2413 min left

5. Fix the code to find sum of the factorial of all odd numbers - 2

Previous | Next

John has an array `arr` of size `n`. He wants to find the sum of the factorial of all odd numbers in `arr` that are less than 10.

John has written a function `buggySumOfOddFactorialsLessThanTen` that accepts `n` and stores `arr` as a list of integers in its arguments. However, there are some bugs in the function logic. Fix the issues to help John complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
4 1 2 3 4	7	Here the odd numbers less than 10 are 1 and 3. Hence, 1! + 3! = 1 + 3*2*1 = 7 is the answer.
5 5 7 9 11 13	368040	Here the odd numbers less than 10 are 5,7 and 9. Hence, 5! + 7! + 9! = 368040 is the answer.
5		

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggySumOfOddFactorialsLessThanTen(int n, List<Integer> arr) {
        // Fix the code here
        int sum = 0;
        for(int i = 0; i < n; i++){
            if(arr.get(i) < 10 && arr.get(i) % 2 != 0)
                sum += factorial(arr.get(i));
        }
        return sum;
    }

    public static int factorial(int n){
        return (n == 1 || n == 0) ? 1 : factorial(n - 1) * n;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<Integer> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(Integer.parseInt(scan.nextLine().trim()));
        }

        int result = buggySumOfOddFactorialsLessThanTen(n, arr);

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

Console | Custom Test Case | Run code | Submit code

All Test cases passed

2413 min left

4. Fix the code to find product of prime numbers in a list - 2

Previous | Next

Sam has an array `arr` containing `n` integers. He wants to calculate the product of all prime numbers in `arr` that are greater than 10. Therefore, he has written a function `buggyProductOfPrimesGreaterThanTen` that accepts `n` and stores `arr` as a list of integers as its arguments.

However, there are some issues in the code for this function. Fix the issues in `buggyProductOfPrimesGreaterThanTen` to help Sam complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
5 11 12 13 14 15	143	The two prime numbers in arr are 11 and 13. Hence, 11 * 13 = 143 is the answer.
7 10 11 12 13 14 15	143	The two prime numbers in arr are 11 and 13. Hence, 11 * 13 = 143 is the answer.

18/18 Answered

Submit

Java 8

Reset code A+ A-

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


public class Solution {
    static boolean isPrime(int num) {
        if (num <= 1) return false;
        if (num <= 3) return true;
        if (num % 2 == 0 || num % 3 == 0) return false;
        int i = 5;
        while (i * i <= num) {
            if (num % i == 0 || num % (i + 2) == 0) return false;
            i += 6;
        }
        return true;
    }

