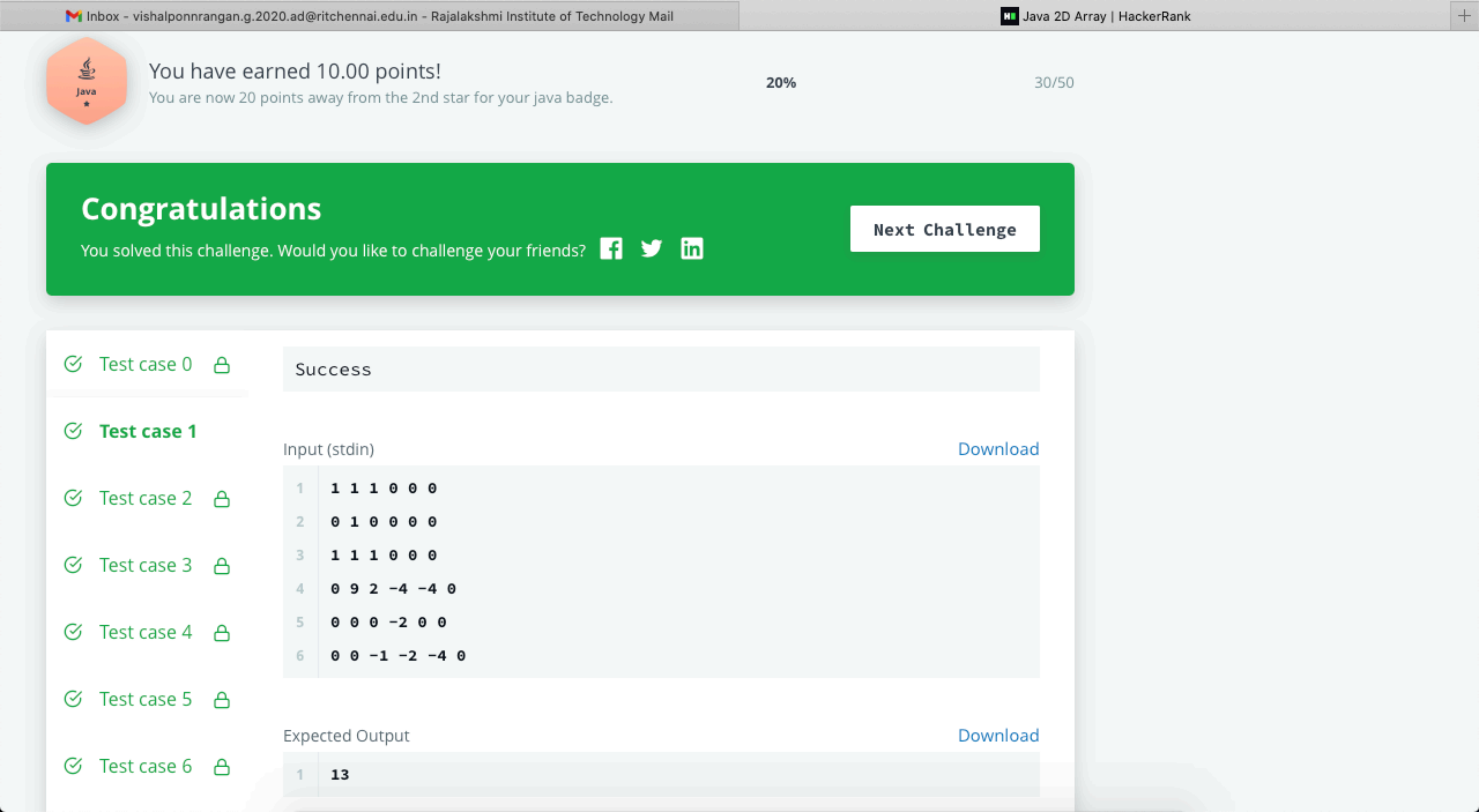


Inbox - vishalponnrangan.g.2020.ad@ritchennai.edu.in - Rajalakshmi Institute of Technology Mail

Java 2D Array | HackerRank

You have earned 10.00 points!

You are now 20 points away from the 2nd star for your java badge.

20%

30/50

Congratulations

You solved this challenge. Would you like to challenge your friends?

