Assignment -2 Of JAVA

Submitted By:

Samrath Singh
18BCS8047
17 CI-4 (A)
Cluster – 1

Question 1: Tywins Tactics

Tywin Lannister is a tactical genius. At the heart of his tactical skills, is the way he has organized his armies, and the way he is able to estimate his soldiers' skill-levels, thus helping him make crucial decisions as to whom to dispatch to which areas of the war.

His army is organized in the form of a hierarchy **-** indeed it is a tree, with him as the root. We say "**A has immediate superior B**" if **A** reports directly to **B**. We further say "**A has superior B**" if there is a chain of soldiers starting with **A**, ending with **B**, and where each soldier reports directly to the next soldier of the chain. Further, each soldier is assigned an initial **skill-level** based on prior experience and battle proficiency.

In order for Tywin to decide whom to send to which battle, he has the following scheme: He chooses a particular soldier **S** as the leader of his temporary 'regiment', and sends in to battle, **S** as well as all the soldiers that have **S** as one of their superiors. He estimates the skill level of the regiment as the total skill level of all the soldiers under **S** (denoted by query "**Q S**").

After a battle, he may want to update the skill levels of some soldiers. If he wants to update the skill level of soldier **S** to value **x**, it is denoted by the query "**U S x**".

You are given the structure of the army, whose size is **N**, the initial skill levels of all the individual soldiers, as well the number of queries **M**. For each query of the type "**Q S**", report the sum of skill-levels of all the soldiers who have **S** as their superior.

Note: The soldiers are numbered 1 to **N**, and Tywin is given the number **1**.

Input

The first line consists of the integers **N** and **M**, denoting the number of soldiers and the number of queries.
This is followed by a single line consisting of **N** nonnegative values - the **skill levels of the N soldiers**.
This is then followed by **N-1** lines consisting of pairs of integers **(u, v)**, denoting that either **u is an immediate superior of v, or vice-versa**.
Finally you are given the **M** queries, of either the form "**U S x**" or "**Q S**".

Output

For each "**Q S**" query, output the sum of skill values of all the soldiers under **S**.

Constraints

- $\mathbf{1 \leq N \leq 10^5}$
- $\mathbf{1 \leq M \leq 10^5}$
- All skill values given with be in the range [**0**, **20,000**]
- $\mathbf{1 \leq S \leq N}$ for all queries
- All soldiers will have soldier **1** (Tywin) as their superior

---

Example

Input:

5 8

7 2 0 5 8

1 2

2 3

2 4

1 5

Q 1

Q 2

U 2 4

Q 1

Q 2

U 5 3

Q 1

Q 2

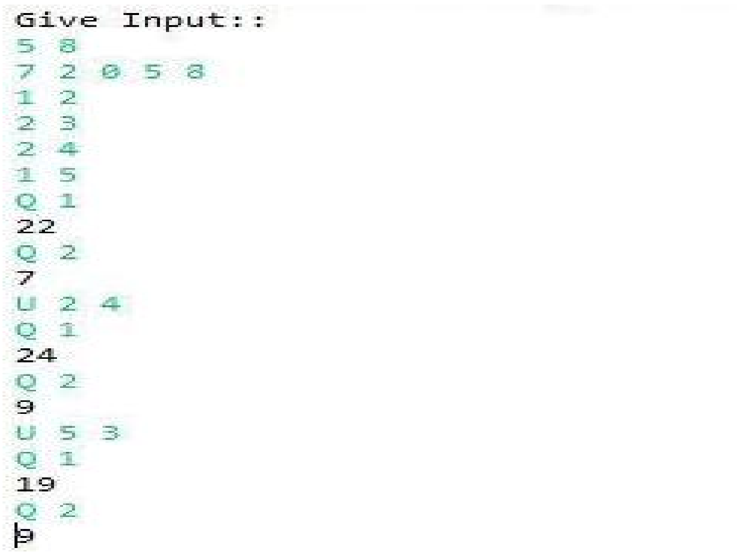
Output:

```
22

7

24

9

19

9
```

Solution:

```
import java.util.LinkedHashSet;
import java.util.*;

public class TywinsTack {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.println("Give Input:: ");
      int no_of_person = sc.nextInt();
      int no_of_query = sc.nextInt();
      LinkedHashSet <Integer>all = new LinkedHashSet<Integer>(); int check;
      int sum = 0;
      int exper [] = new int[no_of_person];
      int arr2D [][] = new int [no_of_person-1][2]; for(int i = 0; i < no_of_person; i++) { exper[i] =
sc.nextInt();

      }

      for(int row = 0; row < no_of_person-1; row++)
      {
         for(int col = 0; col<2; col++) {
            arr2D[row][col] = sc.nextInt();

      }
      }

      int ch[];

      for(int i = 0; i<no_of_query; i++){ String criteria = sc.next(); if(criteria.equals("Q")) {

         check = sc.nextInt(); sum = 0;

         sum = sum +exper[check-1]; all.add(check); while(all.size() > 0) {
                check = (int)(all.toArray())[0];

                for(int row = 0; row < no_of_person-1; row++) { if(arr2D[row][0]==check) {

                   int col = 1;

                   sum = sum + exper[arr2D[row][col] - 1]; all.add(arr2D[row][col]);

                }

                }

                all.remove(check);

             }
             System.out.println(sum);

          }

          else if(criteria.equals("U")) { exper[sc.nextInt()-1] =   sc.nextInt();

          }
          }

       }

    }
```

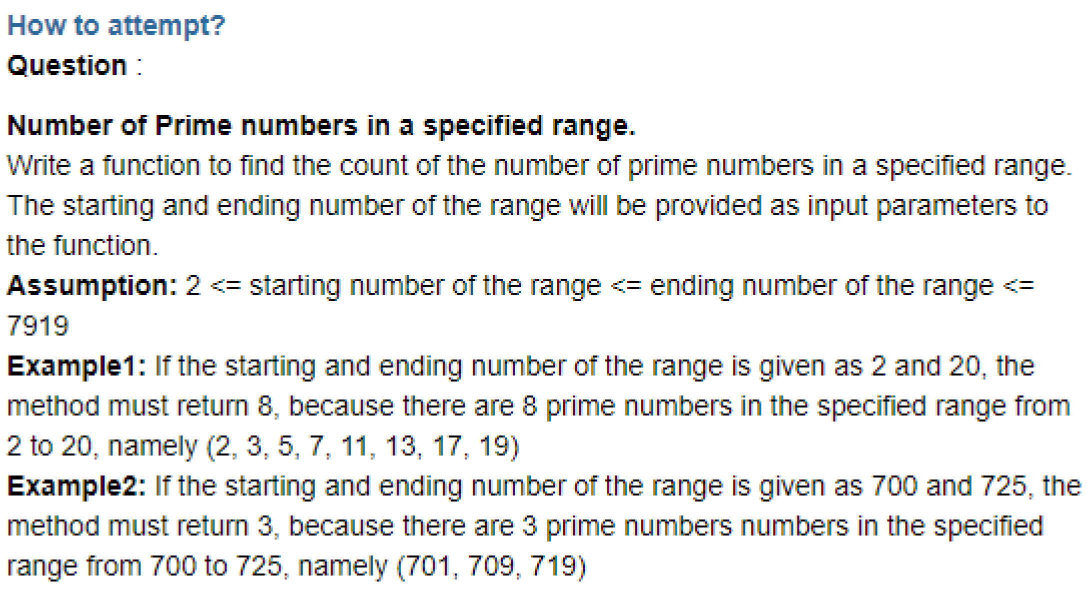
Output:



```
Give Input::
5 8
7 2 0 5 8
1 2
2 3
2 4
1 5
Q 1
22
Q 2
7
U 2 4
Q 1
24
Q 2
9
U 5 3
Q 1
19
Q 2
9
```

Question 2: Mettl Problem

Mettl Link: https://tests.mettl.com/authenticateKey/87c41143



How to attempt?

Question :

Number of Prime numbers in a specified range.

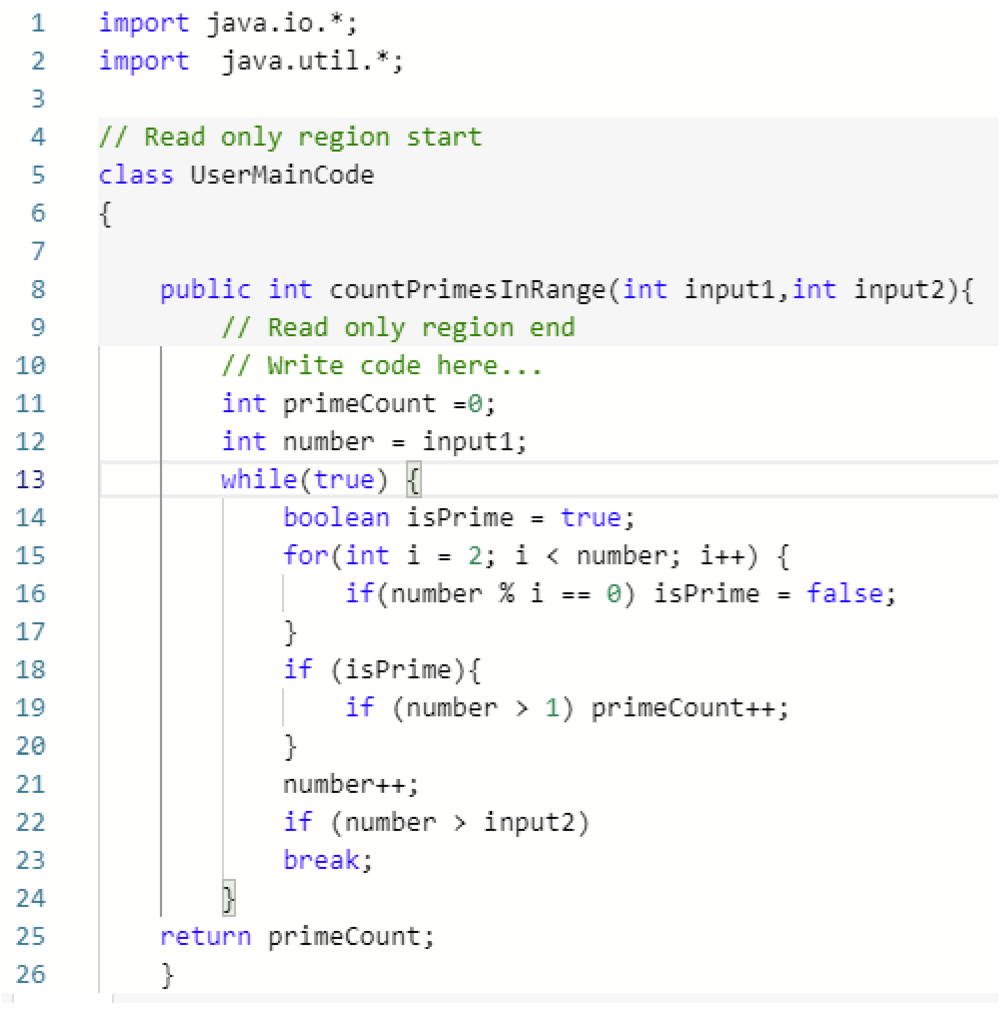
Write a function to find the count of the number of prime numbers in a specified range. The starting and ending number of the range will be provided as input parameters to the function.

Assumption: 2 <= starting number of the range <= ending number of the range <= 7919

Example1: If the starting and ending number of the range is given as 2 and 20, the method must return 8, because there are 8 prime numbers in the specified range from 2 to 20, namely (2, 3, 5, 7, 11, 13, 17, 19)

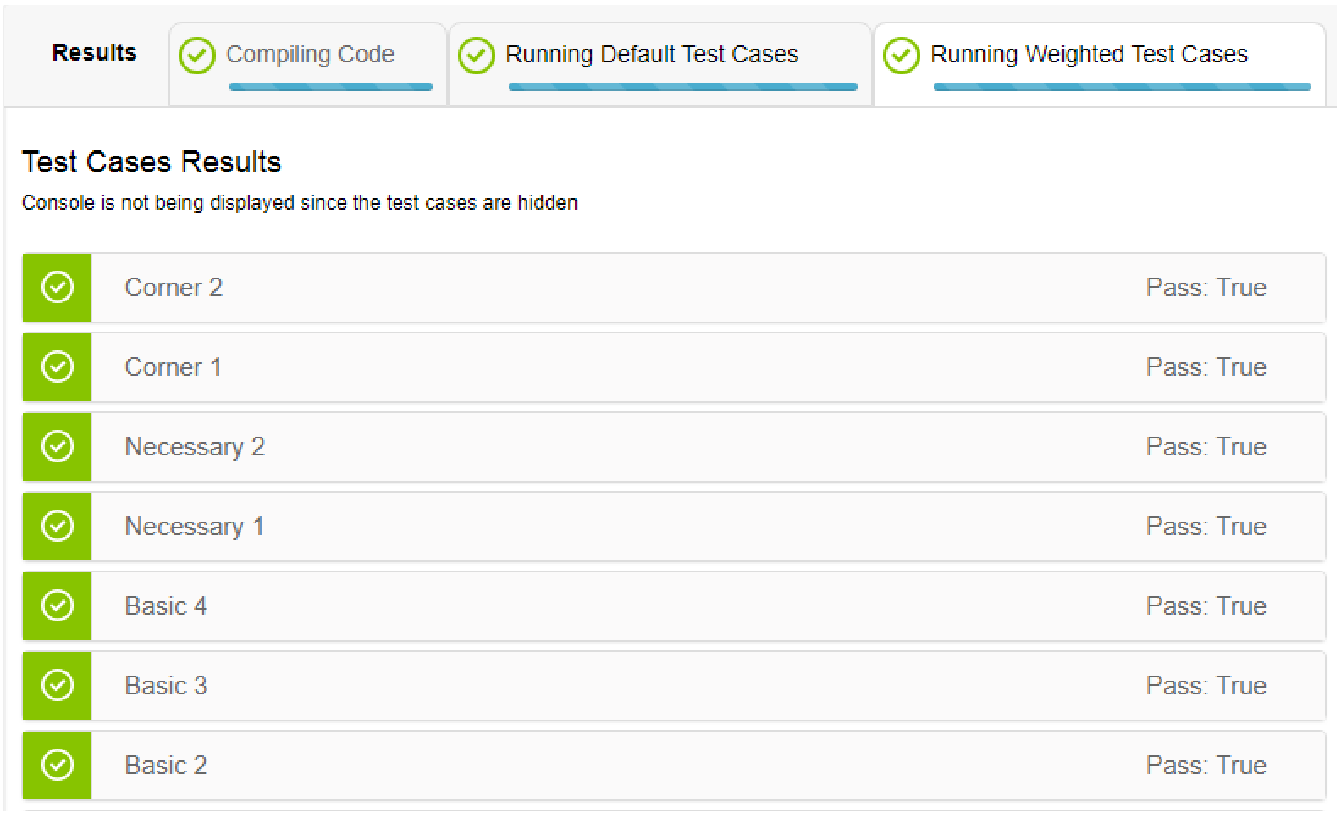
Example2: If the starting and ending number of the range is given as 700 and 725, the method must return 3, because there are 3 prime numbers numbers in the specified range from 700 to 725, namely (701, 709, 719)

