 20, 15};
        int k = 3;
        System.out.println(kth(num, k));
    }
}
```

Output:

TC: O(nlogn)



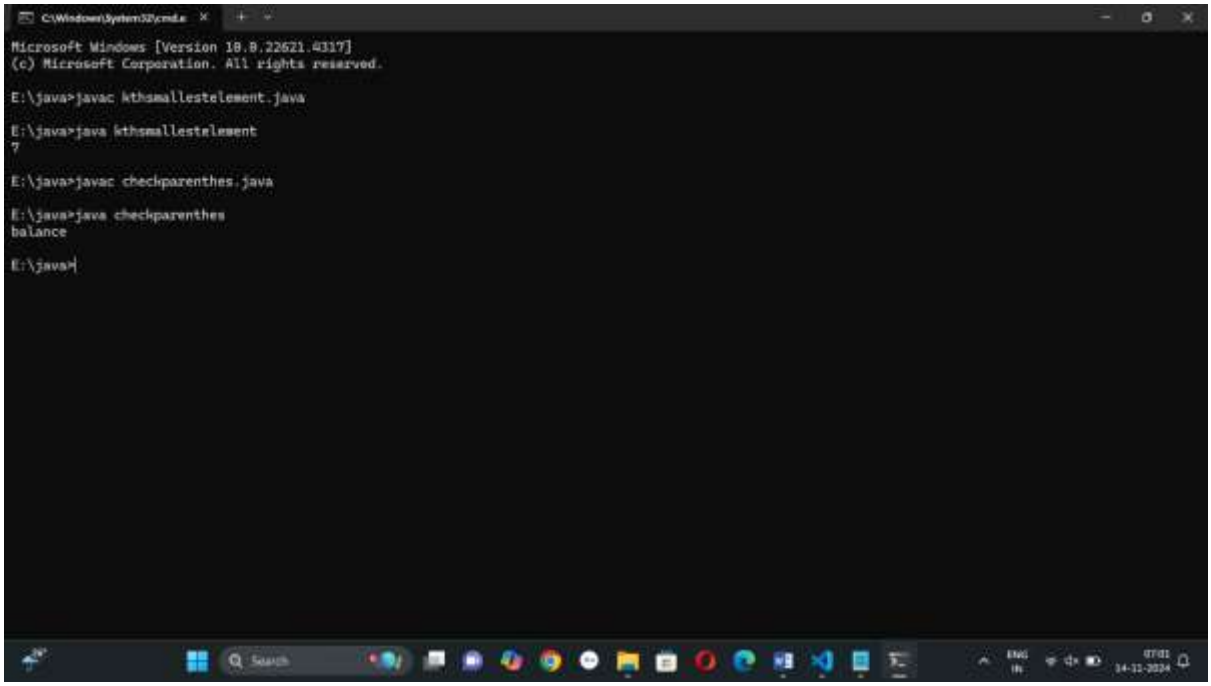
2 checkparenthes

Code:

```
import java.util.*;
class checkparenthes{
    boolean check(String s){
        Stack<Character> st = new Stack<>();
        for(char ch : s.toCharArray()){
            if(ch == '('){
                st.push(ch);
            }else if (ch == ')') {
                if (st.isEmpty() || st.pop() != '(') {
                    return false;
                }
            }
        }
        return st.isEmpty();
    }
   public static void main(String[] arg){
       String s = "()c";
       checkparenthes ch = new checkparenthes();
       if(ch.check(s)){
           System.out.println("balance");
       }else{
           System.out.println("not balance");
       }
   }
}
```

Output:

TC: O(n)



3) next Greater element

Code:

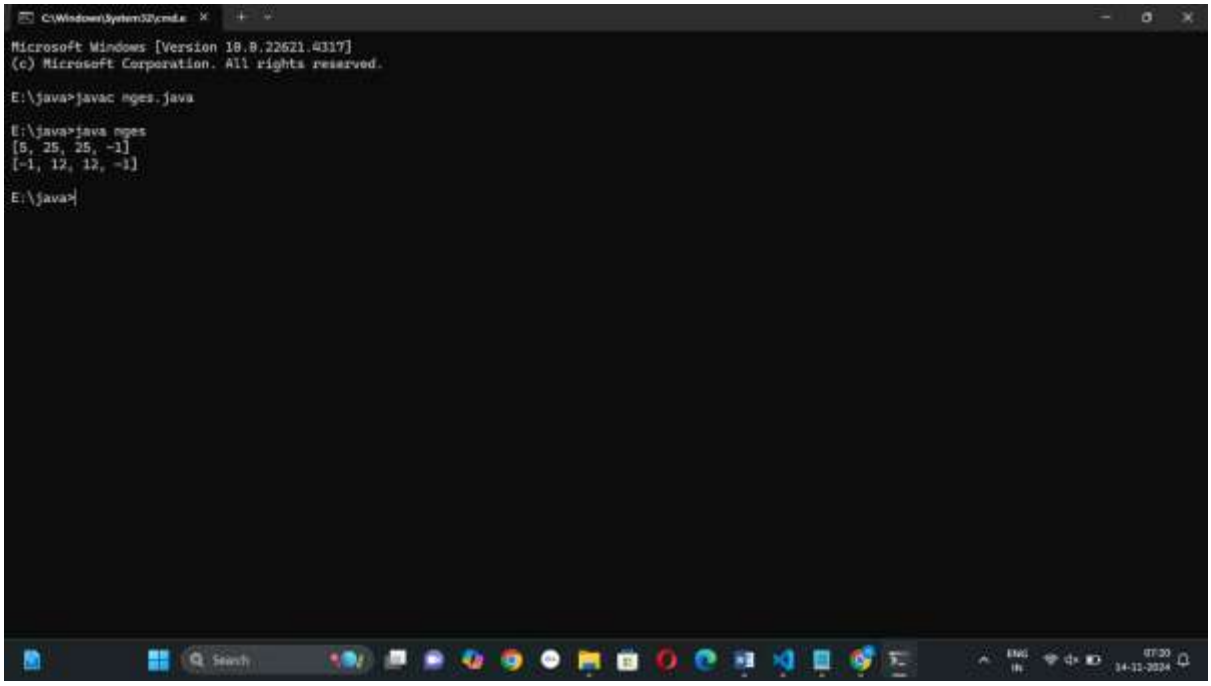
```
import java.util.ArrayList;
import java.util.Stack;

class nges{
    public ArrayList<Integer> nextLargerElement(int[] arr) {
        int n = arr.length;
        ArrayList<Integer> result = new ArrayList<>(n);
        Stack<Integer> stack = new Stack<>();
```

```
    for (int i = 0; i < n; i++) {
        result.add(-1);
    }

    for (int i = n - 1; i >= 0; i--) {
        while (!stack.isEmpty() && stack.peek() <= arr[i]) {
            stack.pop();
        }

        if (!stack.isEmpty()) {
            result.set(i, stack.peek());
        }

        stack.push(arr[i]);
    }

    return result;
}

public static void main(String[] args) {
    nges sol = new nges();

    int[] arr = {4, 5, 2, 25};
    System.out.println(sol.nextLargerElement(arr));
```

```
        int[] arr2 = {13, 7, 6, 12};
        System.out.println(sol.nextLargerElement(arr2));
    }
}
```

Output:

TC:O(n)



4) Binary Search

Code :

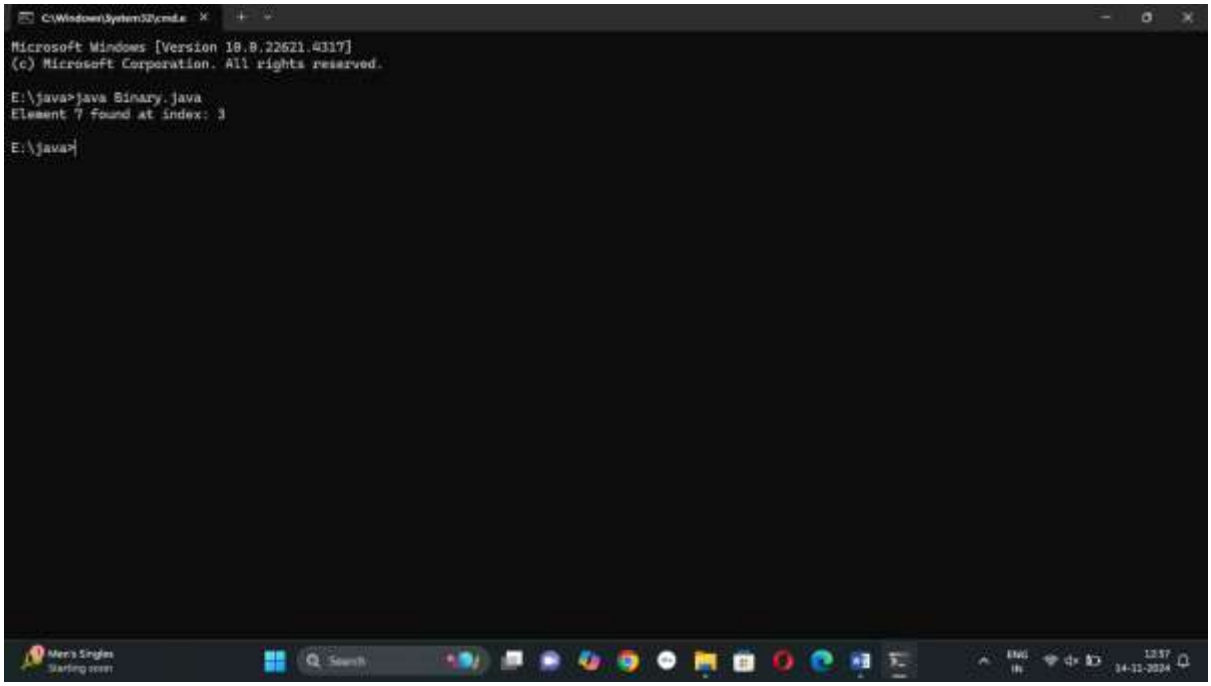
```
class Binary{

