class A extends Object

{

//code here

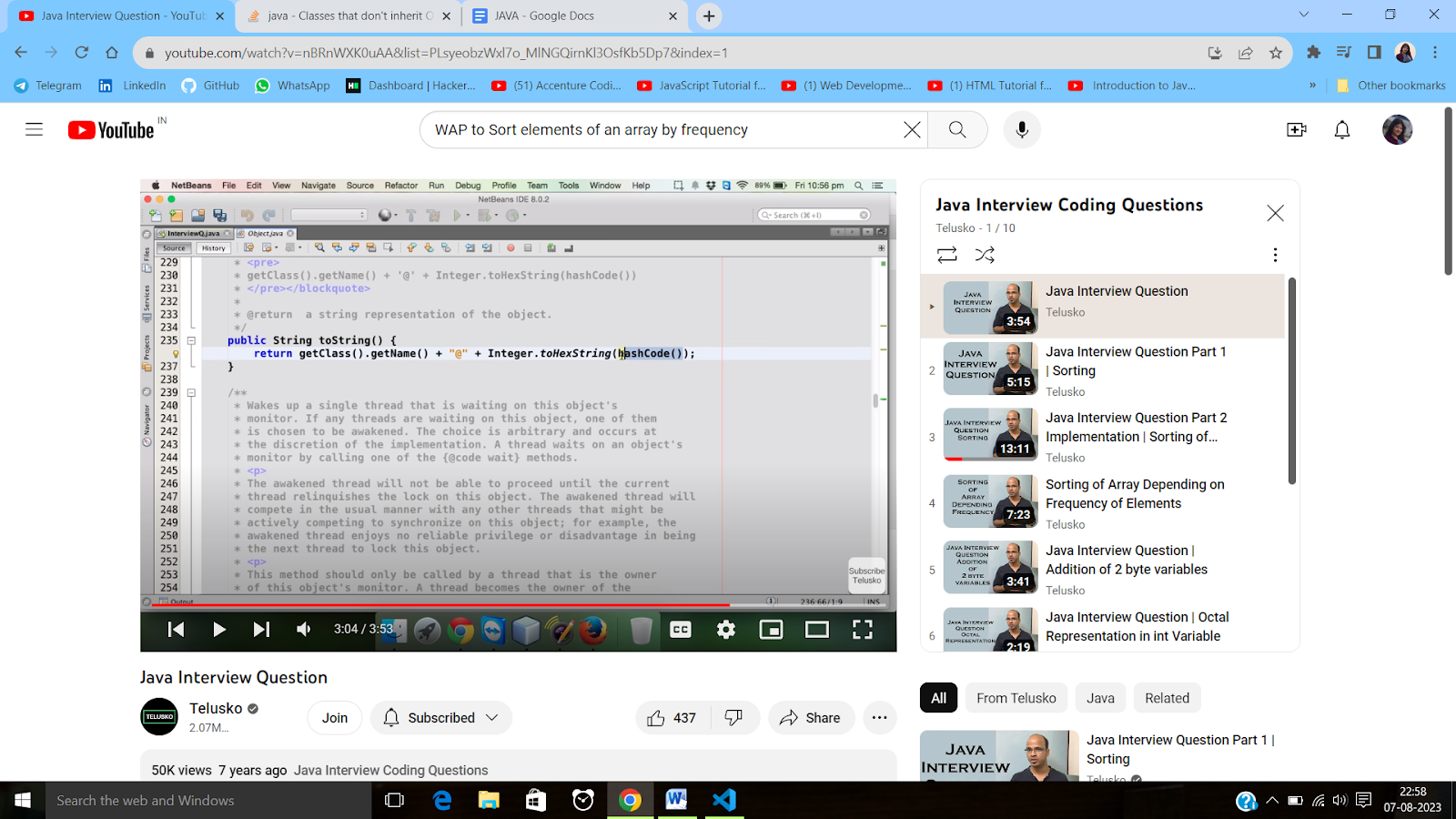
}

Java classes by default extend the Object class. All classes implicitly extend the Object class from java.lang.Object if they don’t extend any other super class.