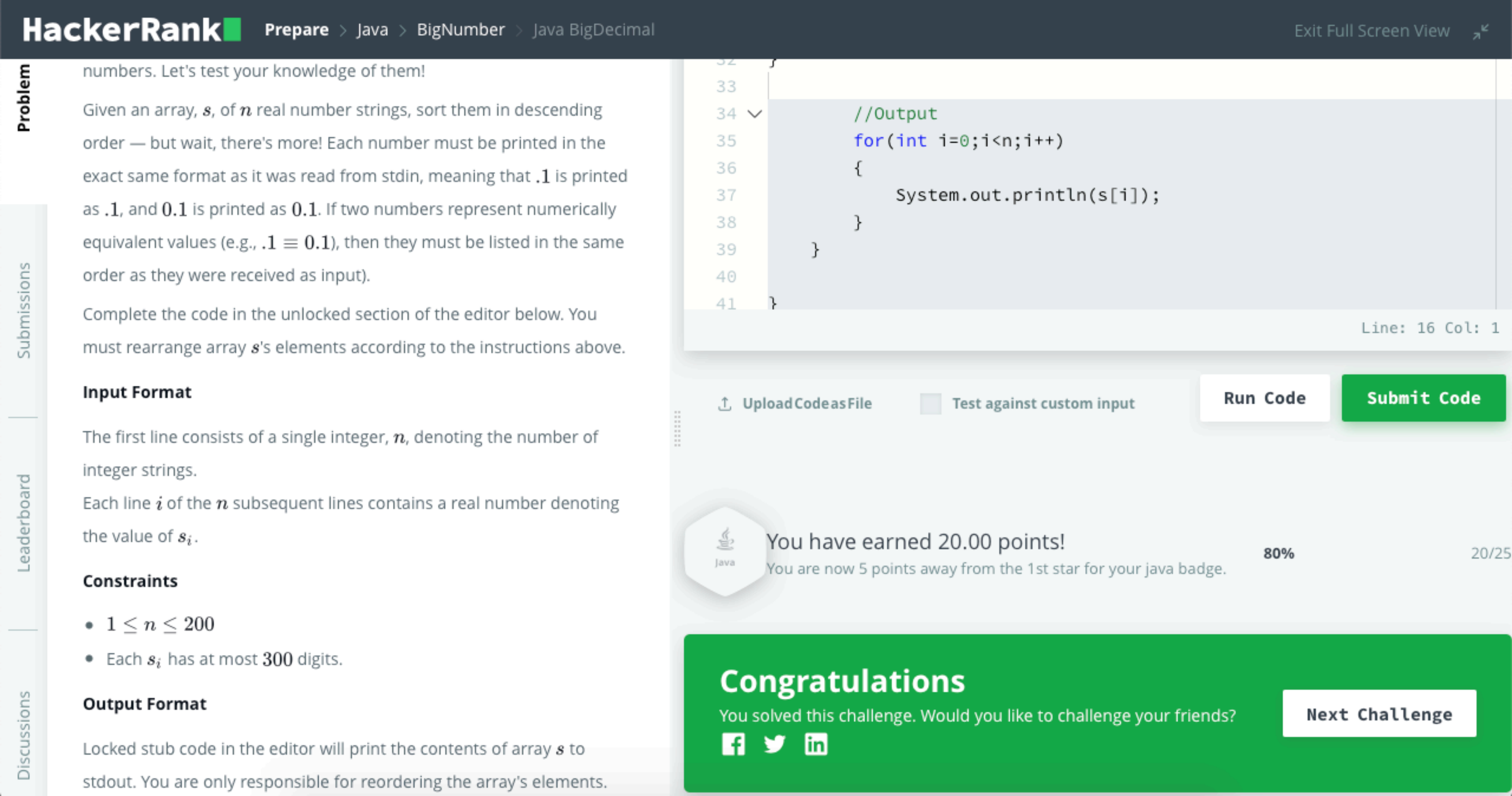
Success

Input (stdin) Download

```
1 1 1 0 0 0
0 1 0 0 0 0
1 1 1 0 0 0
0 9 2 -4 -4 0
0 0 0 -2 0 0
0 0 -1 -2 -4 0
```

Expected Output Download

```
13
```



numbers. Let's test your knowledge of them!

Given an array, s, of n real number strings, sort them in descending order — but wait, there's more! Each number must be printed in the exact same format as it was read from stdin, meaning that $.1$ is printed as $.1$, and 0.1 is printed as 0.1. If two numbers represent numerically equivalent values (e.g., $.1 \equiv 0.1$), then they must be listed in the same order as they were received as input).

Complete the code in the unlocked section of the editor below. You must rearrange array s's elements according to the instructions above.

Input Format

The first line consists of a single integer, n, denoting the number of integer strings.

Each line i of the n subsequent lines contains a real number denoting the value of s_i.

Constraints

- $1 \leq n \leq 200$
- Each s_i has at most 300 digits.

Output Format

Locked stub code in the editor will print the contents of array s to stdout. You are only responsible for reordering the array's elements.