Solution:



```java
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

    public int countPrimesInRange(int input1,int input2){
        // Read only region end
        // Write code here...
        int primeCount =0;
        int number = input1;
        while(true) {
            boolean isPrime = true;
            for(int i = 2; i < number; i++) {
                if(number % i == 0) isPrime = false;
            }
            if (isPrime){
                if (number > 1) primeCount++;
            }
            number++;
            if (number > input2)
            break;
        }
    return primeCount;
    }
```

Output:



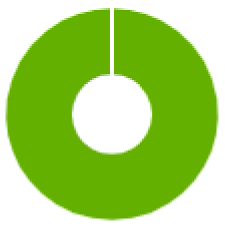
Results | Compiling Code | Running Default Test Cases | Running Weighted Test Cases

Test Cases Results

Console is not being displayed since the test cases are hidden

✓	Corner 2	Pass: True
✓	Corner 1	Pass: True
✓	Necessary 2	Pass: True
✓	Necessary 1	Pass: True
✓	Basic 4	Pass: True
✓	Basic 3	Pass: True
✓	Basic 2	Pass: True

SUMMARY OF ATTEMPTS



1 Correct
(Scored 40/40)

Question 3: Brain Storming Problem

Given a string s and a non-empty string p, find all the start indices of p's anagrams in s.

Strings consists of lowercase English letters only and the length of both strings s and p will not be larger than 20,100.

The order of output does not matter.

Example 1:
Input:
s: "cbaebabacd" p: "abc"
Output:
[0, 6]

Explanation:
The substring with start index = 0 is "cba", which is an anagram of "abc".
The substring with start index = 6 is "bac", which is an anagram of "abc".

Example 2:
Input:
s: "abab" p: "ab"
Output:
[0, 1, 2]

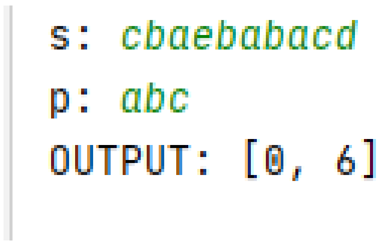
Explanation:
The substring with start index = 0 is "ab", which is an anagram of "ab".
The substring with start index = 1 is "ba", which is an anagram of "ab".
The substring with start index = 2 is "ab", which is an anagram of "ab".

Solution:

```
import java.util.*;
public class findAnagrams {
  public List<Integer> find(String s, String p) {
    List<Integer> rst = new ArrayList<>();
    if (s == null || s.length() == 0 || s.length() < p.length()) {
      return rst;
    }

    int[] map_p = new int[26];
    for (int i = 0; i < p.length(); i++) {
      map_p[p.charAt(i) - 'a']++;
    }
```

```
        for (int i = 0; i < s.length() - p.length(); i++) {
            int[] map_s = new int[26];
            for (int j = 0; j < p.length(); j++) {
                map_s[s.charAt(i+j) - 'a']++;
            }
            if (isMatch(map_p, map_s)) {
                rst.add(i);
            }
        }
        return rst;
    }
    public boolean isMatch(int[] arr1, int[] arr2) {
        for (int i = 0; i < arr1.length; i++) {
            if (arr1[i] != arr2[i]) {
                return false;
            }
        }
        return true;
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("s: ");
        String s = sc.nextLine();
        System.out.print("p: ");
        String p = sc.nextLine();
        findAnagrams fa = new findAnagrams();
        System.out.println("OUTPUT: "+fa.find(s,p));
    }
}
```

Output:



```
s: cbaebabacd
p: abc
OUTPUT: [0, 6]
```

Question 4: Java Stack

In computer science, a stack or LIFO (last in, first out) is an abstract data type that serves as a collection of elements, with two principal operations: push, which adds an element to the collection, and pop, which removes the last element that was added.(Wikipedia)

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 stringsare: "{}()", "[{()}]", "({()})"

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.

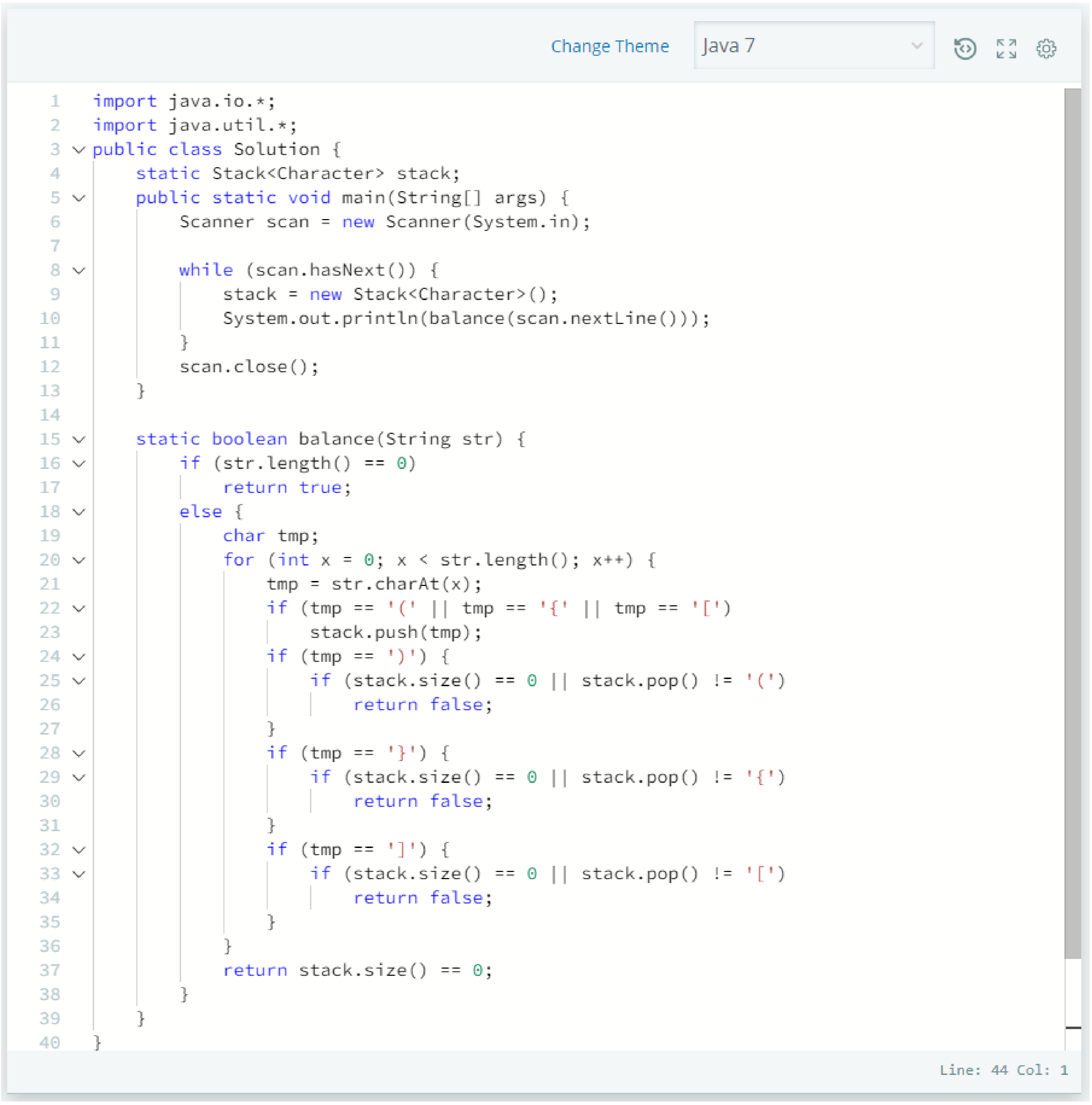
Sample Input

```
{}()
({()})
{}(
[]
```

Sample Output

```
true
true
false
true
```

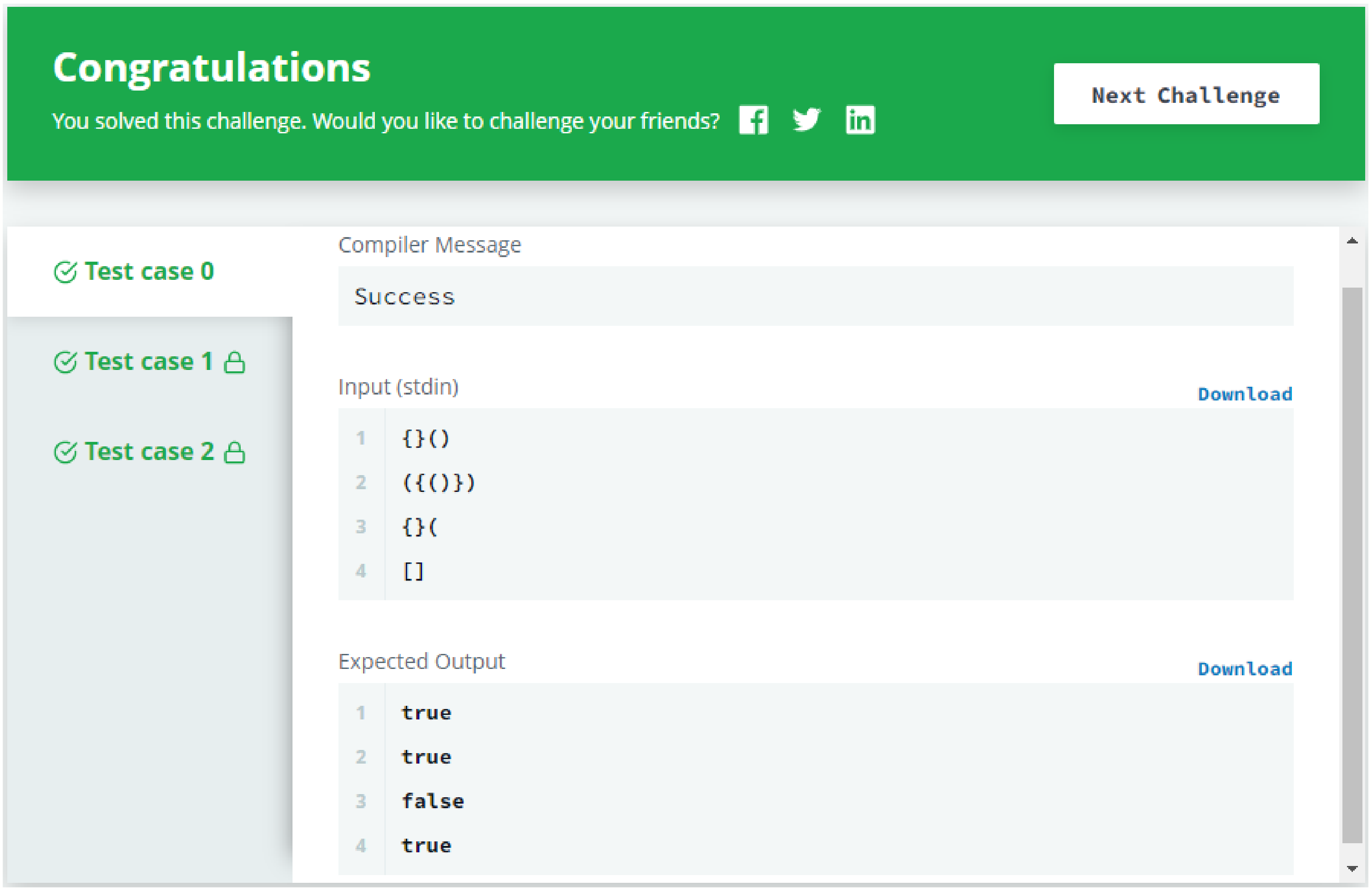
Solution:


Change Theme
Java 7

```java
import java.io.*;
import java.util.*;
public class Solution {
    static Stack<Character> stack;
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        while (scan.hasNext()) {
            stack = new Stack<Character>();
            System.out.println(balance(scan.nextLine()));
        }
        scan.close();
    }

    static boolean balance(String str) {
        if (str.length() == 0)
            return true;
        else {
            char tmp;
            for (int x = 0; x < str.length(); x++) {
                tmp = str.charAt(x);
                if (tmp == '(' || tmp == '{' || tmp == '[')
                    stack.push(tmp);
                if (tmp == ')') {
                    if (stack.size() == 0 || stack.pop() != '(')
                        return false;
                }
                if (tmp == '}') {
                    if (stack.size() == 0 || stack.pop() != '{')
                        return false;
                }
                if (tmp == ']') {
                    if (stack.size() == 0 || stack.pop() != '[')
                        return false;
                }
            }
            return stack.size() == 0;
        }
    }
}
```

Line: 44 Col: 1

Output:



Congratulations

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

Next Challenge

Test case 0

Test case 1

Test case 2

Compiler Message

```
Success
```

Input (stdin)

Download

```
{}()
({()})
{}(
[]
```

Expected Output

Download

```
true
true
false
true
```

Question 5: Prime Checker

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 *main* method of the class *Solution*. In case there is no prime number, then a blank line should be printed.