 A obj = new A();

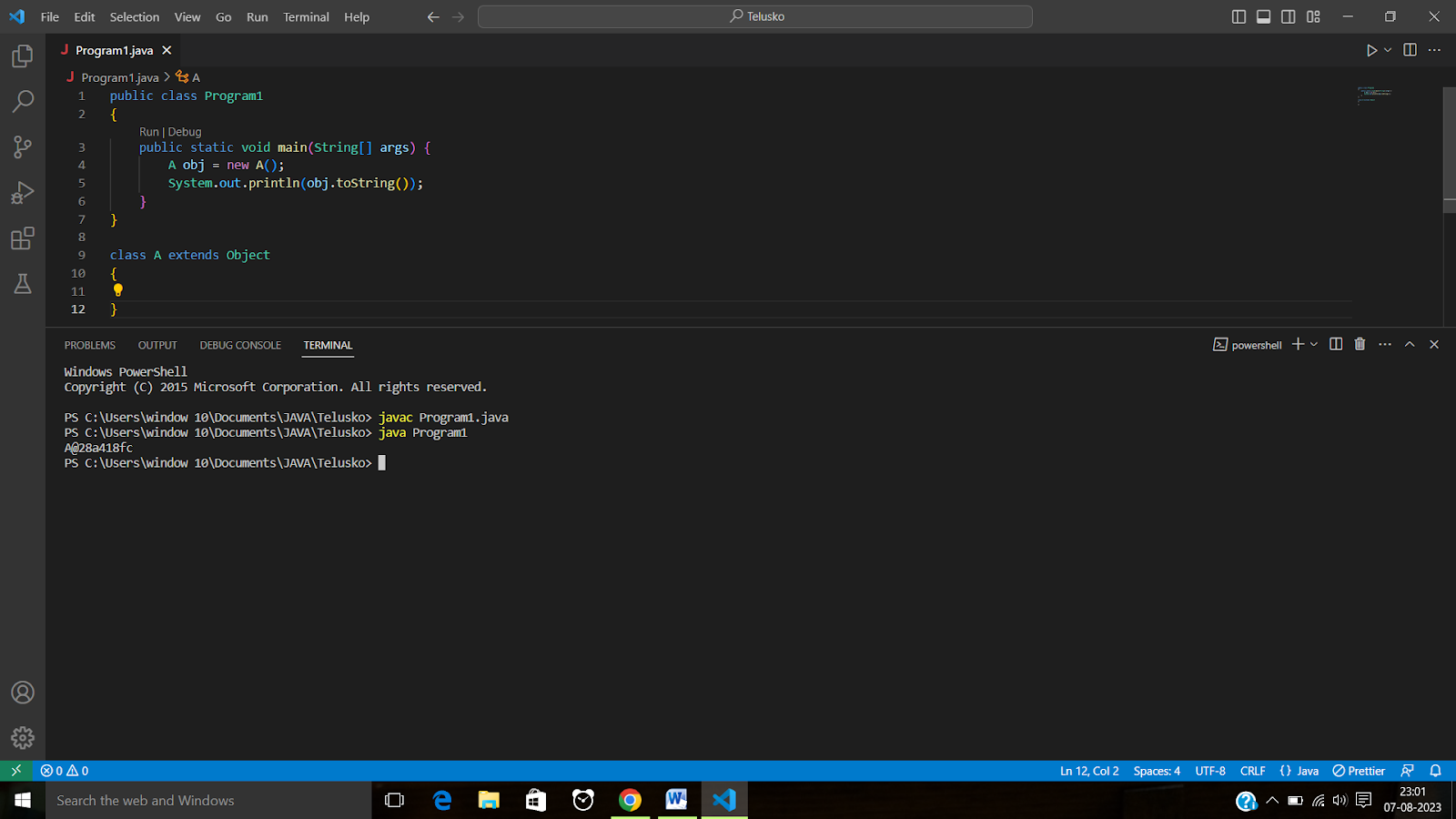
  System.out.println(obj.toString());

By default, the toString() method is called whenever an object of a class is to be printed even if we don’t explicitly mention it /even if there is no method called toString() in class A.