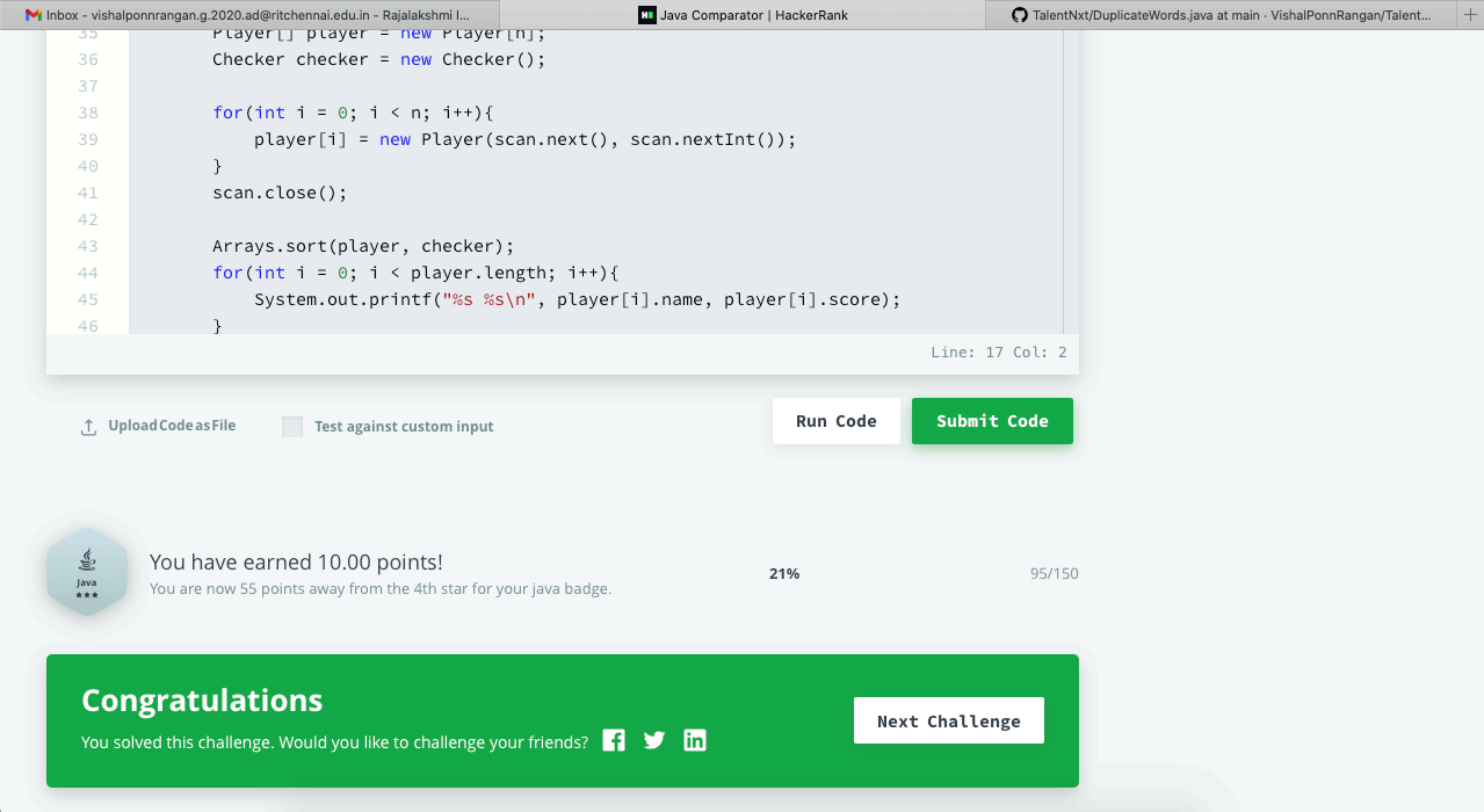
```
        }

            //Output
            for(int i=0;i<n;i++)
            {
                System.out.println(s[i]);
            }
        }

}
```

You have earned 20.00 points!

You are now 5 points away from the 1st star for your java badge.

Congratulations

You solved this challenge. Would you like to challenge your friends?


Inbox - vishalponnrangan.g.2020.ad@ritchennai.edu.in - Rajalakshmi I...
Java Comparator | HackerRank
TalentNxt/DuplicateWords.java at main · VishalPonnRangan/Talent...

```
        Player[] player = new Player[n];
        Checker checker = new Checker();

        for(int i = 0; i < n; i++){
            player[i] = new Player(scan.next(), scan.nextInt());
        }
        scan.close();

        Arrays.sort(player, checker);
        for(int i = 0; i < player.length; i++){
            System.out.printf("%s %s\n", player[i].name, player[i].score);
        }
```

Line: 17 Col: 2

Upload Code as File
□ Test against custom input
Run Code
Submit Code
Java
★★★

You have earned 10.00 points!