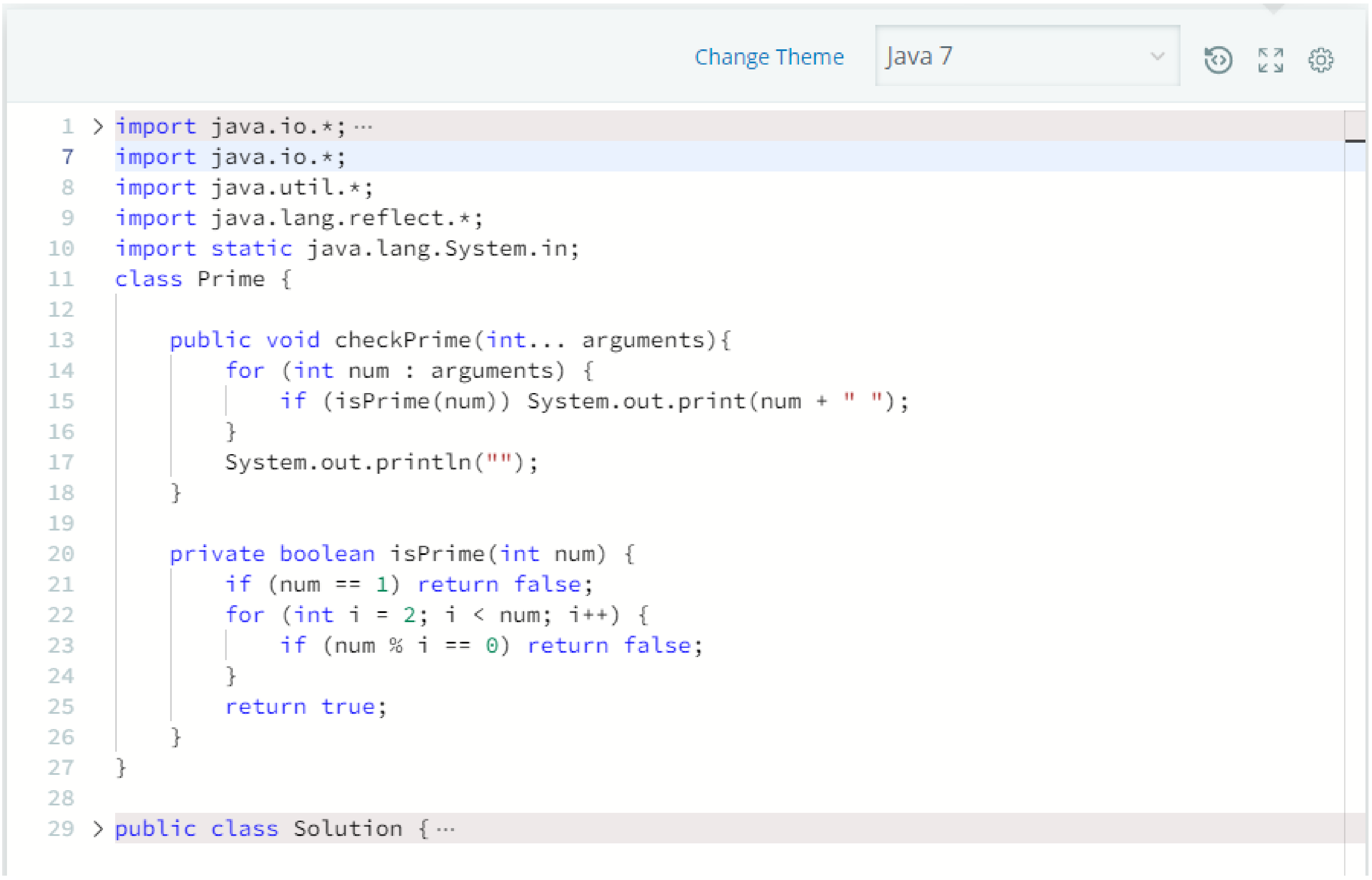
Sample Input

```
2
1
3
4
5
```

Sample Output

```
2
2
2 3
2 3 5
```

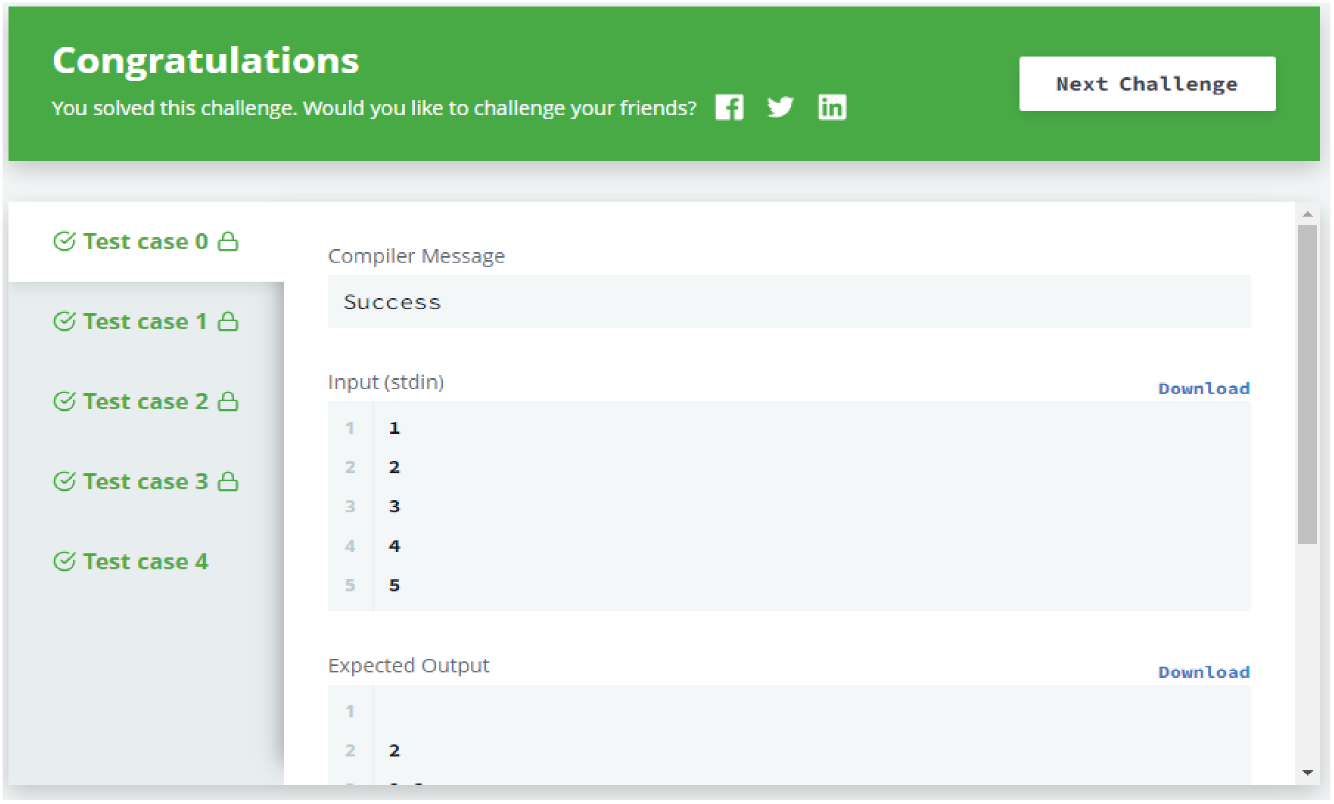
Solution:



Change Theme   Java 7

```
import java.io.*;⋯
import java.io.*;
import java.util.*;
import java.lang.reflect.*;
import static java.lang.System.in;
class Prime {

    public void checkPrime(int... arguments){
        for (int num : arguments) {
            if (isPrime(num)) System.out.print(num + " ");
        }
        System.out.println("");
    }

    private boolean isPrime(int num) {
        if (num == 1) return false;
        for (int i = 2; i < num; i++) {
            if (num % i == 0) return false;
        }
        return true;
    }
}

public class Solution {⋯
```

Output:



Congratulations

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

Next Challenge

Test case 0
Test case 1
Test case 2
Test case 3
Test case 4

Compiler Message

```
Success
```

Input (stdin) Download

```
1
2
3
4
5
```

Expected Output Download

```

2
```

Question 6: Java BigDecimal

Java's BigDecimal class can handle arbitrary-precision signed decimal 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.

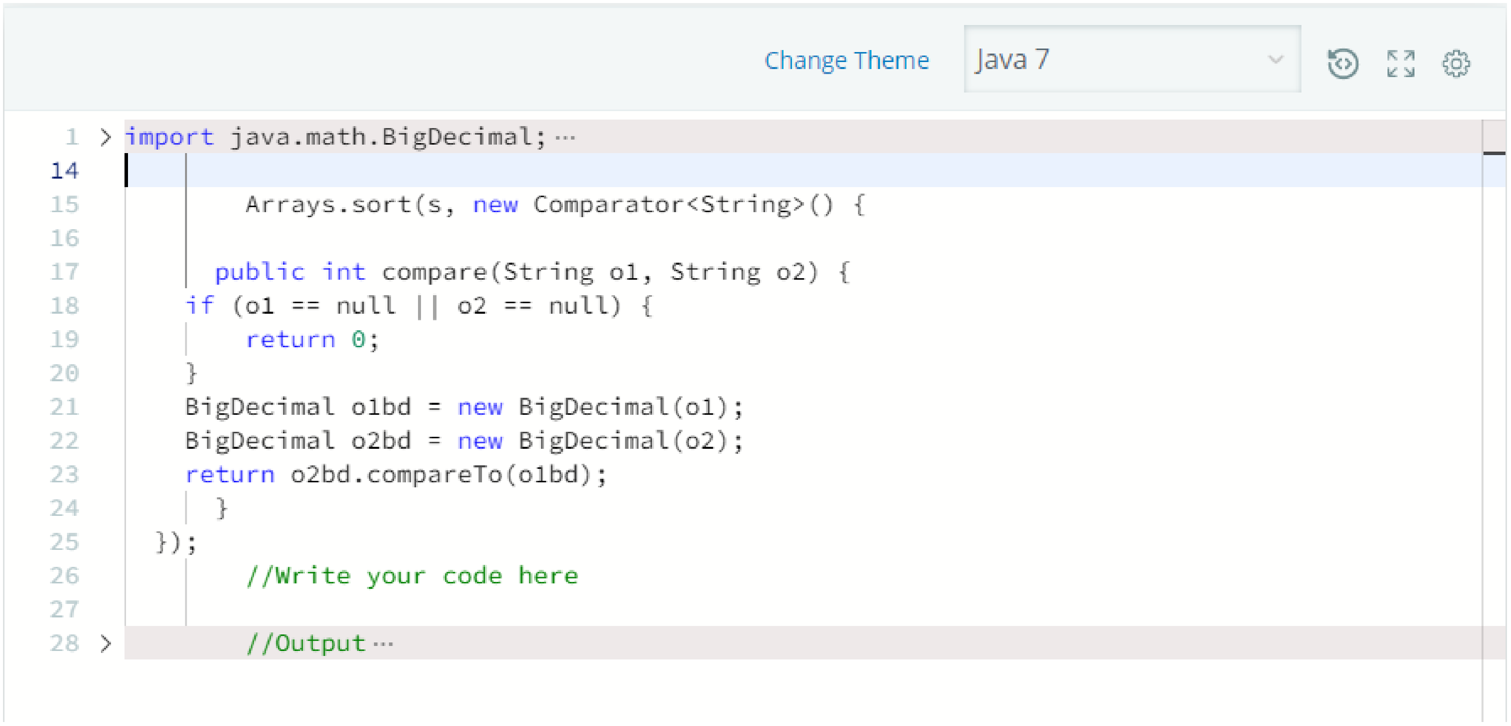
Sample Input

```
9
-100
50
0
56.6
90
0.12
.12
02.34
000.000
```