Here, A [A@28a418fc ] is the className, " @” is the separator and the remaining is the hash code.

Generally Hashcodes are in binary form so “ .tohexString() “ is applied to get the hexadecimal form.



Question :-

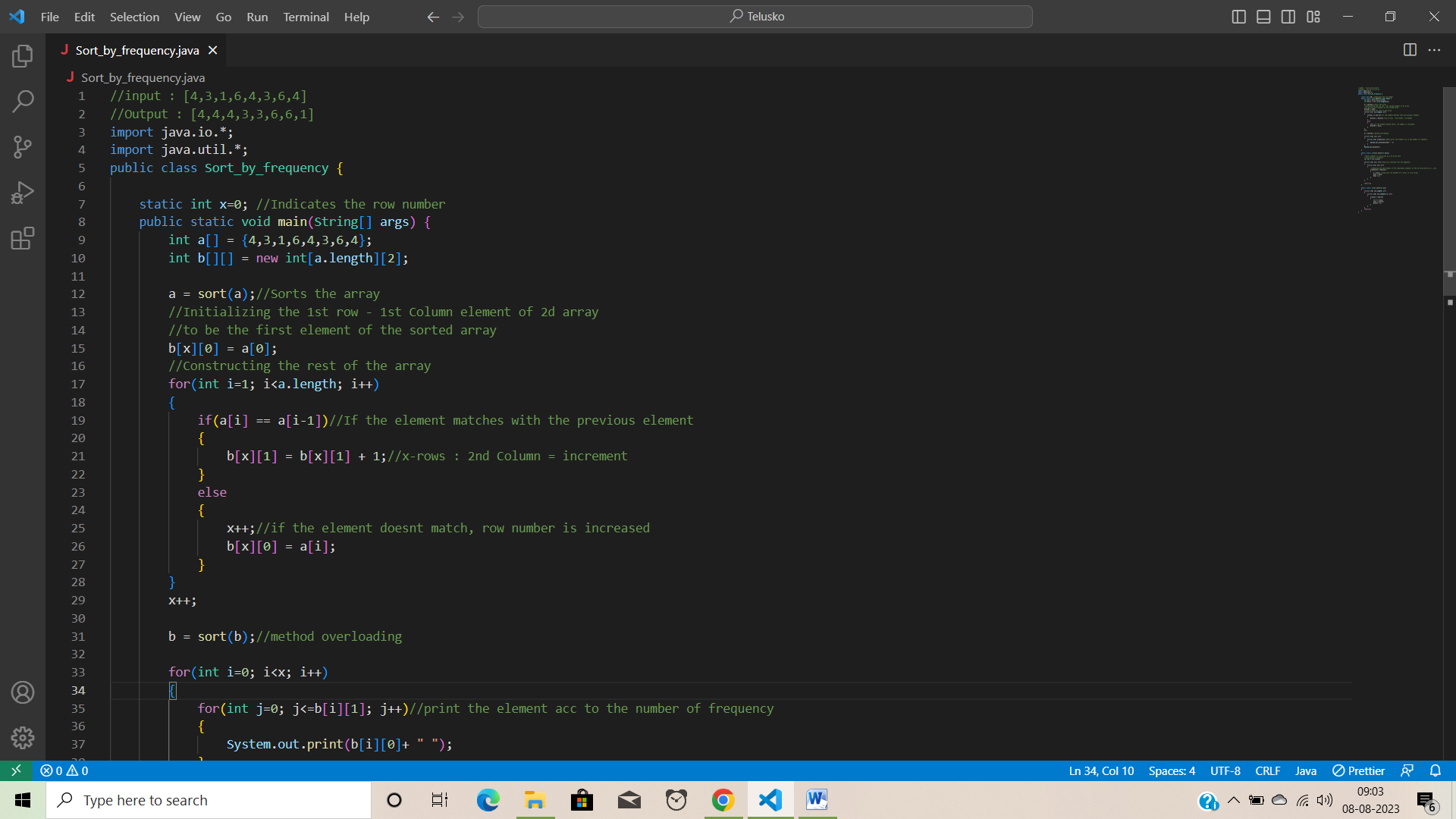
You are given an array of length n

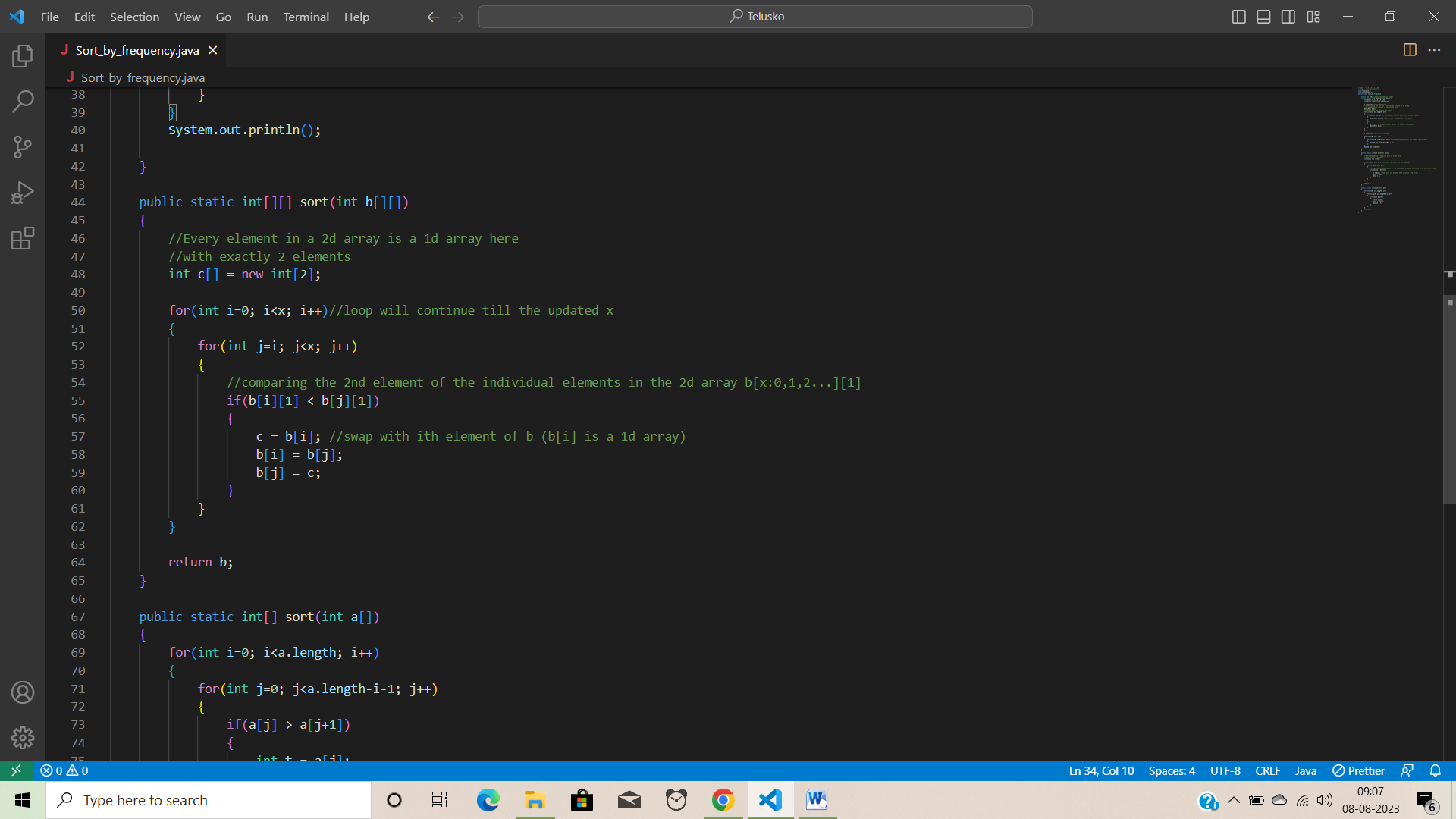
You have to sort the array elements in descending order of their frequency.