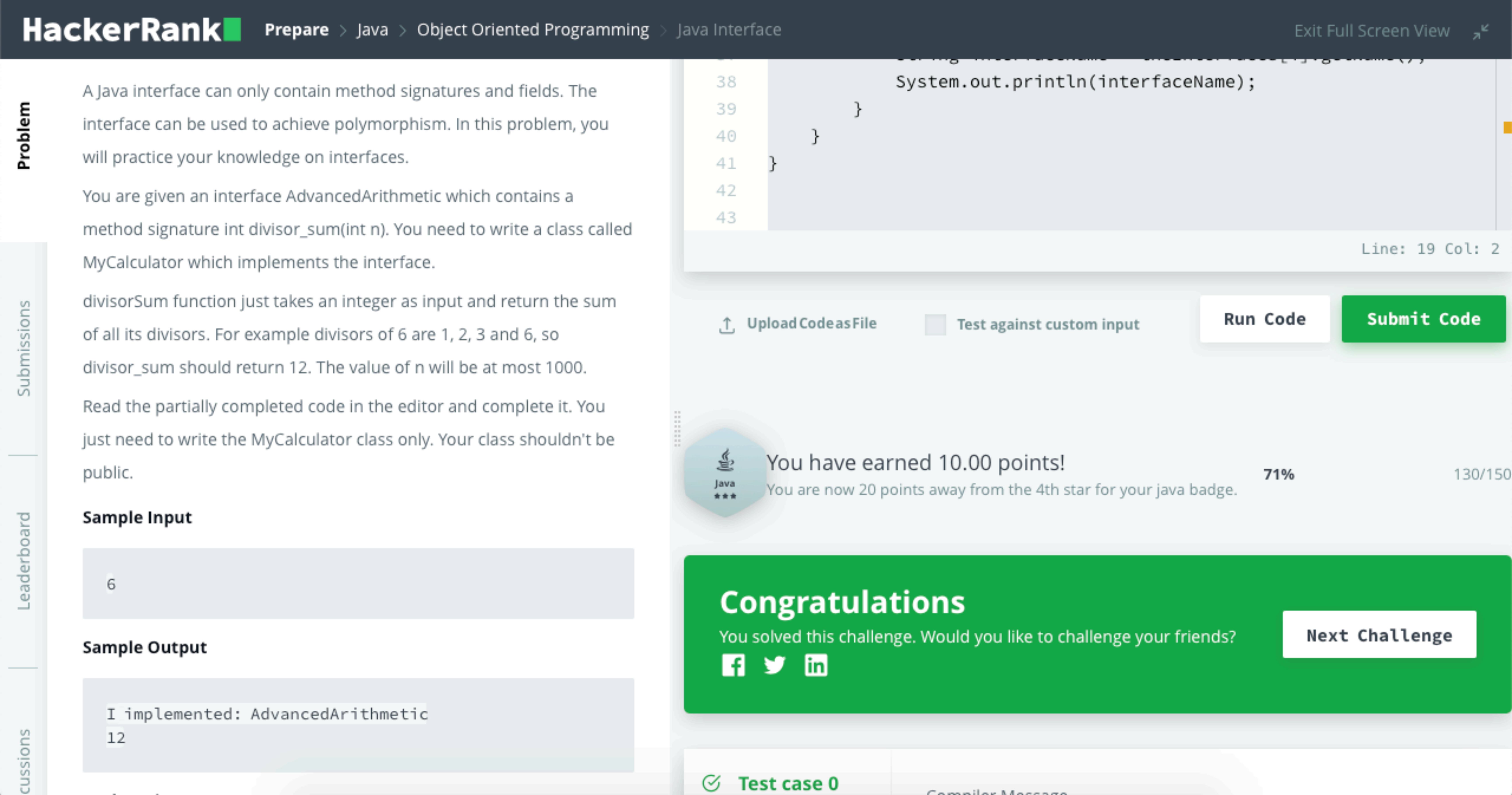
You are now 55 points away from the 4th star for your java badge.

21%

95/150

Congratulations

You solved this challenge. Would you like to challenge your friends?



A Java interface can only contain method signatures and fields. The interface can be used to achieve polymorphism. In this problem, you will practice your knowledge on interfaces.

You are given an interface AdvancedArithmetic which contains a method signature int divisor_sum(int n). You need to write a class called MyCalculator which implements the interface.

divisorSum function just takes an integer as input and return the sum of all its divisors. For example divisors of 6 are 1, 2, 3 and 6, so divisor_sum should return 12. The value of n will be at most 1000.

Read the partially completed code in the editor and complete it. You just need to write the MyCalculator class only. Your class shouldn't be public.