Sample Output

```
90
56.6
50
02.34
0.12
.12
0
000.000
-100
```

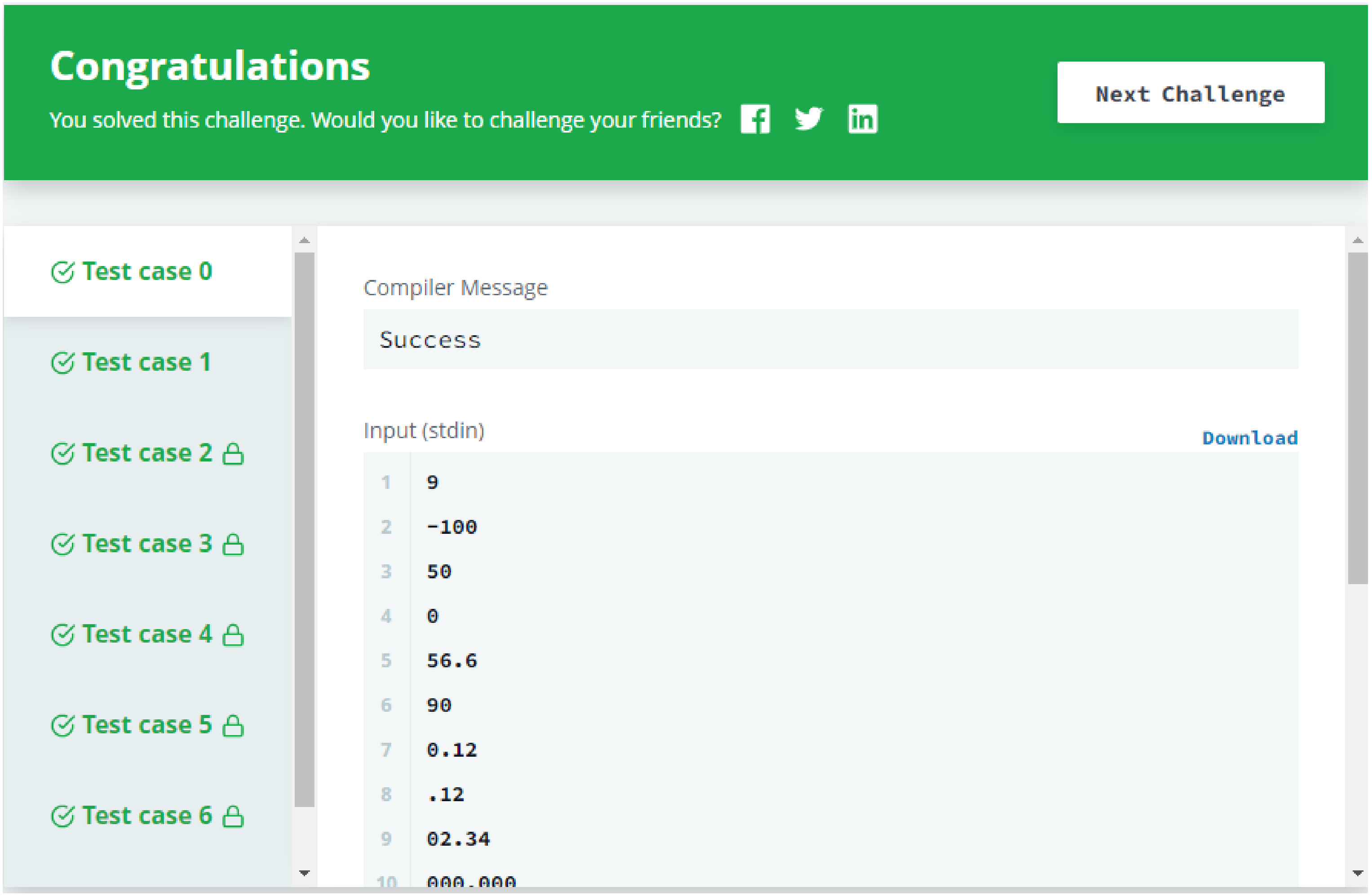
Solution:


Change Theme

Java 7

```
import java.math.BigDecimal;···

        Arrays.sort(s, new Comparator<String>() {

      public int compare(String o1, String o2) {
    if (o1 == null || o2 == null) {
        return 0;
    }
    BigDecimal o1bd = new BigDecimal(o1);
    BigDecimal o2bd = new BigDecimal(o2);
    return o2bd.compareTo(o1bd);
      }
  });
        //Write your code here

        //Output···
```

Output:


Congratulations

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

Next Challenge

Test case 0

Test case 1

Test case 2

Test case 3

Test case 4

Test case 5

Test case 6

Compiler Message

```
Success
```

Input (stdin)

Download

```
9
-100
50
0
56.6
90
0.12
.12
02.34
000.000
```

Question 1: PrimeSum

Abbas has been assigned the task to find the minimum number of prime numbers which when added will be equal to a given number N.

The prototype of the function is as below -

int howManyPrimeNumsWillGet(int input1);

The function takes as input a number **input1**

1<= **input1**<=10^6

The function should return the minimum number of prime numbers which when summed will result in **input1**.

If it is not possible to generate **input1** by adding prime numbers, the function should return -1

<u>Example1 –</u>

If **input1**=28

Output: The function should return 2

Explanation:

11+17 = 28.

Number of prime numbers which when added will produce 28 is **2**

So, the function returns **2**

<u>Example2 –</u>

If **input1**=14

Output: The function should return 2.

Explanation:

11+3 = 14.

Number of prime numbers which when added will produce 14 is **2**

So, the function returns **2**

Example3 –

If **input1**=43

Output: The function should return 1

Explanation:

43 can be achieved by adding 11+13+19, BUT 43 itself is a prime number, so the function should return **1**.

Example4 –

If **input1**=121

Output: The function should return 3

Explanation:

121 can be achieved by adding three prime numbers, i.e. 11+13+97=121, so the function should return **3**.

Example5 –

If **input1**=33

Output: The function should return 3

Explanation:

33 can be achieved by adding three prime numbers, i.e. 11+11+11=33,

and can also be achieved by adding 2+31 = 33, so the function should return **2**.

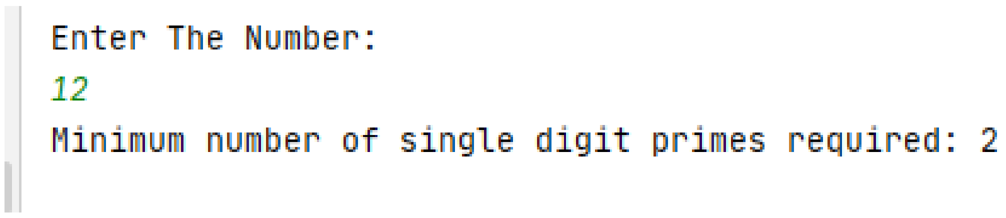
NOTE: The above examples are only few examples to help you understand the question. The actual test-case values will be different from these, so you must ensure to check the result for all possible cases.

Solution:

```
import java.util.*;

class primeSum {

  static boolean check(int i, int val)
  {
    if (i - val < 0)
```

```
        return false;
      else
        return true; }
    static double MinimumPrimes(int n){
      double[] dp;
      dp = new double[n+1];
      for (int i = 1; i <= n; i++)
        dp[i] = 1e9;

      dp[0] = dp[2] = dp[3] = dp[5] = dp[7] = 1;
      for (int i = 1; i <= n; i++) {

        if (check(i, 2))
          dp[i] = Math.min(dp[i], 1 + dp[i - 2]);

        if (check(i, 3))
          dp[i] = Math.min(dp[i], 1 + dp[i - 3]);

        if (check(i, 5))

          dp[i] = Math.min(dp[i], 1 + dp[i - 5]);

        if (check(i, 7))
          dp[i] = Math.min(dp[i], 1 + dp[i - 7]);
      }
      if (dp[n] == (1e9))
        return -1;
      else
        return dp[n]; }
    public static void main(String args[]) {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter The Number: ");
      int n = sc.nextInt();
      int minimal = (int)MinimumPrimes(n);

      if (minimal != -1)
        System.out.println("Minimum number of single "+
              "digit primes required: "+minimal);
      else
        System.out.println("Not Possible");
    }}
```

Output:



```
Enter The Number:
12
Minimum number of single digit primes required: 2
```

Question 2: Mettl Problem

Mettl Problem: https://tests.mettl.com/authenticateKey/28c41d9d

How to attempt?

Question :

Is Palindrome Number?

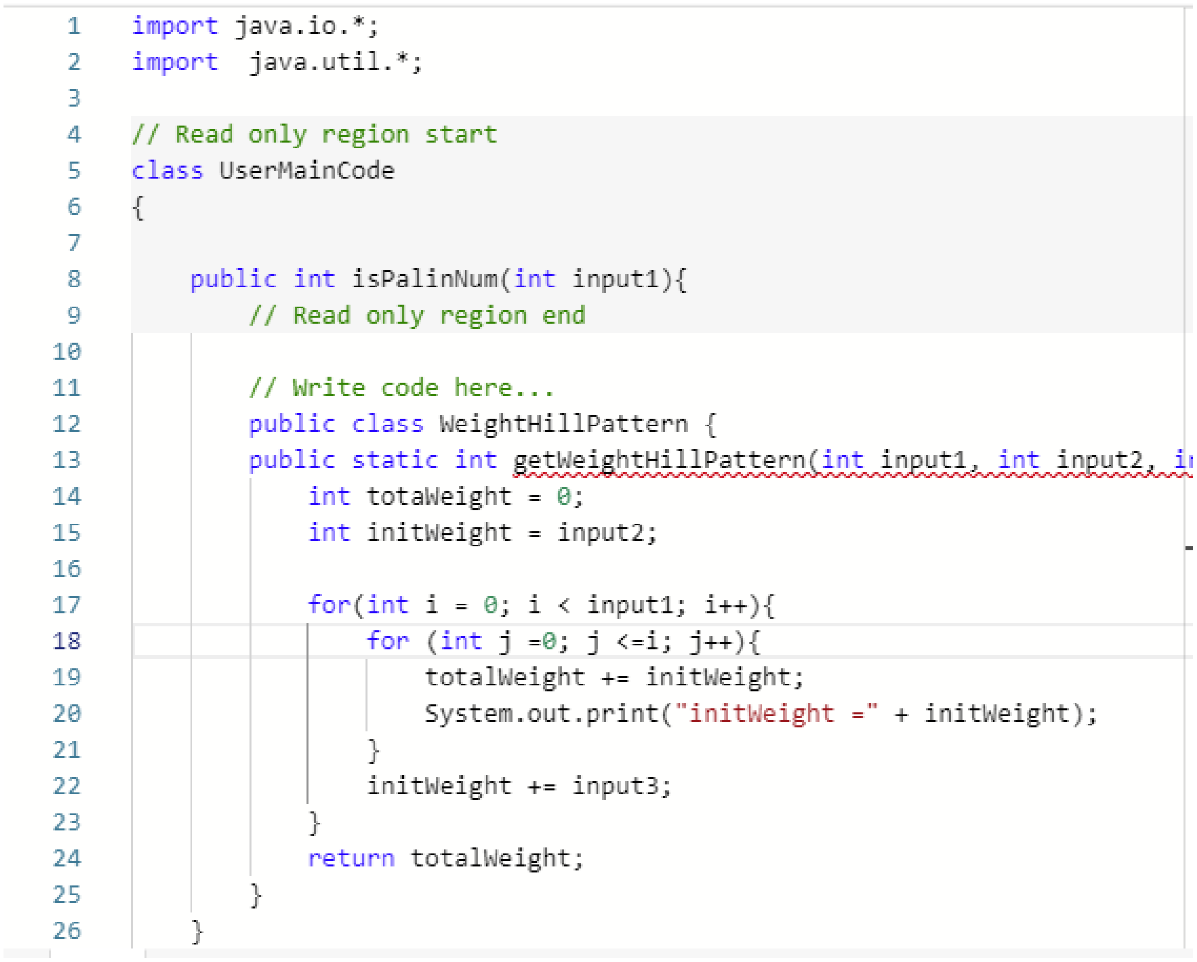
Write a function to find whether the given number N is a palindrome.

A palindrome number is one that reads the same backwards as well as forwards. For e.g. 252, 18981, 5005 are examples of palindrome numbers.

The number will be passed to the function as an input parameter of type int.
If the number is a palindrome, the function should return 2, else it should return 1.

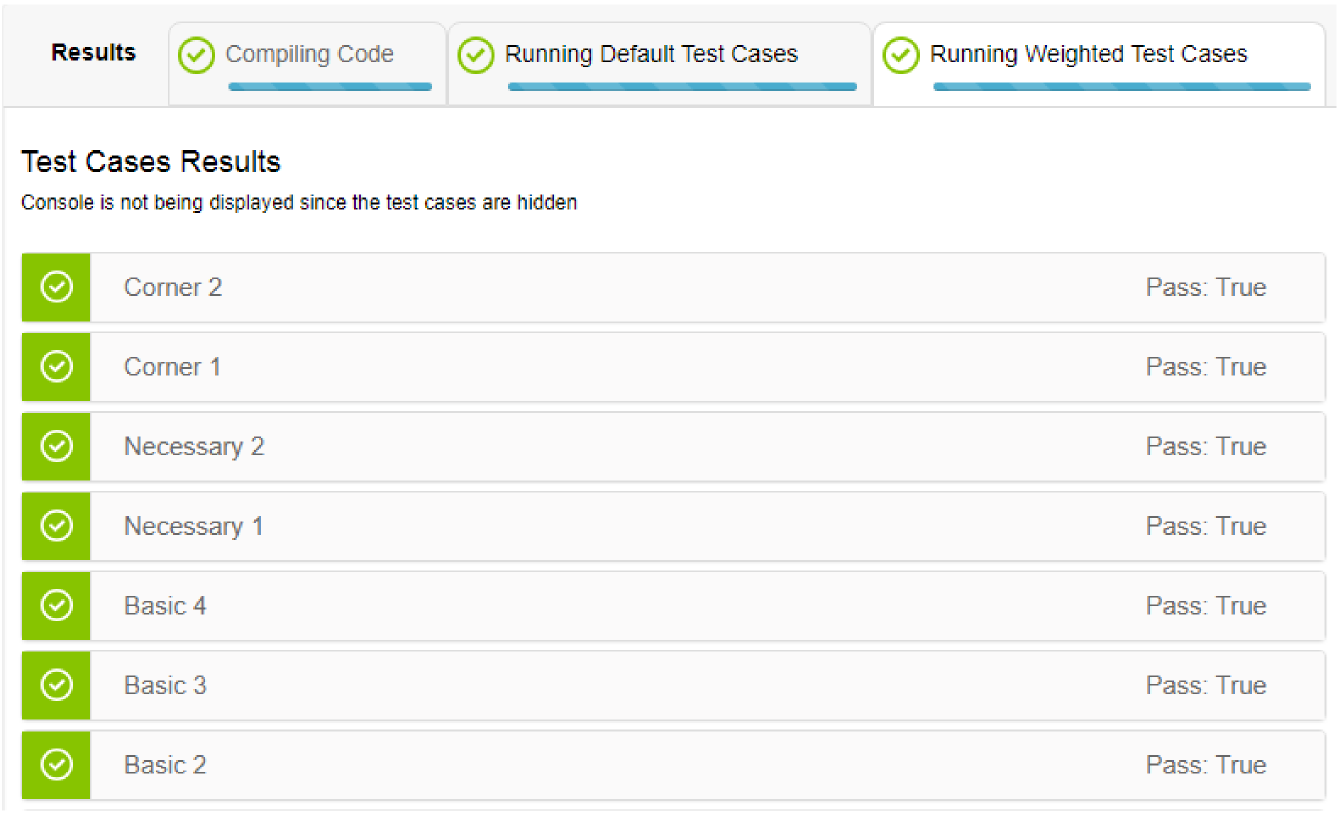
Assumption: The input number will be a positive integer number >= 1 and <= 25000.