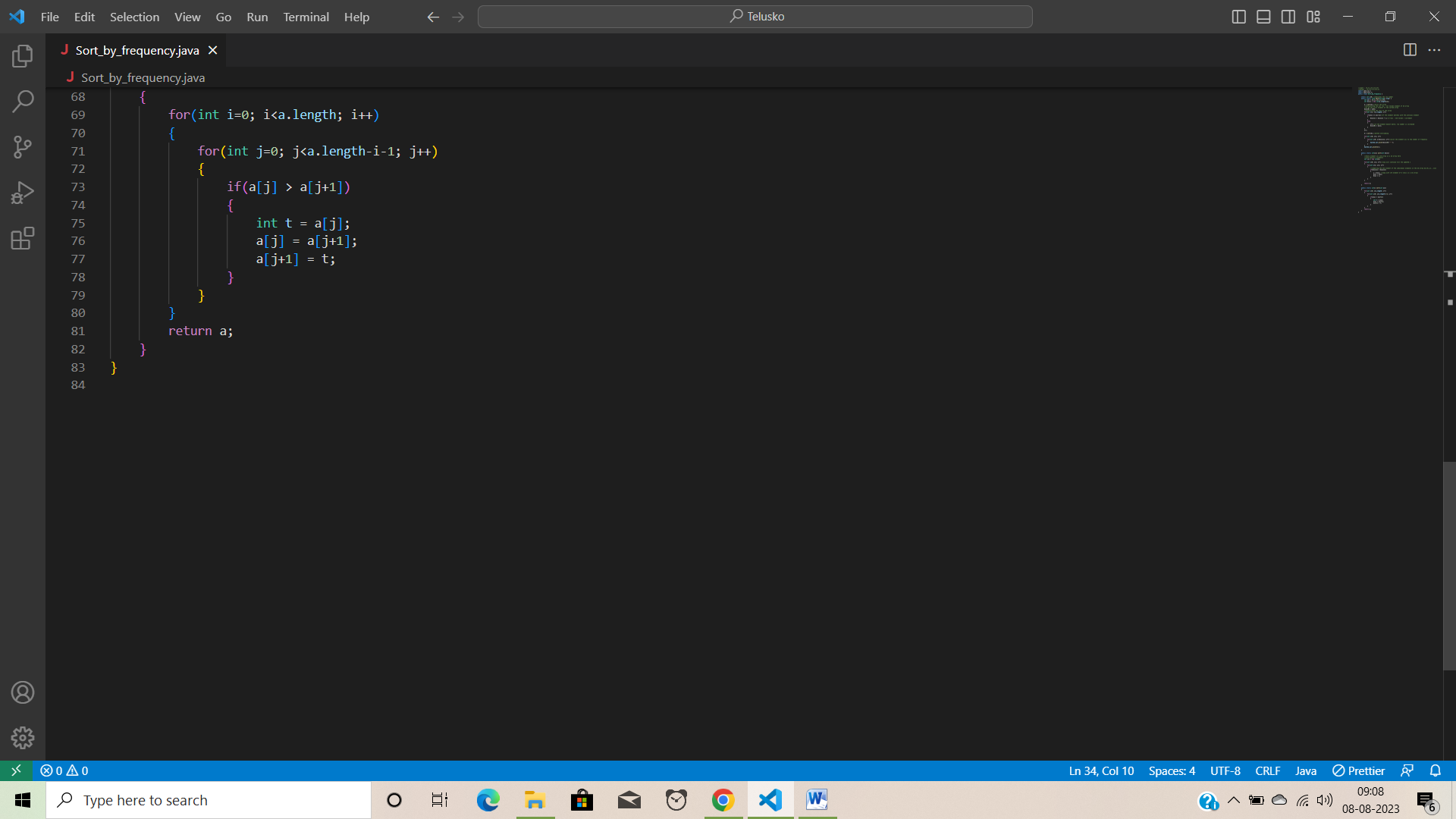
Use of collections is not allowed.