Sample Input

```
6
```

Sample Output

```
I implemented: AdvancedArithmetic
12
```

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

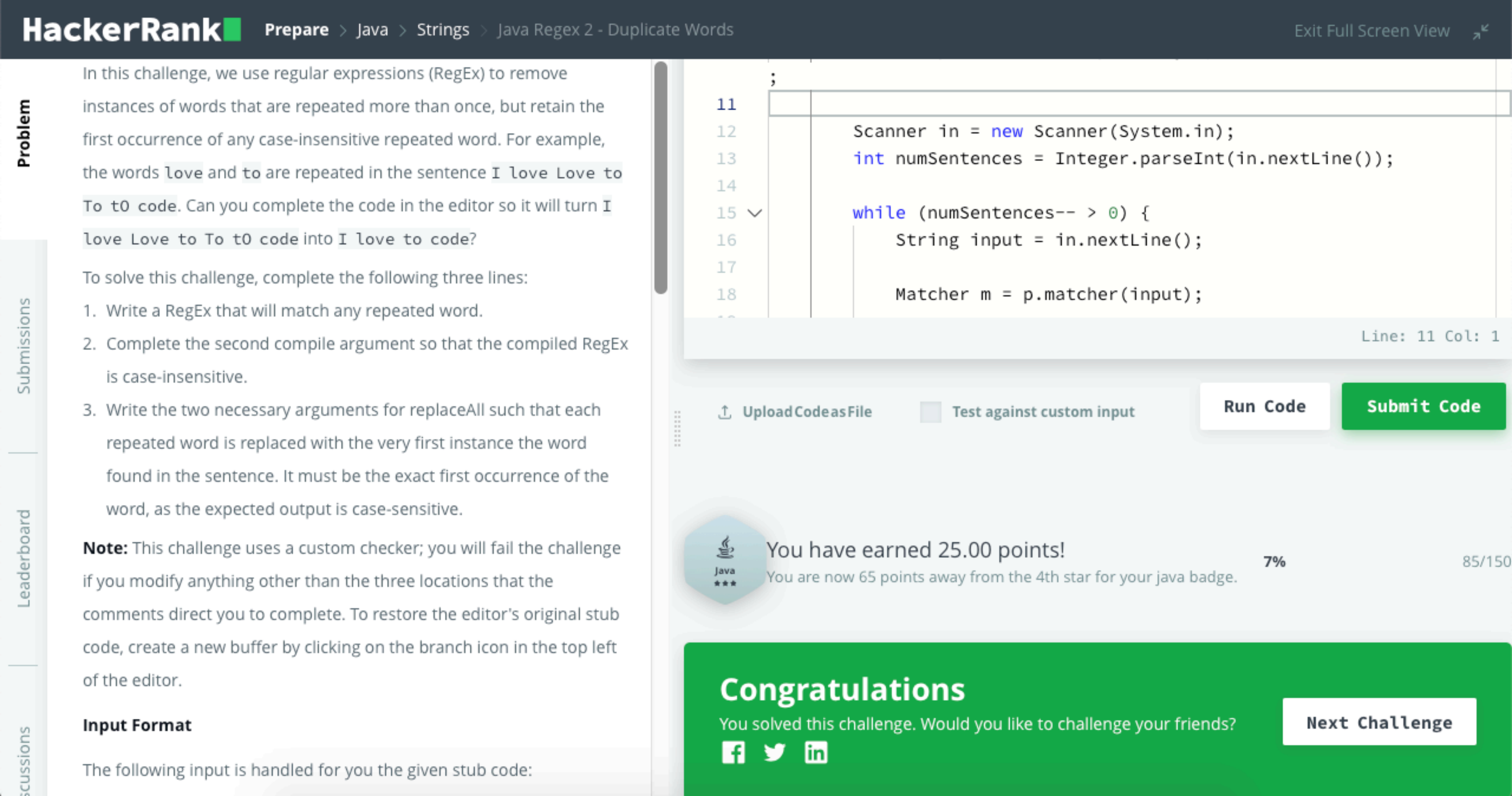
You have earned 10.00 points!

You are now 20 points away from the 4th star for your java badge.

Congratulations

You solved this challenge. Would you like to challenge your friends?

✓ Test case 0



In this challenge, we use regular expressions (RegEx) to remove instances of words that are repeated more than once, but retain the first occurrence of any case-insensitive repeated word. For example, the words `love` and `to` are repeated in the sentence `I love Love to To tO code`. Can you complete the code in the editor so it will turn `I love Love to To tO code` into `I love to code`?

To solve this challenge, complete the following three lines:

1. Write a RegEx that will match any repeated word.
2. Complete the second compile argument so that the compiled RegEx is case-insensitive.
3. Write the two necessary arguments for replaceAll such that each repeated word is replaced with the very first instance the word found in the sentence. It must be the exact first occurrence of the word, as the expected output is case-sensitive.

Note: This challenge uses a custom checker; you will fail the challenge if you modify anything other than the three locations that the comments direct you to complete. To restore the editor's original stub code, create a new buffer by clicking on the branch icon in the top left of the editor.

Input Format

The following input is handled for you the given stub code:

```
;

    Scanner in = new Scanner(System.in);
    int numSentences = Integer.parseInt(in.nextLine());

    while (numSentences-- > 0) {
        String input = in.nextLine();

        Matcher m = p.matcher(input);
```

Line: 11 Col: 1

You have earned 25.00 points!