Solution:



```java
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

    public int isPalinNum(int input1){
        // Read only region end

        // Write code here...
        public class WeightHillPattern {
        public static int getWeightHillPattern(int input1, int input2, i
            int totaWeight = 0;
            int initWeight = input2;

            for(int i = 0; i < input1; i++){
                for (int j =0; j <=i; j++){
                    totalWeight += initWeight;
                    System.out.print("initWeight =" + initWeight);
                }
                initWeight += input3;
            }
            return totalWeight;
        }
    }
```

Output:



Results | Compiling Code | Running Default Test Cases | Running Weighted Test Cases

Test Cases Results

Console is not being displayed since the test cases are hidden

✓	Corner 2	Pass: True
✓	Corner 1	Pass: True
✓	Necessary 2	Pass: True
✓	Necessary 1	Pass: True
✓	Basic 4	Pass: True
✓	Basic 3	Pass: True
✓	Basic 2	Pass: True

SUMMARY OF ATTEMPTS

1 Correct
(Scored 40/40)

Question 3: Brain Storming Problem

Given an array of integers and an integer k, you need to find the total number of continuous subarrays whose sum equals to k.

Example 1:

Input:nums = [1,1,1], k = 2

Output: 2

Note:

The length of the array is in range [1, 20,000].

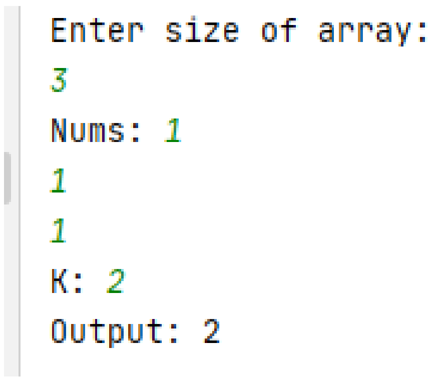
The range of numbers in the array is [-1000, 1000] and the range of the integer k is [-1e7, 1e7].

Solution:

```
import java.util.*;
public class subArraySum {
   public int sumArray(int[] nums, int k) {
      HashMap<Integer, Integer> map = new HashMap<>();
      map.put(0, 1);

      int count = 0;
      int sum = 0;

      for(int i=0; i<nums.length; i++){
         sum += nums[i];
         int n = map.getOrDefault(sum-k, 0);
         count += n;
         map.put(sum, map.getOrDefault(sum,0)+1);
      }
      return count;
   }
   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);
      System.out.println("Enter size of array: ");
      int leng = input.nextInt();
      int[] numbers = new int[leng];

      System.out.print("Nums: ");
      for (int i = 0; i < numbers.length; i++)
      {
         numbers[i] = input.nextInt();
      }
      System.out.print("K: ");
      int k = input.nextInt();
      subArraySum sm = new subArraySum();
      System.out.println("Output: "+sm.sumArray(numbers,k));
```

```
    }
  }
```

Output:



```
Enter size of array:
3
Nums: 1
1
1
K: 2
Output: 2
```

Question 4: Java Currency Formatter

HackerRank

Given a double-precision number, $payment$, denoting an amount of money, use the NumberFormat class' getCurrencyInstance method to convert $payment$ into the US, Indian, Chinese, and French currency formats. Then print the formatted values as follows:

```
US: formattedPayment
India: formattedPayment
China: formattedPayment
France: formattedPayment
```

where $formattedPayment$ is $payment$ formatted according to the appropriate Locale's currency.

Note: India does not have a built-in Locale, so you must construct one where the language is `en` (i.e., English).

Input Format

A single double-precision number denoting $payment$.

Constraints

- $0 \le payment \le 10^9$

Output Format

On the first line, print `US: u` where u is $payment$ formatted for US currency.
On the second line, print `India: i` where i is $payment$ formatted for Indian currency.
On the third line, print `China: c` where c is $payment$ formatted for Chinese currency.
On the fourth line, print `France: f`, where f is $payment$ formatted for French currency.

Sample Input

```
12324.134
```

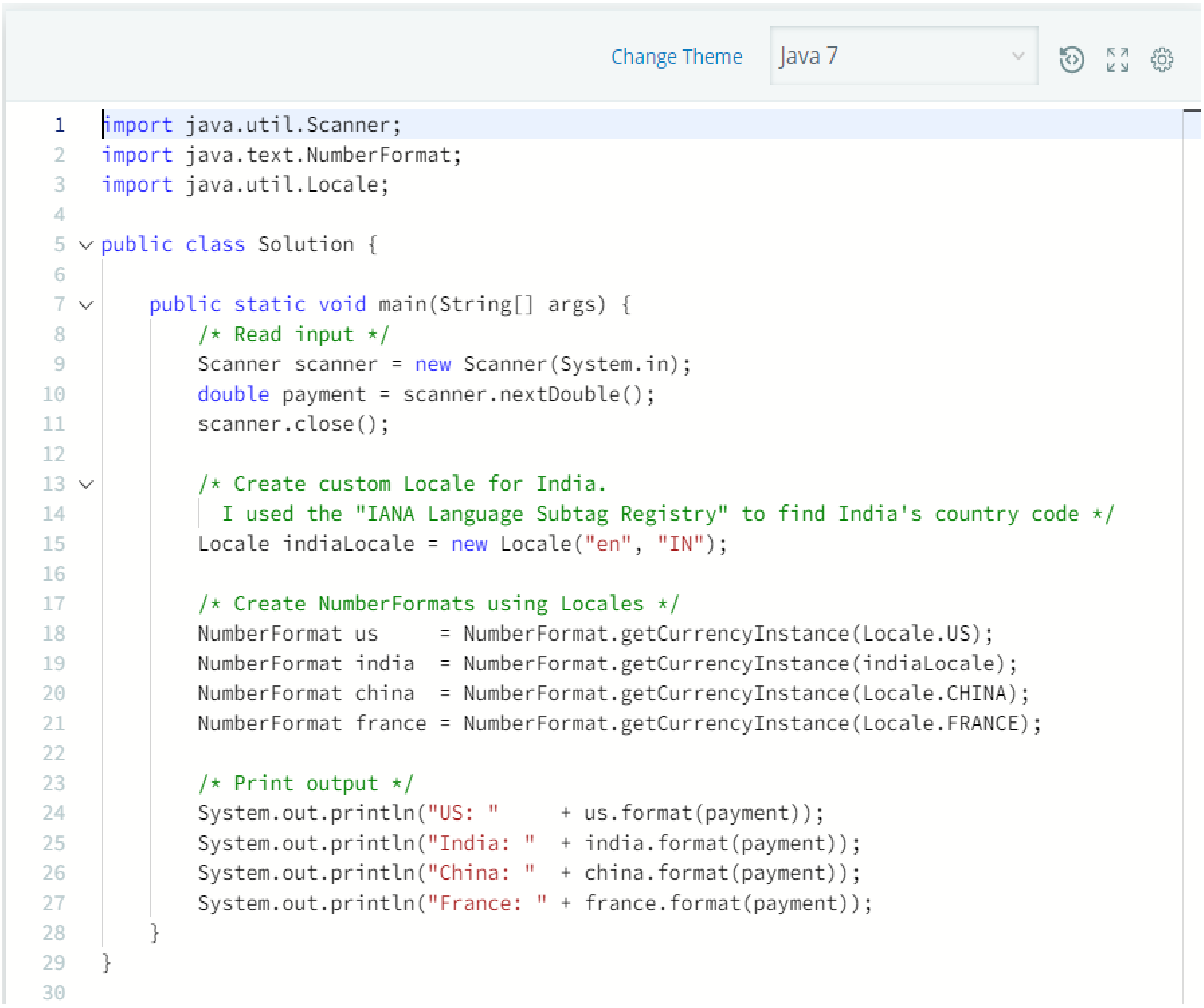
Sample Output

```
US: $12,324.13
India: Rs.12,324.13
China: #12,324.13
France: 12 324,13 €
```

Explanation

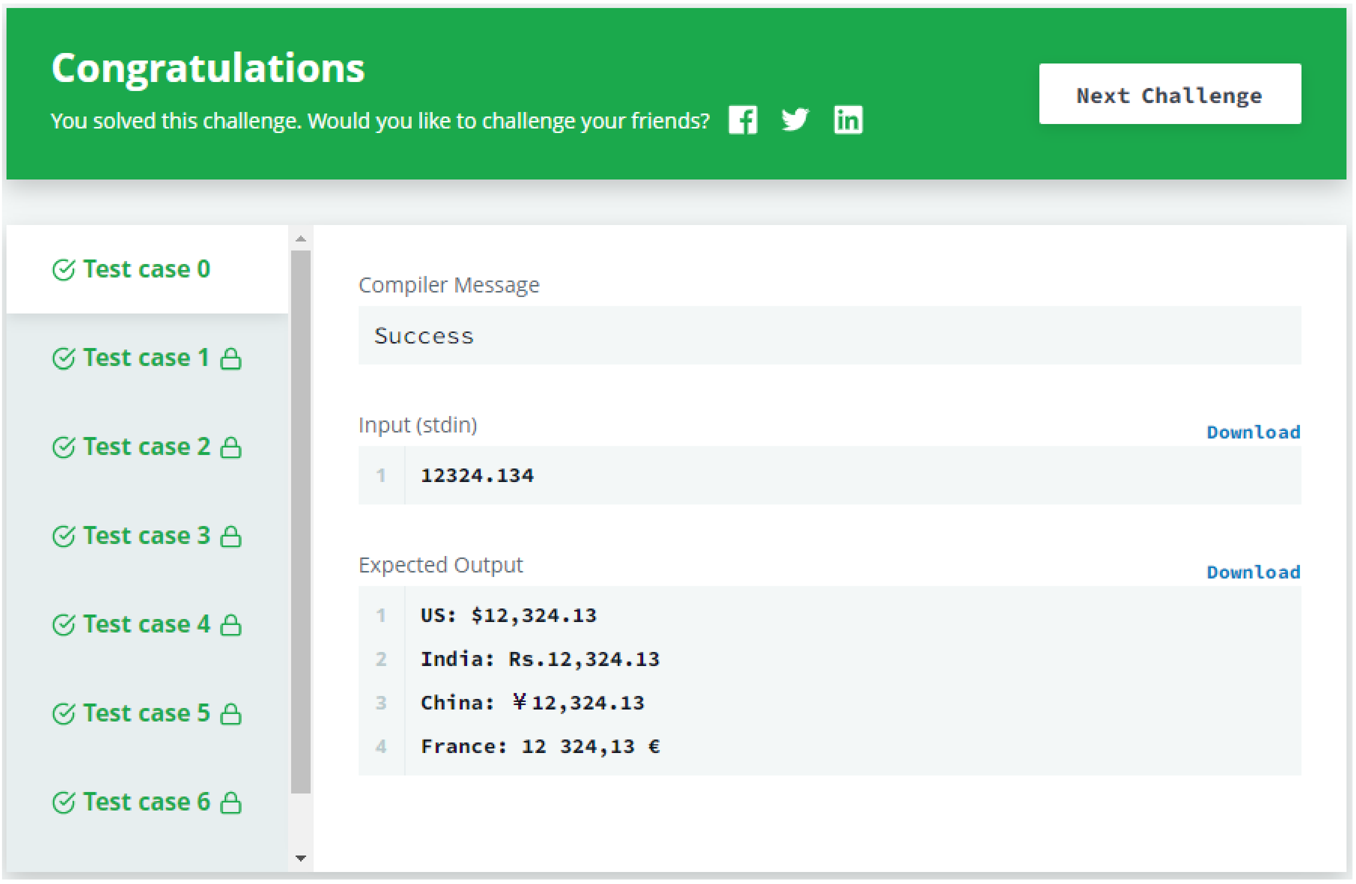
Each line contains the value of $payment$ formatted according to the four countries' respective currencies.

Solution:



```java
import java.util.Scanner;
import java.text.NumberFormat;
import java.util.Locale;

public class Solution {

    public static void main(String[] args) {
        /* Read input */
        Scanner scanner = new Scanner(System.in);
        double payment = scanner.nextDouble();
        scanner.close();

        /* Create custom Locale for India.
          I used the "IANA Language Subtag Registry" to find India's country code */
        Locale indiaLocale = new Locale("en", "IN");

        /* Create NumberFormats using Locales */
        NumberFormat us     = NumberFormat.getCurrencyInstance(Locale.US);
        NumberFormat india  = NumberFormat.getCurrencyInstance(indiaLocale);
        NumberFormat china  = NumberFormat.getCurrencyInstance(Locale.CHINA);
        NumberFormat france = NumberFormat.getCurrencyInstance(Locale.FRANCE);

        /* Print output */
        System.out.println("US: "     + us.format(payment));
        System.out.println("India: "  + india.format(payment));
        System.out.println("China: "  + china.format(payment));
        System.out.println("France: " + france.format(payment));
    }
}
```

Output:



Compiler Message

```
Success
```

Input (stdin)

```
12324.134
```

Expected Output

```
US: $12,324.13
India: Rs.12,324.13
China: ￥12,324.13
France: 12 324,13 €
```

HackerRank

Question 5: Java Substring Comparisons

We define the following terms:

- Lexicographical Order, also known as *alphabetic* or *dictionary* order, orders characters as follows:

$$\mathtt{A} < \mathtt{B} < \ldots < \mathtt{Y} < \mathtt{Z} < \mathtt{a} < \mathtt{b} < \ldots < \mathtt{y} < \mathtt{z}$$

  For example, ball < cat , dog < dorm , Happy < happy , Zoo < ball .

- A substring of a string is a contiguous block of characters in the string. For example, the substrings of abc are a , b, c , ab, bc, and abc.

Given a string, s, and an integer, k, complete the function so that it finds the lexicographically *smallest* and *largest* substrings of length k.