Solution pathway:

1. Sort the array
2. Create a 2D array (1st column: distinct elements, 2nd column: frequency)
3. Sort the 2d array acc. to 2nd column
4. Print the array.



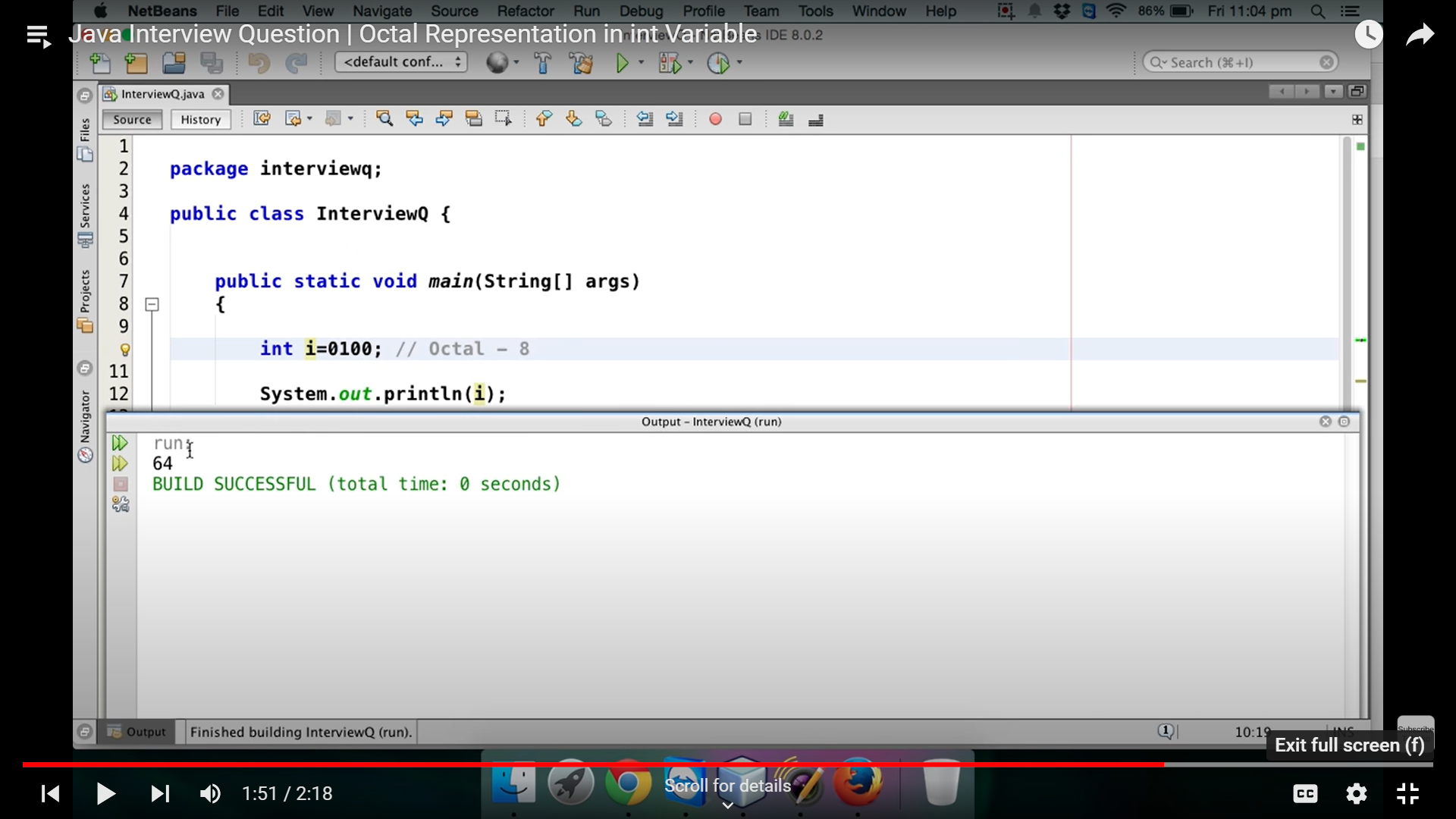




* “ + “ Arithmetic operator results in a integer output or output of higher order. When two byte variables are added ,the result is “ Integer “ and not Byte.

* Whenever, there is a 0 in front of a number, that number is considered as Octal in java.

Hence, in the below example int i = 0100 results in 64



**Matrix Exponentiation:**

1-> Matrix multiplication

2-> Binary exponentiation

Program in java to find out Nth Fibonacci number in best time complexity

* <https://youtu.be/8hDbRQffpmE>

Given an integer ‘N’, return all the divisors of ‘N’ in ascending order.

* import java.util.\*;

public class Solution{

     public static List< Integer > printDivisors(int n)

{

        int i=1;

         List<Integer> arr = new ArrayList<Integer>(4);

         for(i=1; i<=n; i++)

         {

             if(n%i==0)

             {

                 arr.add(i);

             }

         }

         return arr;

     }

}

There are two ways for requesting the JVM to run the garbage collector.

* Using System.gc() method
* Using Runtime.getRuntime().gc() method

Ways to make an object eligible for the garbage collector:

* Nullifying the reference variable
* By assigning a reference variable to another
* By anonymous object.

Given an arrayList ‘ARR’ of integers and a position ‘M’. You have to reverse the array after that position.

Example:

6 3

Input: 1 2 3 4 5 6

Output: 1 2 3 4 6 5

public static void reverseArray (ArrayList<integer> arr, int m)

{

Collections.reverse (arr.subList (m+1, arr.size()));