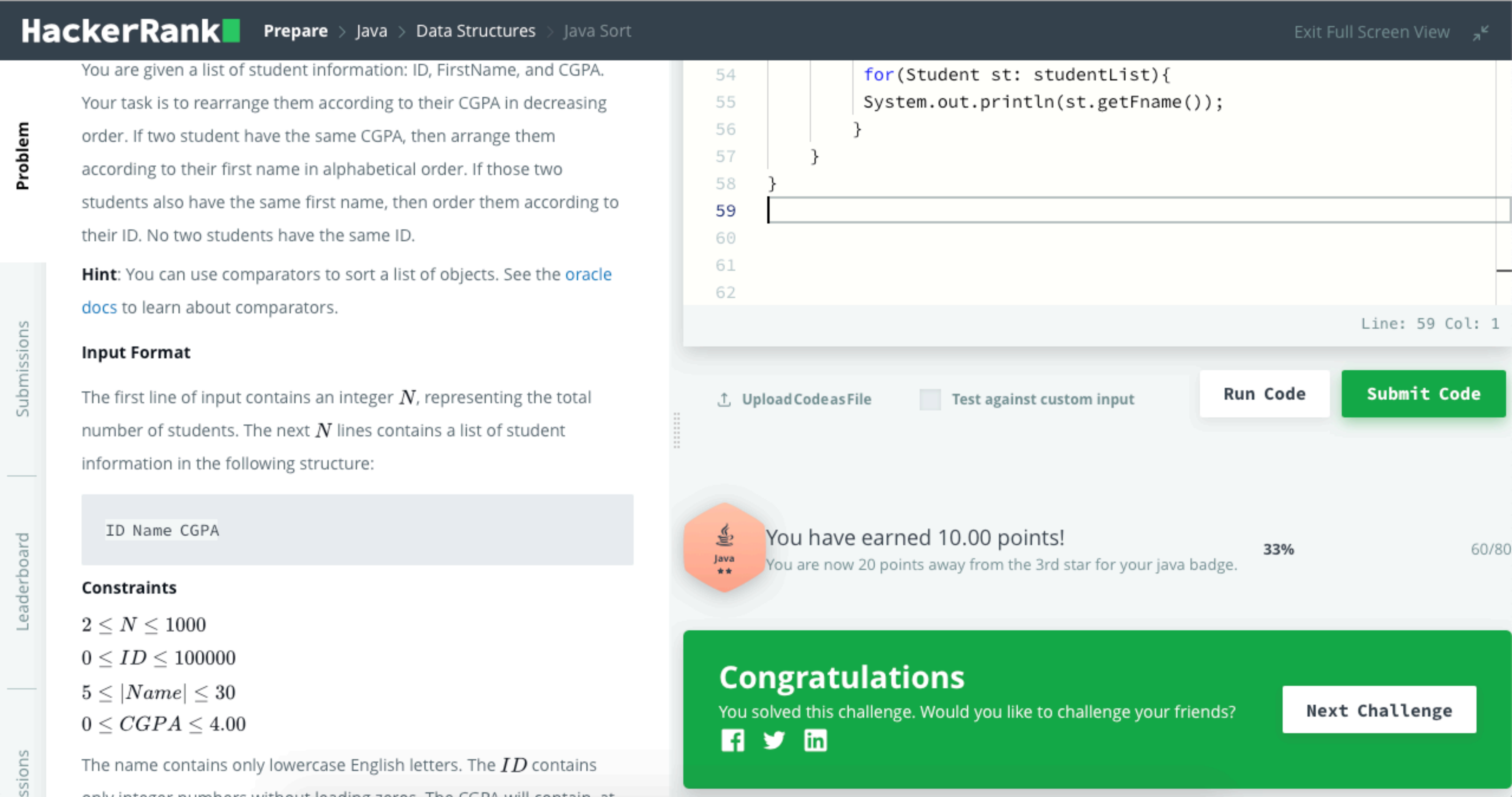
You are now 65 points away from the 4th star for your java badge.

7%

85/150

Congratulations

You solved this challenge. Would you like to challenge your friends?



You are given a list of student information: ID, FirstName, and CGPA. Your task is to rearrange them according to their CGPA in decreasing order. If two student have the same CGPA, then arrange them according to their first name in alphabetical order. If those two students also have the same first name, then order them according to their ID. No two students have the same ID.

Hint: You can use comparators to sort a list of objects. See the oracle docs to learn about comparators.

Input Format

The first line of input contains an integer N, representing the total number of students. The next N lines contains a list of student information in the following structure:

```
ID Name CGPA
```

Constraints

$2 \leq N \leq 1000$

$0 \leq ID \leq 100000$

$5 \leq |Name| \leq 30$

$0 \leq CGPA \leq 4.00$

The name contains only lowercase English letters. The ID contains only integer numbers without leading zeros. The CGPA will contain, at

```
            for(Student st: studentList){
            System.out.println(st.getFname());
            }
        }
}
```

Line: 59 Col: 1

You have earned 10.00 points!