Input Format

The first line contains a string denoting s .
The second line contains an integer denoting k.

Constraints

- $1 \le |s| \le 1000$
- s consists of English alphabetic letters only (i.e., [a-zA-Z]).

Output Format

Return the respective lexicographically smallest and largest substrings as a single newline-separated string.

Sample Input 0

```
welcometojava
3
```

Sample Output 0

```
ava
wel
```

Explanation 0

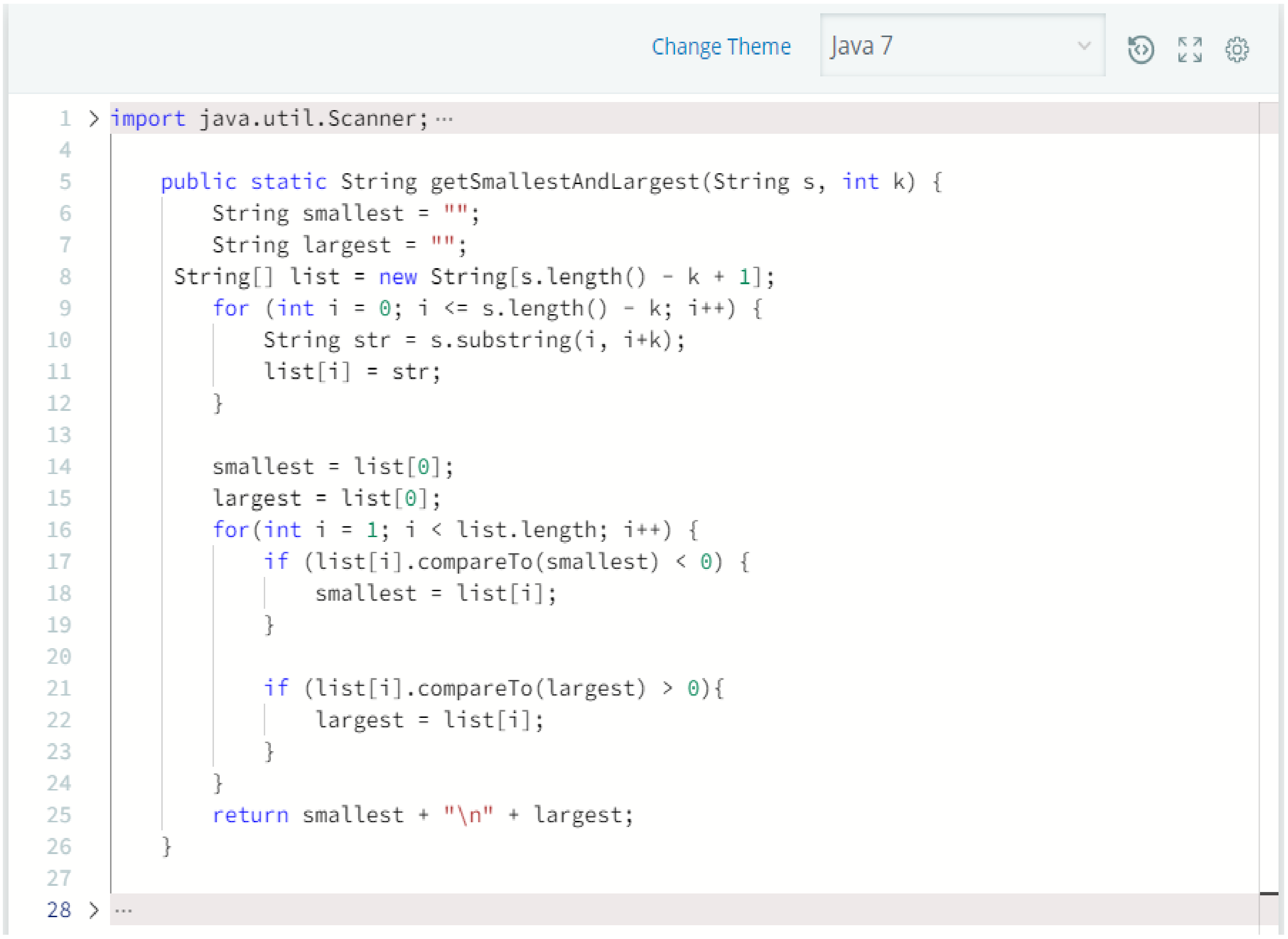
String $s = \mathtt{"welcometojava"}$ has the following lexicographically-ordered substrings of length $k = 3$:

$$\mathtt{["ava", "com", "elc", "eto", "jav", "lco", "met", "oja", "ome", "toj", "wel"]}$$

We then return the first (lexicographically smallest) substring and the last (lexicographically largest) substring as two newline-separated values (i.e., ava\nwel).

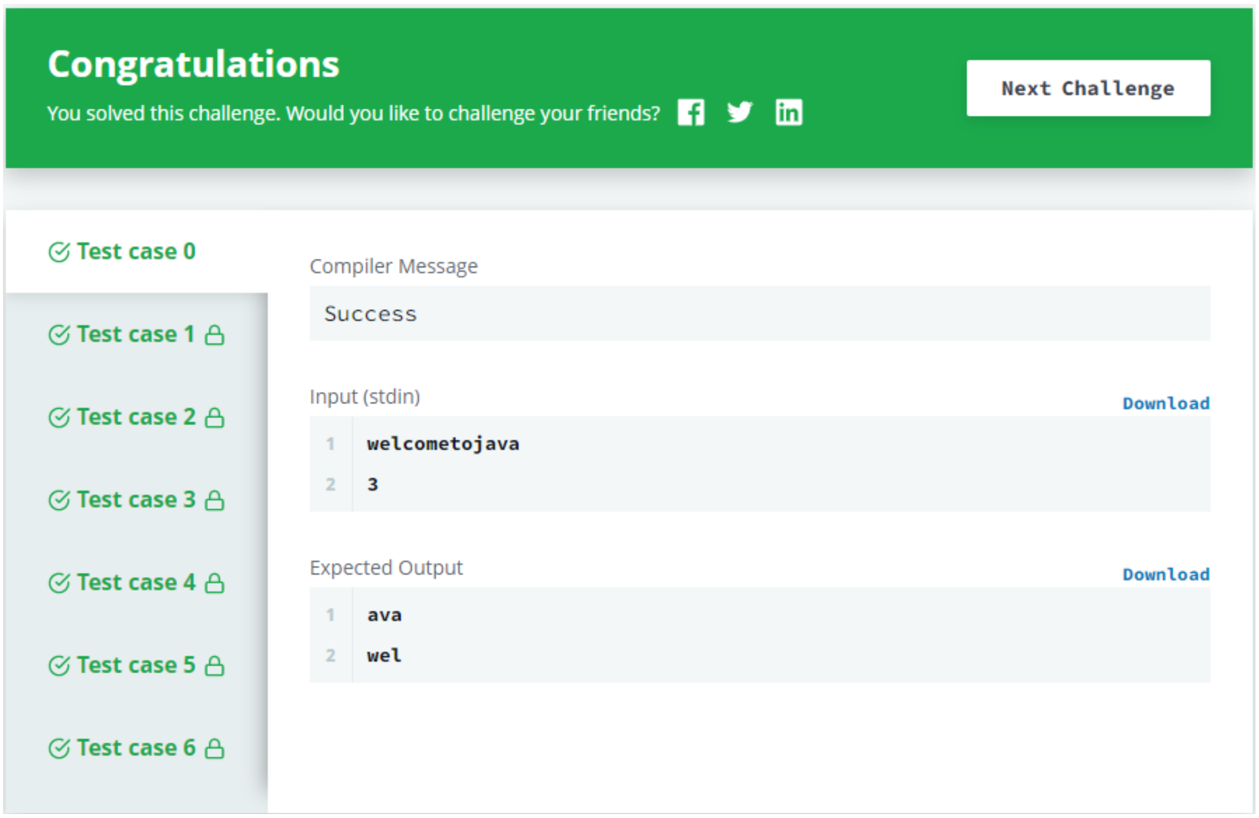
The stub code in the editor then prints ava as our first line of output and wel as our second line o output.

Solution:


Change Theme
Java 7

```java
import java.util.Scanner;···

    public static String getSmallestAndLargest(String s, int k) {
        String smallest = "";
        String largest = "";
     String[] list = new String[s.length() - k + 1];
        for (int i = 0; i <= s.length() - k; i++) {
            String str = s.substring(i, i+k);
            list[i] = str;
        }

        smallest = list[0];
        largest = list[0];
        for(int i = 1; i < list.length; i++) {
            if (list[i].compareTo(smallest) < 0) {
                smallest = list[i];
            }

            if (list[i].compareTo(largest) > 0){
                largest = list[i];
            }
        }
        return smallest + "\n" + largest;
    }

···
```

Output:



Congratulations

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

Compiler Message

```
Success
```

Input (stdin)

```
welcometojava
3
```

Expected Output

```
ava
wel
```

Question 1: isPalindrome

Write a method to determine whether the input string is a Palindrome or not.

What is a Palindrome?

A palindrome is a string that spells the same from either directions, for example - abba, appa, amma, malayalam, nayan, deed, level, madam, rotator, reviver, stats, tenet, ...

The prototype of the method should be -

public static void isPalindrome(String input1);

where **input1** is the string to be checked for being palindrome.

If the input string is a palindrome, the method should assign **true** to the **output1** member.

If the input string is NOT a palindrome, the method should assign **false** to the **output1** member.

NOTE: The case of the letters in the string should not matter, i.e. Madam , MAdam , madAM , madam , MADAM , should all be considered a palindrome.

ASSUMPTIONS: Within the scope of this assessment, you can assume the following, and so you do not have to write code to handle the below conditions -

1. The passed input string will always be a single word and not a sentence

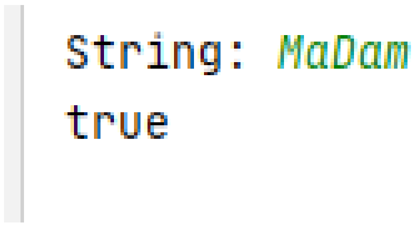
The passed input string will only contain alphabets.

Solution:

```
import java.util.*;
public class palindrome {

  public static void isPalindrome(String input1)
  {
    String b = "";
    int n = input1.length();
    for(int i = n - 1; i >= 0; i--)
    {
      b = b + input1.charAt(i);
    }
    if(input1.equalsIgnoreCase(b))
```

```
        {

          System.out.println("true");
        }
        else
        {
          System.out.println("false");
        }

      }
        public static void main(String args[])
        {
          String a;
          Scanner sc = new Scanner(System.in);
          System.out.print("String: ");
          a = sc.nextLine();
          isPalindrome(a);
        }
    }
```

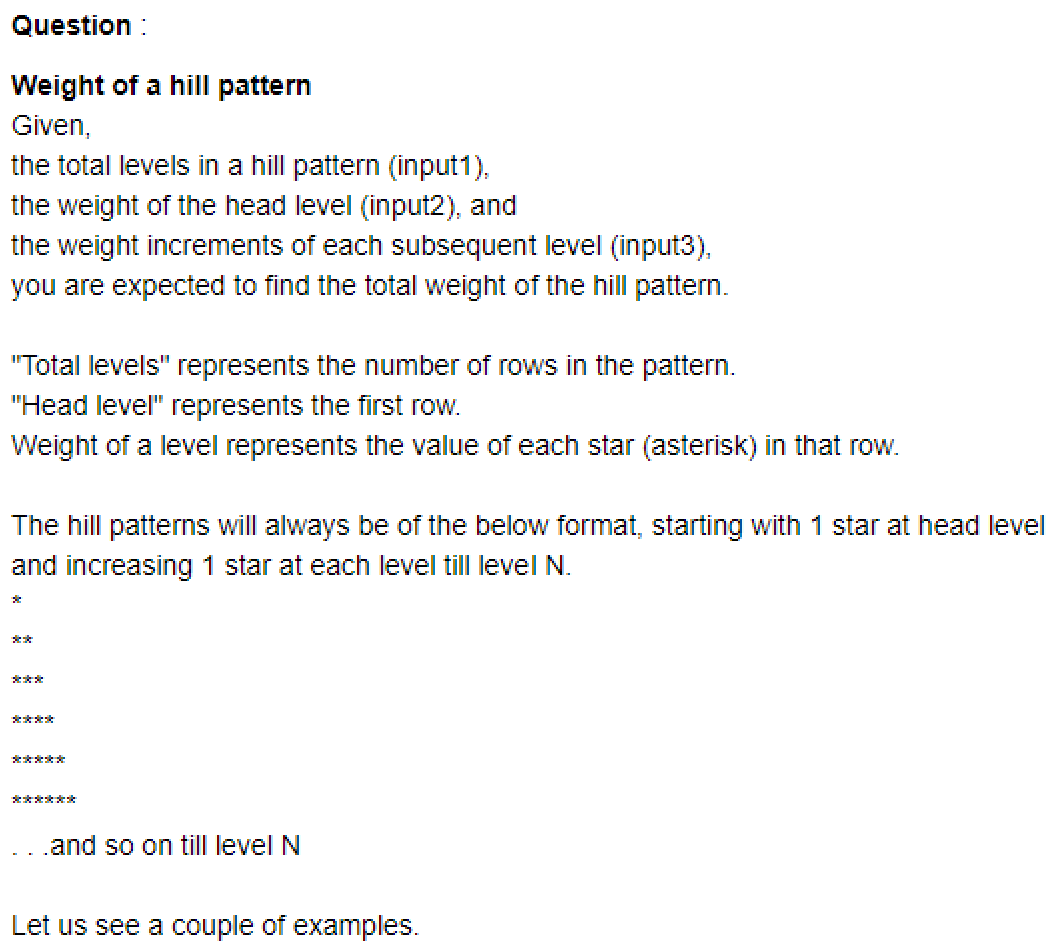
Output:



```
String: MaDam
true
```

Question 2: Mettl Problem

Mettl Problem: https://tests.mettl.com/authenticateKey/d612c0e6



Question :

Weight of a hill pattern
Given,
the total levels in a hill pattern (input1),
the weight of the head level (input2), and
the weight increments of each subsequent level (input3),
you are expected to find the total weight of the hill pattern.