    public int binarysearch(int[] arr, int k) {
        int low = 0;
        int high = arr.length - 1;
```

```
    while (low <= high) {
        int mid = low + (high - low) / 2;
        if (arr[mid] == k) {
            return mid;
        } else if (arr[mid] > k) {
            high = mid - 1;
        } else {
            low = mid + 1;
        }
    }
    return -1;
}


public static void main(String[] args) {
        Binary solution = new Binary();

    int[] arr = {1, 3, 5, 7, 9, 11};
    int k = 7;


    int result = solution.binarysearch(arr, k);

    if (result != -1) {
        System.out.println("Element " + k + " found at index: " + result);
```

```
        } else {
            System.out.println("Element " + k + " not found in the array.");
        }
    }
}
```

Output:

TC: O(1)



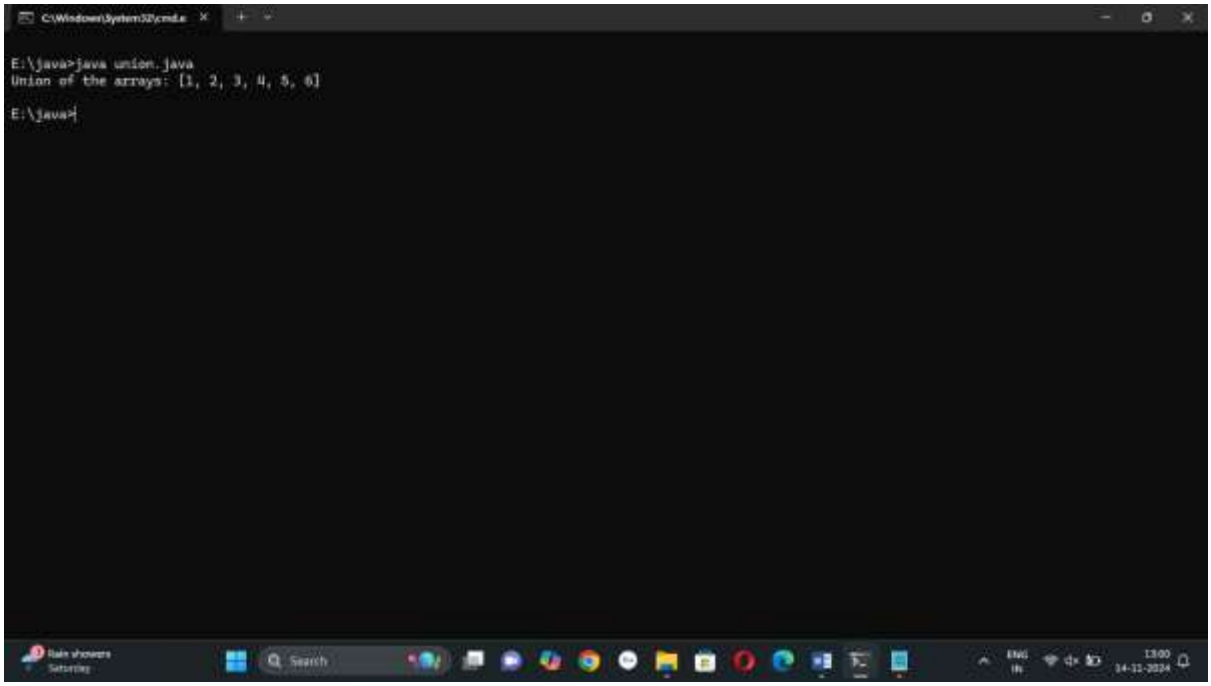
5) Find the union