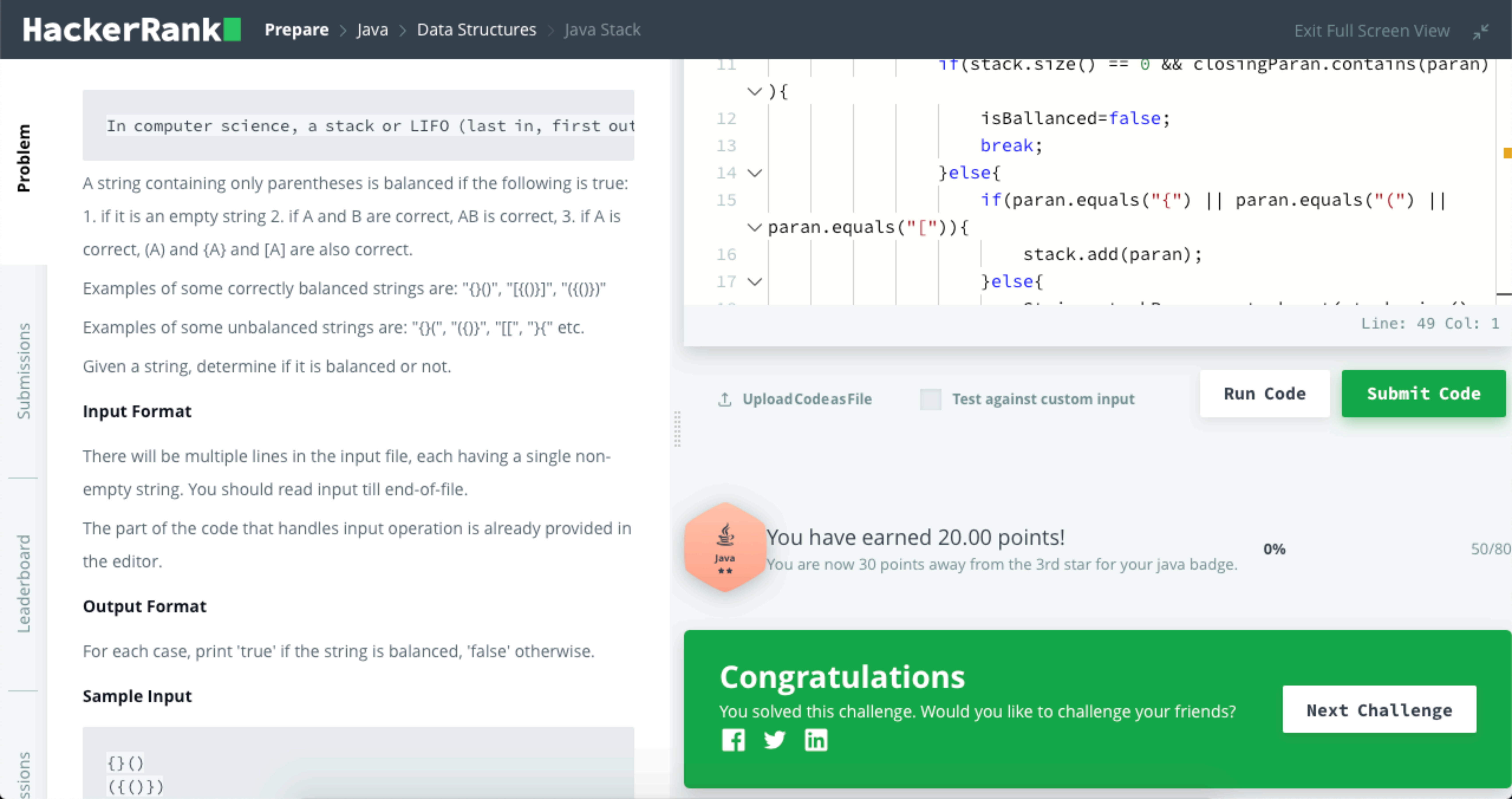
You are now 20 points away from the 3rd star for your java badge.

33%

60/80

Congratulations

You solved this challenge. Would you like to challenge your friends?



```
In computer science, a stack or LIFO (last in, first out
```

A string containing only parentheses is balanced if the following is true: 1. if it is an empty string 2. if A and B are correct, AB is correct, 3. if A is correct, (A) and {A} and [A] are also correct.

Examples of some correctly balanced strings are: "{}()", "[{()}]", "({()})"

Examples of some unbalanced strings are: "{}(", "({)}", "[[", "}{" etc.

Given a string, determine if it is balanced or not.

Input Format

There will be multiple lines in the input file, each having a single non-empty string. You should read input till end-of-file.

The part of the code that handles input operation is already provided in the editor.

Output Format

For each case, print 'true' if the string is balanced, 'false' otherwise.