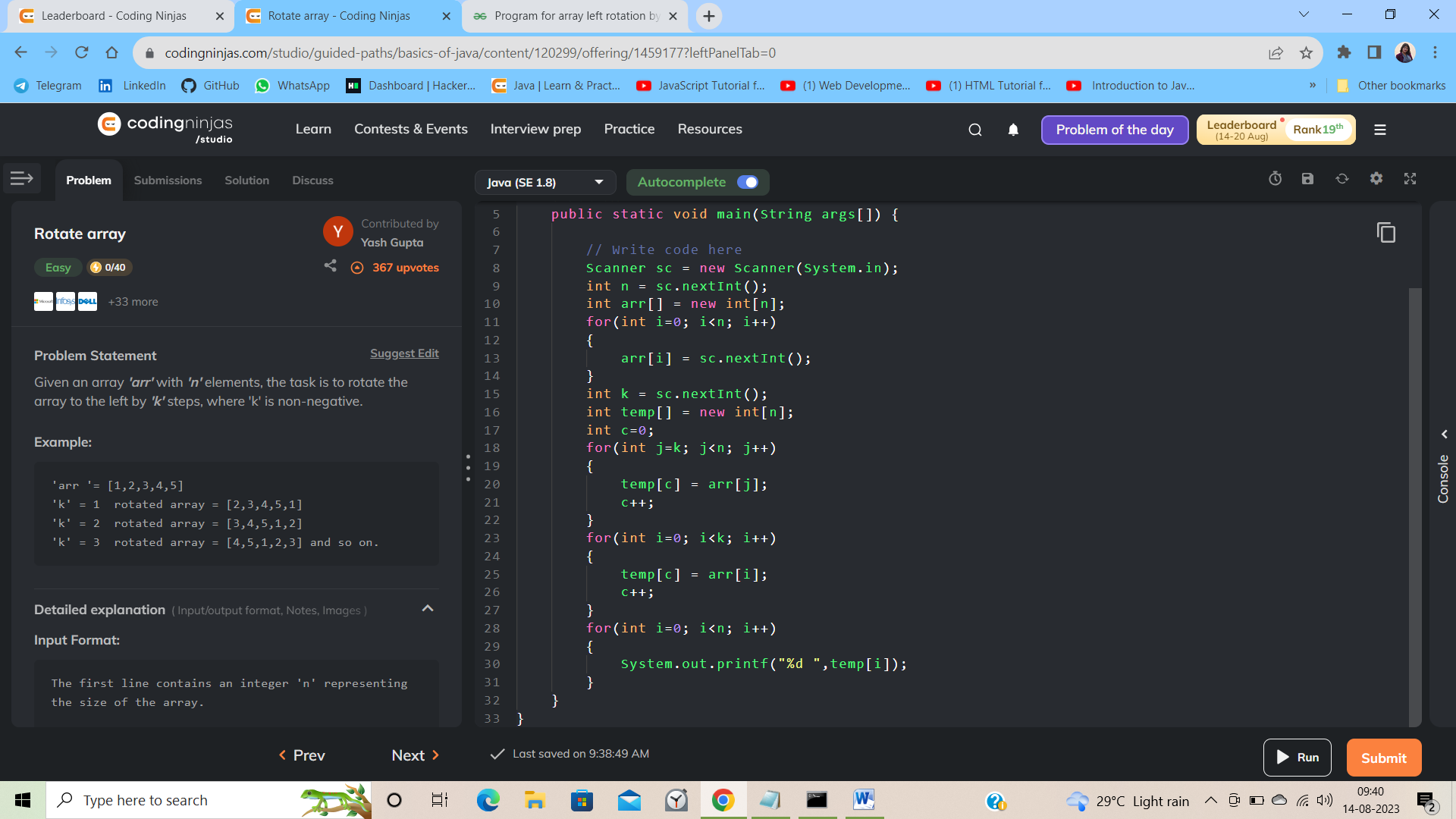
}

Given an array ‘arr’ with ‘n’ elements , the task is to rotate the array to the left by ‘k’ steps, where ‘k’ is non negative.

Arr[] = {1,2,3,4,5,6,7}

D = 2

Output : 3 4 5 6 7 1 2



1. Print first ‘N’ natural numbers in reverse order

for(i=N; i>=0; i--){

System.out.println(i);

}

1. Prime number checking

Approach 1: Basic hit and trial method

Count =0;

for(i=0; i<n+1; i++){

If(n%i==0){

Count+=1;

}}

If(n<2 || count>2){

System.out.println(“Not prime”);

}

Else{

System.out.println(“Prime”);

}

Approach 2: Reducing the number of iterations

If(n<2){

isPrime = false;

}

Else{

for(i=2;i<n;i++){

If(n%i==0)

{

isPrime = false;

break;

}

}

}

Approach 3: Reducing the number of iterations to half because no number can have factors greater that its half.

If(n<2){

isPrime = false;

}

Else{

for(i=2;i<n/2;i++){

If(n%i==0)

{

isPrime = false;

break;

}

}

}

Approach 4: If a number is not prime then in any factorization of N, atleast one of the factors must be smaller than the square root of N, that’s why;

If(n<2){

isPrime = false;

}

Else{

for(i=2;i<Math.sqrt(n);i++){

If(n%i==0)

{

isPrime = false;

break;

}

}

}

Approach 5: Excluding the even numbers

If(n <= 1){

Return false;

}

Else if(n==2){ //2 is the only prime number

Return true;

}

Else if(n%2 == 0){

Return false;

}

For (int i=3; i<=Math.sqrt(n); i+=2){

If(n%i==0){

Return false;

}

}

Return true;

}