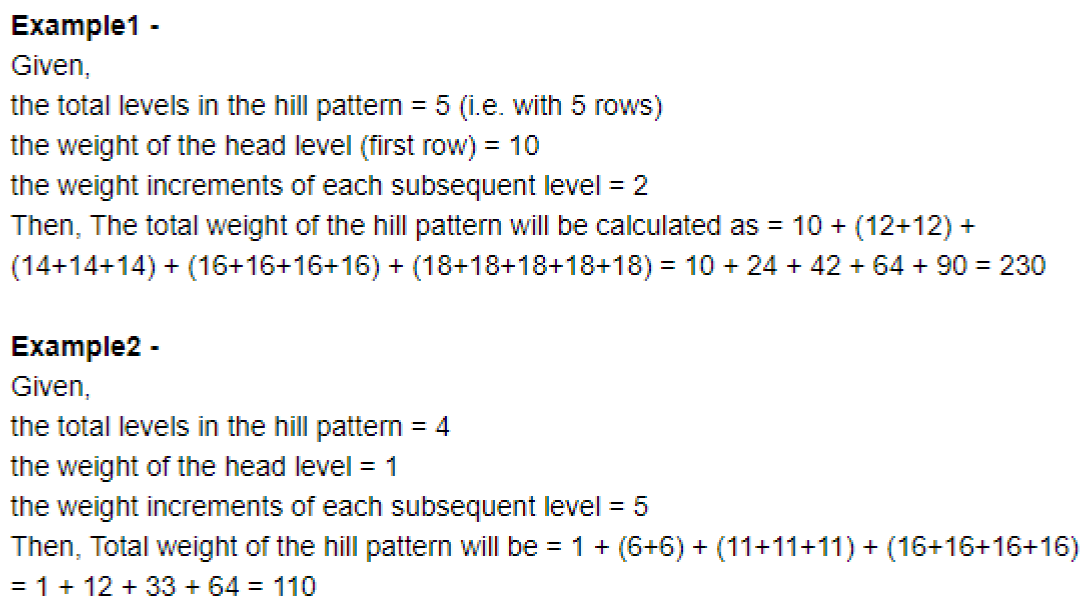
"Total levels" represents the number of rows in the pattern.
"Head level" represents the first row.
Weight of a level represents the value of each star (asterisk) in that row.

The hill patterns will always be of the below format, starting with 1 star at head level and increasing 1 star at each level till level N.

```
*
**
***
****
*****
******
```

. . .and so on till level N

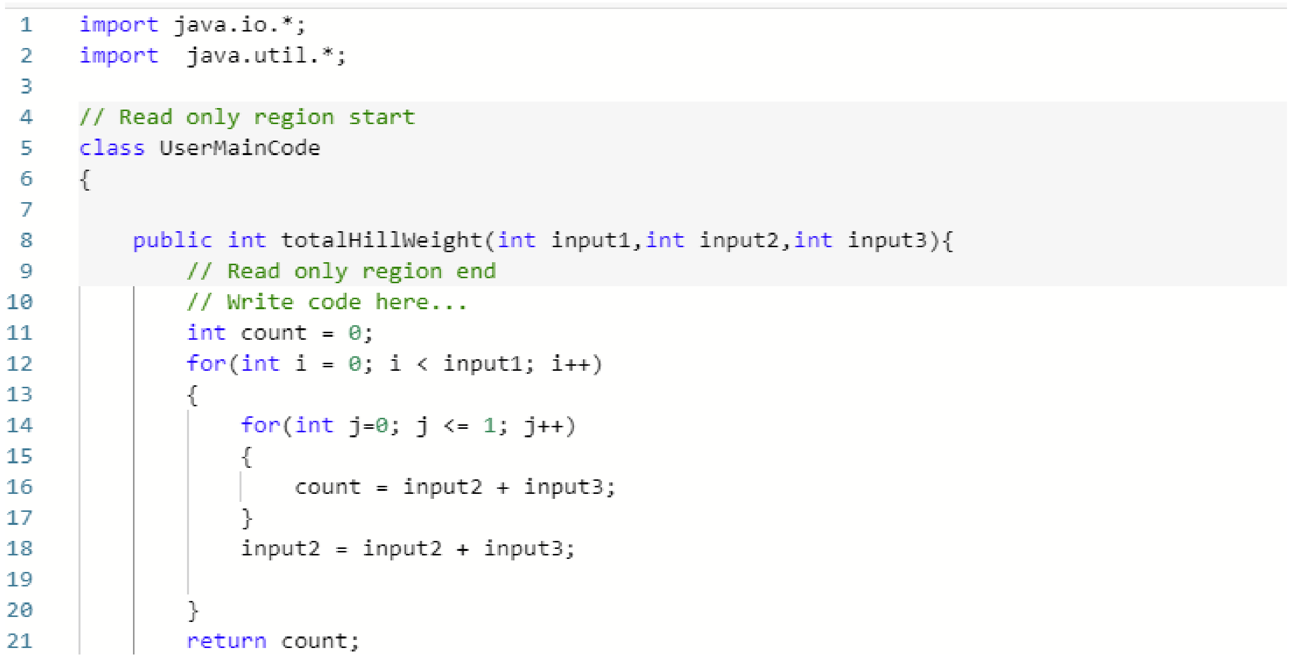
Let us see a couple of examples.



Example1 -
Given,
the total levels in the hill pattern = 5 (i.e. with 5 rows)
the weight of the head level (first row) = 10
the weight increments of each subsequent level = 2
Then, The total weight of the hill pattern will be calculated as = 10 + (12+12) + (14+14+14) + (16+16+16+16) + (18+18+18+18+18) = 10 + 24 + 42 + 64 + 90 = 230

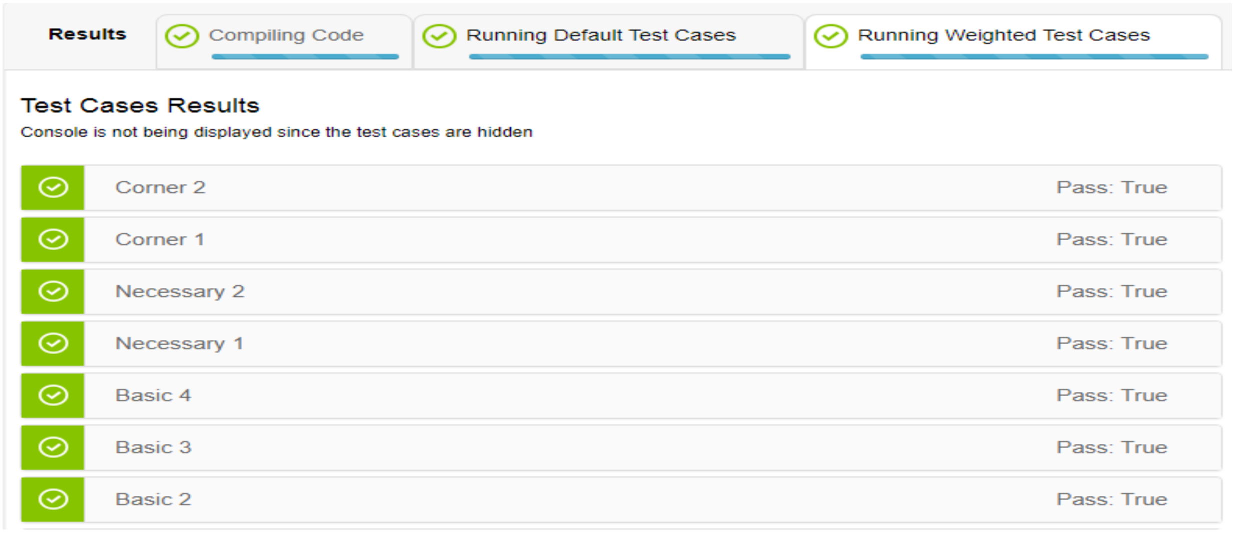
Example2 -
Given,
the total levels in the hill pattern = 4
the weight of the head level = 1
the weight increments of each subsequent level = 5
Then, Total weight of the hill pattern will be = 1 + (6+6) + (11+11+11) + (16+16+16+16) = 1 + 12 + 33 + 64 = 110

Solution:



```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

    public int totalHillWeight(int input1,int input2,int input3){
        // Read only region end
        // Write code here...
        int count = 0;
        for(int i = 0; i < input1; i++)
        {
            for(int j=0; j <= 1; j++)
            {
                count = input2 + input3;
            }
            input2 = input2 + input3;

        }
        return count;
```

Output:

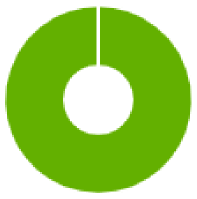


Test Cases Results

Console is not being displayed since the test cases are hidden

✓	Corner 2	Pass: True
✓	Corner 1	Pass: True
✓	Necessary 2	Pass: True
✓	Necessary 1	Pass: True
✓	Basic 4	Pass: True
✓	Basic 3	Pass: True
✓	Basic 2	Pass: True

SUMMARY OF ATTEMPTS



1 Correct
(Scored 40/40)

Question 3: Given an array of strings, group anagrams together.

Example:

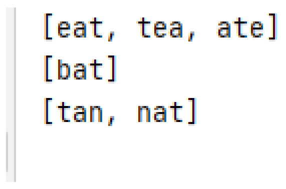
Input: ["eat", "tea", "tan", "ate", "nat", "bat"],
Output:
```
[
  ["ate","eat","tea"],
  ["nat","tan"],
  ["bat"]
]
```
Note:

All inputs will be in lowercase.
The order of your output does not matter.

Solution:

```
import java.util.*;
public class GroupAnagrams {
   public static List<List<String>> groupAnagrams1(String[] str) {
      if (str.length == 0) {
         return new ArrayList();
      }
      Map<String, List> ans = new HashMap<String, List>();
      for (String s : str) {
         char[] ca = s.toCharArray();
         Arrays.sort(ca);
         String key = String.valueOf(ca);
         if (!ans.containsKey(key)) {
            ans.put(key, new ArrayList());
         }
         ans.get(key).add(s);
      }
      return new ArrayList(ans.values());
   }
   public static void main(String args[]) {
      String[] data = {"eat", "tea", "tan", "ate", "nat", "bat"};
      List<List<String>> list = groupAnagrams1(data);
      for (List<String> f : list) {
         System.out.println(f);
      }
   }
}
```

Output:



```
[eat, tea, ate]
[bat]
[tan, nat]
```

Question 4: Java Inheritance II

Write the following code in your editor below:

1. A class named *Arithmetic* with a method named *add* that takes 2 integers as parameters and returns an integer denoting their sum.
2. A class named *Adder* that inherits from a superclass named *Arithmetic*.

Your classes should not be be **public**.

Input Format

You are not responsible for reading any input from stdin; a locked code stub will test your submission by calling the *add* method on an *Adder* object and passing it 2 integer parameters.

Output Format

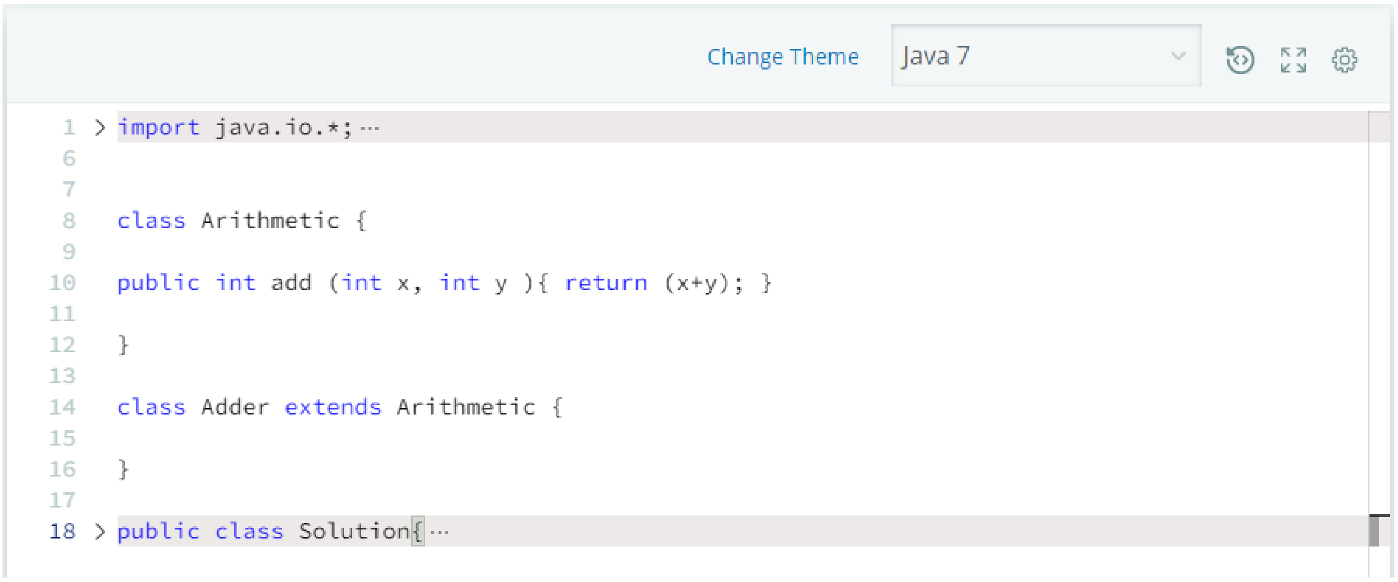
You are not responsible for printing anything to stdout. Your *add* method must return the sum of its parameters.