Sample Input

```
{}()
({()})
```

```
                if(stack.size() == 0 && closingParan.contains(paran)
){
                    isBallanced=false;
                    break;
                }else{
                    if(paran.equals("{") || paran.equals("(") ||
paran.equals("[")){
                        stack.add(paran);
                    }else{
```

Line: 49 Col: 1

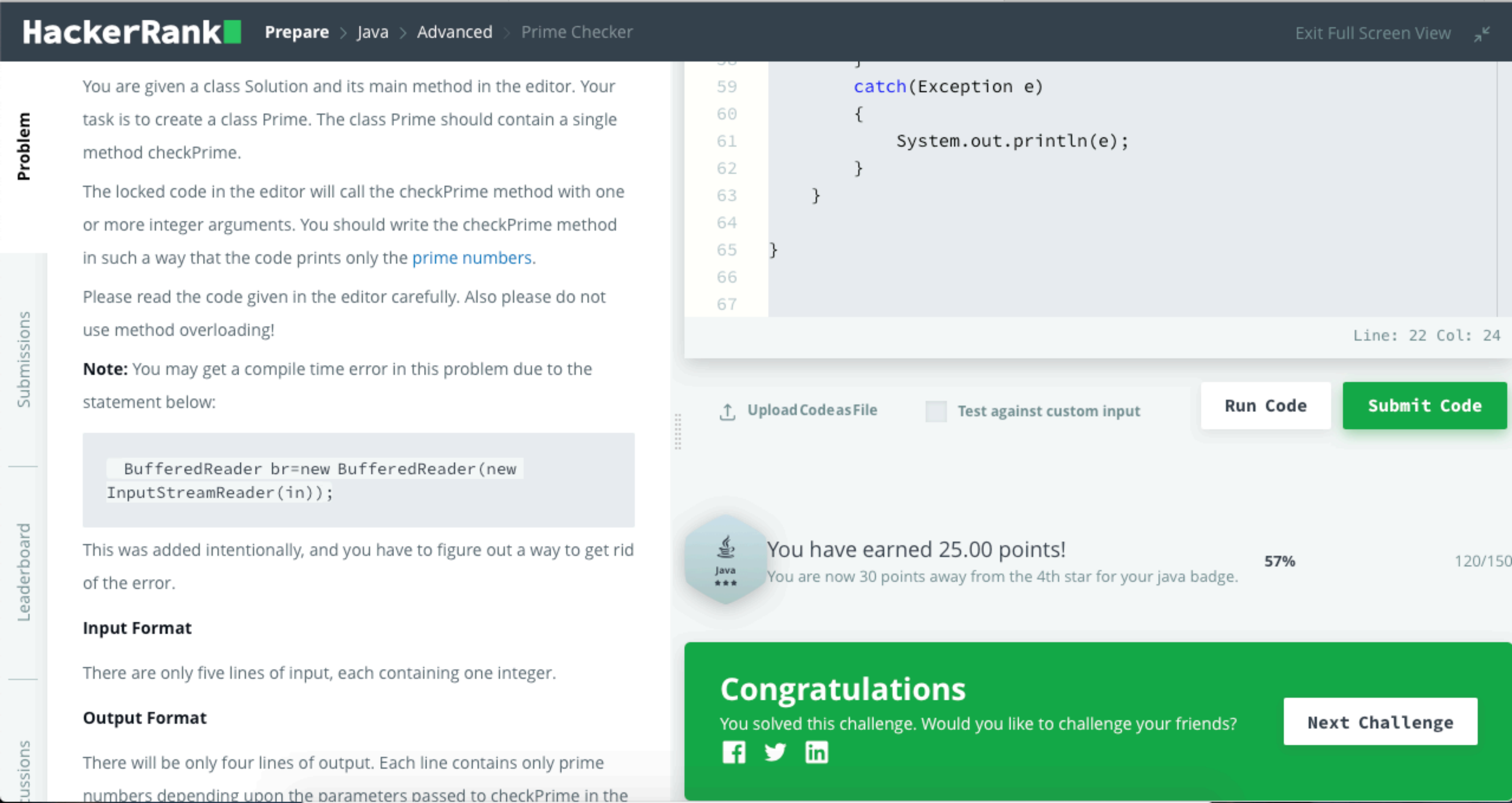
You have earned 20.00 points!
You are now 30 points away from the 3rd star for your java badge.

0%

50/80

Congratulations

You solved this challenge. Would you like to challenge your friends?



You are given a class Solution and its main method in the editor. Your task is to create a class Prime. The class Prime should contain a single method checkPrime.

The locked code in the editor will call the checkPrime method with one or more integer arguments. You should write the checkPrime method in such a way that the code prints only the prime numbers.

Please read the code given in the editor carefully. Also please do not use method overloading!

Note: You may get a compile time error in this problem due to the statement below:

```
BufferedReader br=new BufferedReader(new
InputStreamReader(in));
```

This was added intentionally, and you have to figure out a way to get rid of the error.

Input Format

There are only five lines of input, each containing one integer.

Output Format

There will be only four lines of output. Each line contains only prime numbers depending upon the parameters passed to checkPrime in the

```
        catch(Exception e)
        {
            System.out.println(e);
        }
    }

}
```

You have earned 25.00 points!

You are now 30 points away from the 4th star for your java badge.

Congratulations

You solved this challenge. Would you like to challenge your friends?