```
import java.util.HashSet;
```

```
class union {
    public static HashSet<Integer> findUnion(int a[], int b[]) {
        HashSet<Integer> set = new HashSet<>();

        for (int i : a) {
            set.add(i);
        }

        for (int i : b) {
            set.add(i);
        }
        return set;
    }

    public static void main(String[] args) {
        int[] a = {1, 2, 3, 4};
        int[] b = {3, 4, 5, 6};

        HashSet<Integer> unionSet = findUnion(a, b);
        System.out.println("Union of the arrays: " + unionSet);
    }
}
```

Output:

TC: O(n+m)



6) EquilibriumPoint Point Found

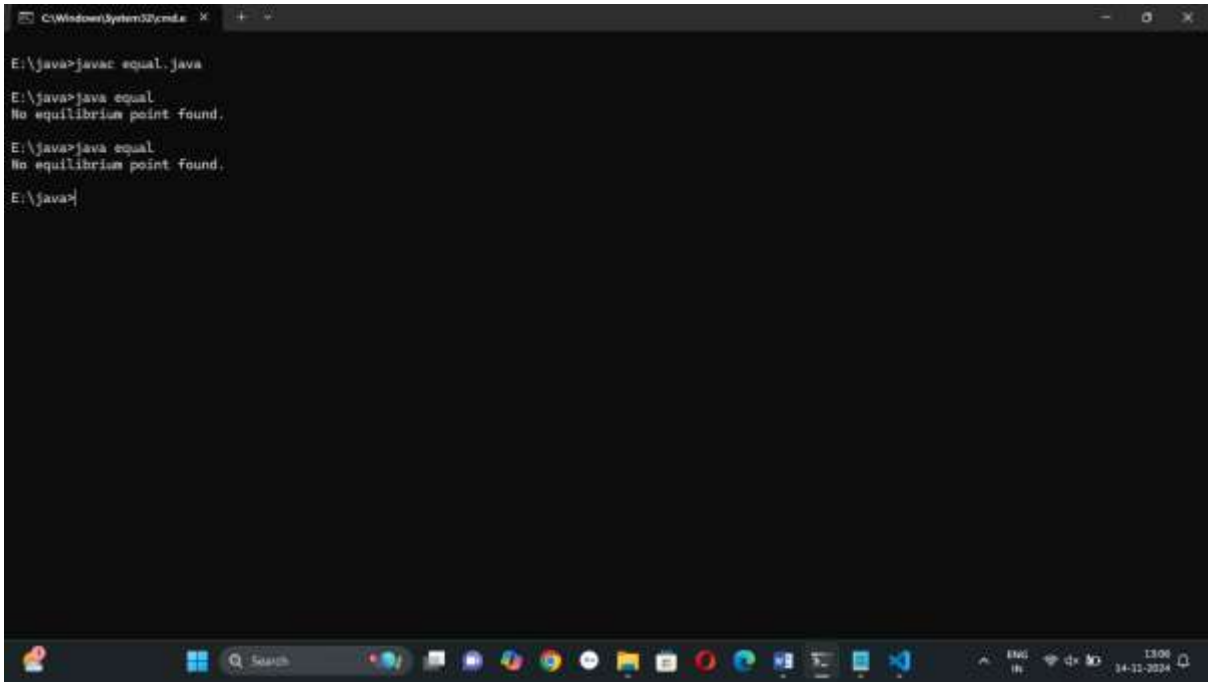
Code:

```
class equal {

    public static int equilibriumPoint(int[] arr) {
        int left = 0;
        int right = arr.length - 1;
        long leftSum = 0;
        long rightSum = 0;

        while (left < right) {
            if (leftSum < rightSum) {
                leftSum += arr[left];
```

```
            left++;
        } else {
            rightSum += arr[right];
            right--;
        }
    }

    return leftSum == rightSum ? left + 1 : -1;
  }



  public static void main(String[] args) {
    int[] arr = {3, 4, 8, 1, 20, 6};
    int result = equilibriumPoint(arr);


    if (result != -1) {
        System.out.println("Equilibrium point found at index (1-based): " +
result);
    } else {
        System.out.println("No equilibrium point found.");
    }
  }
}
```

Output:

TC:O(n)


C:\Windows\System32\cmd.e
E:\java>javac equal.java
E:\java>java equal
No equilibrium point found.
E:\java>java equal
No equilibrium point found.
E:\java>
Search
ENG
IN
13:06
14-11-2024