Sample Output

The *main* method in the *Solution* class above should print the following:

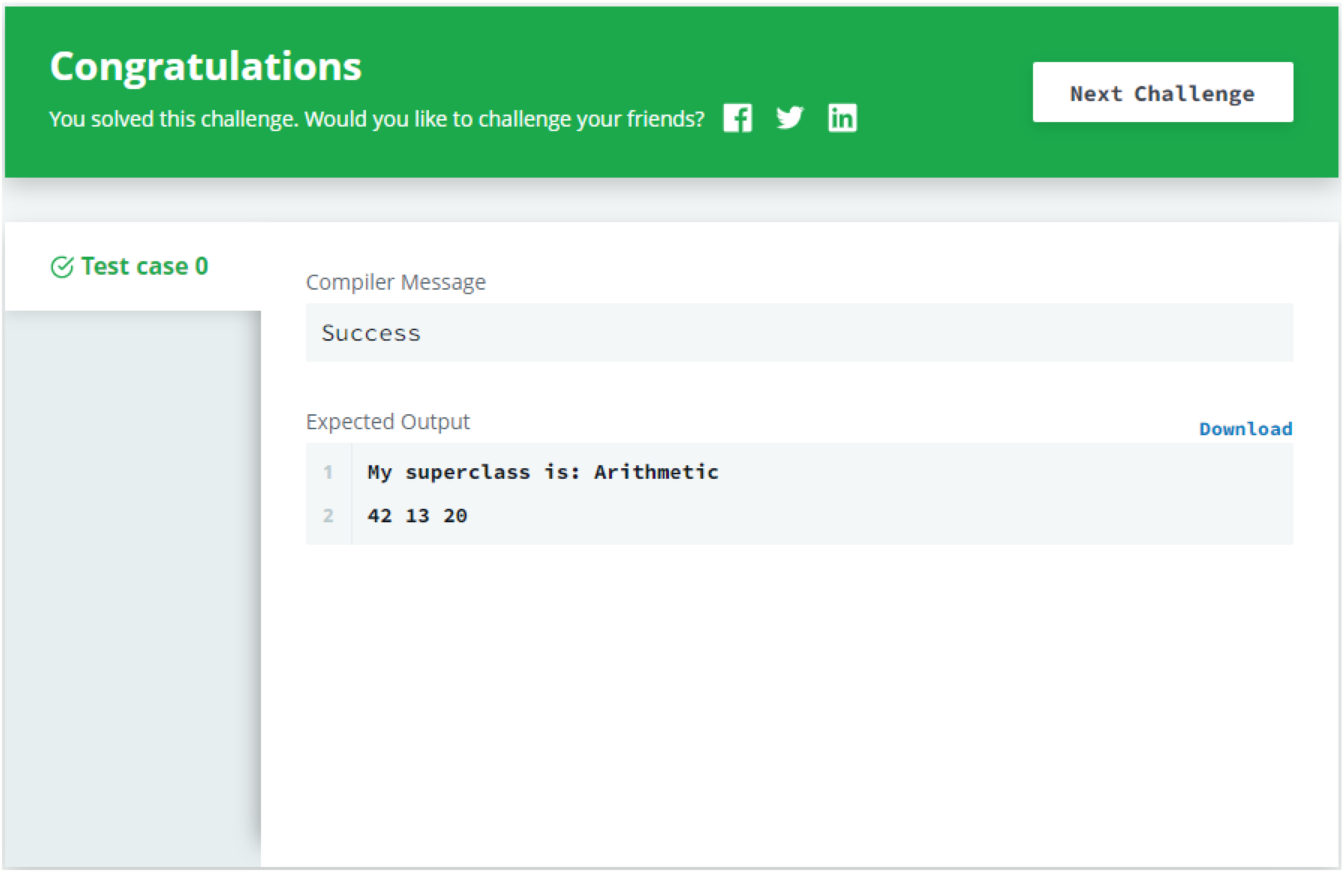
```
My superclass is: Arithmetic
42 13 20
```

Solution:



```java
import java.io.*;...


class Arithmetic {

public int add (int x, int y ){ return (x+y); }

}

class Adder extends Arithmetic {

}

public class Solution{...
```

Output:


Congratulations
You solved this challenge. Would you like to challenge your friends?
Next Challenge
Test case 0
Compiler Message
Download

Success

Expected Output

```
My superclass is: Arithmetic
42 13 20
```

Question 5: Java Instanceof keyword

The Java *instanceof* operator is used to test if the object or instance is an instanceof the specified type.

In this problem we have given you three classes in the editor:

- Student class
- Rockstar class
- Hacker class

In the main method, we populated an *ArrayList* with several instances of these classes. *count* method calculates how many instances of each type is present in the ArrayList. The code prints three integers, the number of instance of Student class, the number of instance of Rockstar class, the number of instance of Hacker class.

But some lines of the code are missing, and you have to fix it by modifying only **3** lines! Don't add, delete or modify any extra line.

To restore the original code in the editor, click on the top left icon in the editor and create a new buffer.

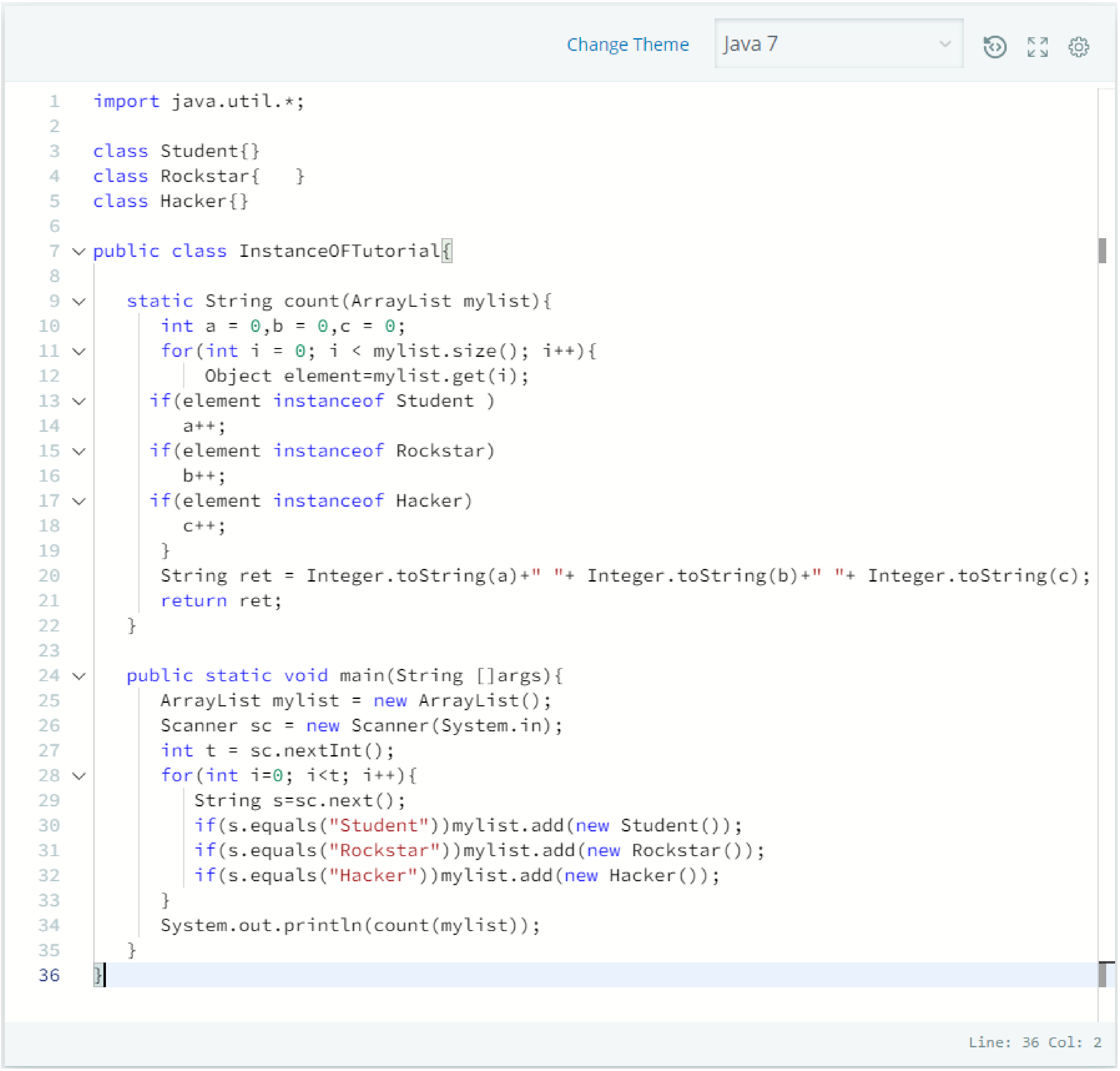
Sample Input

```
5
Student
Student
Rockstar
Student
Hacker
```

Sample Output

```
3 1 1
```

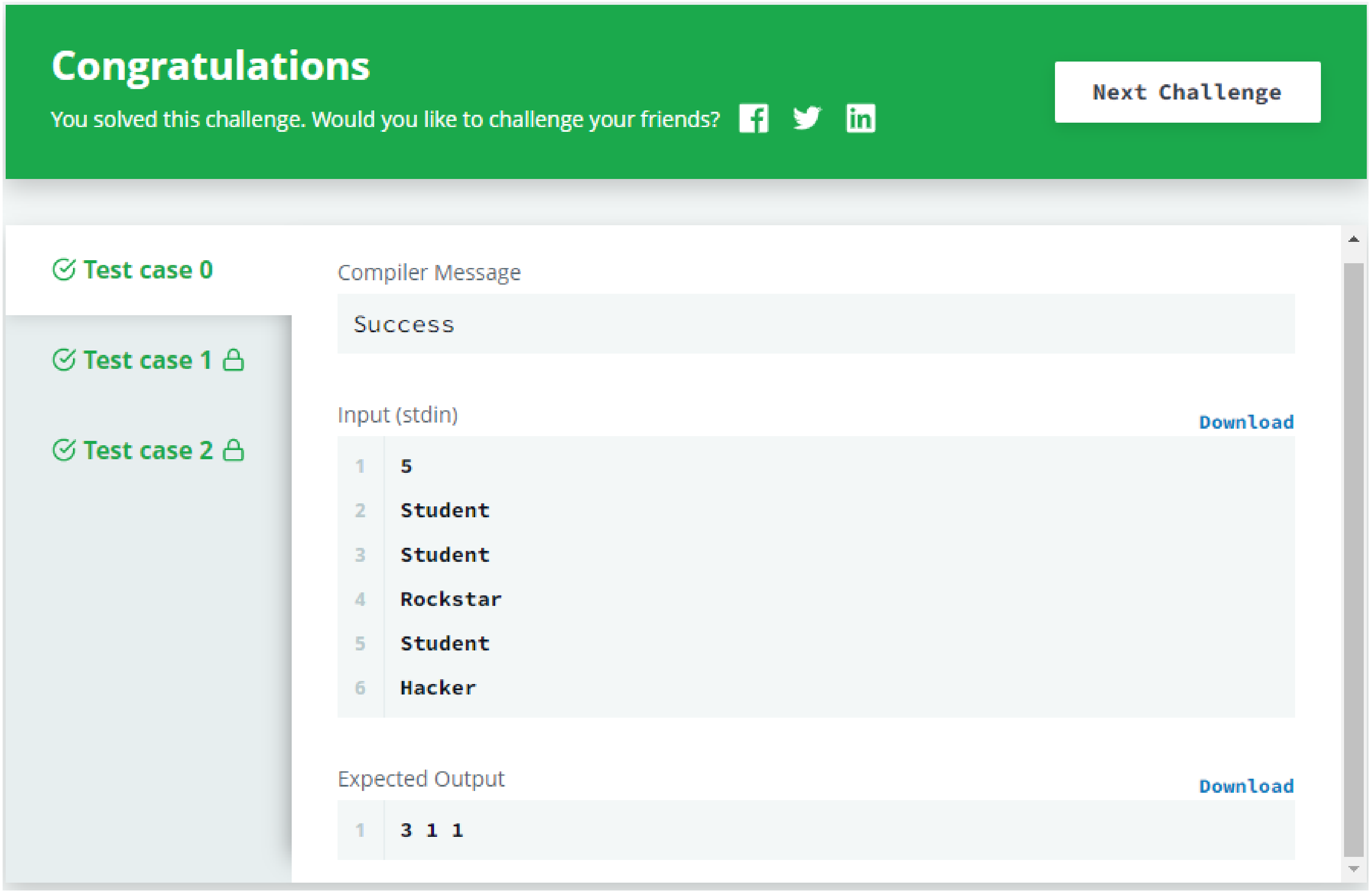
Solution:


Change Theme Java 7

```java
import java.util.*;

class Student{}
class Rockstar{   }
class Hacker{}

public class InstanceOFTutorial{

   static String count(ArrayList mylist){
      int a = 0,b = 0,c = 0;
      for(int i = 0; i < mylist.size(); i++){
         Object element=mylist.get(i);
     if(element instanceof Student )
        a++;
     if(element instanceof Rockstar)
        b++;
     if(element instanceof Hacker)
        c++;
      }
      String ret = Integer.toString(a)+" "+ Integer.toString(b)+" "+ Integer.toString(c);
      return ret;
   }

   public static void main(String []args){
      ArrayList mylist = new ArrayList();
      Scanner sc = new Scanner(System.in);
      int t = sc.nextInt();
      for(int i=0; i<t; i++){
         String s=sc.next();
         if(s.equals("Student"))mylist.add(new Student());
         if(s.equals("Rockstar"))mylist.add(new Rockstar());
         if(s.equals("Hacker"))mylist.add(new Hacker());
      }
      System.out.println(count(mylist));
   }
}
```

Line: 36 Col: 2

Output:



Congratulations

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

Next Challenge

✓ Test case 0

✓ Test case 1

✓ Test case 2

Compiler Message

```
Success
```

Input (stdin)

Download

```
5
Student
Student
Rockstar
Student
Hacker
```

Expected Output

Download

```
3 1 1
```