    public static int buggyProductOfPrimesGreaterThanTen(int n, List<Integer> arr) {
        // Fix the code here
        int product = 1;
        for(int num : arr) {
            if (num > 10 && isPrime(num)) {
                product *= num;
            }
        }
        return product;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<Integer> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(Integer.parseInt(scan.nextLine().trim()));
        }
```

Console | Custom Test Case

Run code | Submit code

All Test cases passed

2413 min left

3. Fix the code to determine if a given string is a valid phone number - 2

Previous | Next

You are given a partially implemented code to determine if a given string is a valid phone number. The code reads a string and calls the function `buggyIsValidPhoneNumber` to check if the string is a valid phone number.

However, the function contains a bug that causes it to produce incorrect results. Your task is to find and fix the bug in the `buggyIsValidPhoneNumber` function.

Notes:
A valid phone number should meet the following requirements:

1. Contain exactly 10 digits, length of the input can be longer if parentheses and/or dashes included.
2. Contain only digits (0-9), optional parentheses, and optional dashes.
3. If parentheses are included, they should appear around the first three digits (e.g., "(123)456-7890").
4. If dashes are included, they should appear either between the fourth and fifth digits or after the seventh digit (e.g., "123-4567890" or "123456-7890").

Thus, if the given phone number is "1234567890", the output should be true. If the given phone number is "(123)456-7890", the output should also be true. However, if the given phone number is "123-4567-890" or "(123-456)-7890", the output should be false because the parentheses and dashes are not in the correct positions.

Function description

Constraints

Input format for debugging

Sample Testcases

18/18 Answered

Submit

Java 8 | Reset code | A+ | A-

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


public class Solution {
    public static boolean buggyIsValidPhoneNumber(String s) {
        // Fix the code here
        if (s.length() == 10) {
            return s.matches("\\d+");
        } else if (s.length() == 13) {
            return s.charAt(0) == '(' && s.charAt(4) == ')' && s.charAt(8) == '-' &&
                   s.substring(1,4).matches("\\d+") &&
                   s.substring(5,8).matches("\\d+") &&
                   s.substring(9,13).matches("\\d+");
        } else if (s.length() == 11 && s.charAt(3) == '-') {
            return s.substring(0,3).matches("\\d+") &&
                   s.substring(4,11).matches("\\d+");
        } else if (s.length() == 11 && s.charAt(6) == '-') {
            return s.substring(0,6).matches("\\d+") &&
                   s.substring(7,11).matches("\\d+");
        }
        return false;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String s = scan.nextLine();

        boolean result = buggyIsValidPhoneNumber(s);

        System.out.println(result? "1": "0");
    }
}
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

2413 min left

2. Fix the code to find the number of times a target substring appears in a given string - 2

Previous | Next

Jack has two strings, `str` and `t`. He wants to find the total number of non-overlapping occurrences of string `t` in `str`.

Jack has written a function `buggyCountNonOverlappingSubstringOccurrences` that accepts `str` and `t` in its arguments. However, there are some bugs in the function logic. Fix the issues in `buggyCountNonOverlappingSubstringOccurrences` to help Jack complete his task.

Notes:

- Let `str = "ababa"` and `t = "aba"`. If we only count non-overlapping occurrences, then "aba" appears only once in "ababa". This is because, after the first occurrence is found at the start of the string, the search continues from the character after the last character of the found occurrence. Thus, after finding the first "aba", the search would continue from the second "b" in "ababa", and no further occurrences of "aba" would be found. Therefore, for `str = "ababa"` and `t = "aba"`, `buggyCountNonOverlappingSubstringOccurrences(str, t)` should return 1.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
ababa	1	"aba" has 1 non-overlapping

18/18 Answered

Submit

Java 8

Reset code A+ A-

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


public class Solution {
    public static int buggyCountNonOverlappingSubstringOccurrences(String str, String t) {
        // Fix the code here
        if (t.isEmpty()) {
            return 0;
        }

        int count = 0;
        int start = 0;
        while ((start = str.indexOf(t, start)) != -1) {
            count++;
            start += t.length();
        }

        return count;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String str = scan.nextLine();

        String t = scan.nextLine();

        int result = buggyCountNonOverlappingSubstringOccurrences(str, t);

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

Console | Custom Test Case

Run code | Submit code

All Test cases passed

2414 min left

1. Fix the code to determine if a given string is a valid credit card number - 2

Previous | Next

You are given a partially implemented code to determine if a given string is a valid credit card number. The code reads a string and calls the function `buggyIsValidCreditCard` to check if the string is a valid credit card number.

However, the function contains a bug that causes it to produce incorrect results.

A valid credit card number should:

1. Contain exactly 16 digits.
2. Start with 4, 5, or 6.
3. Contain only digits (0-9).
4. Pass the Luhn algorithm (a simple checksum formula used to validate a variety of identification numbers, such as credit card numbers).

The Luhn algorithm works as follows:

1. Double the value of every second digit from the rightmost digit.
2. If doubling a number results in a two-digit number, add the two digits to get a single-digit number.
3. Take the sum of all the single-digit numbers obtained in the previous steps.
4. If the total modulo 10 is equal to 0 (if the total ends in zero), the number is valid according to the Luhn formula; otherwise, it is not valid.

For example, if the input string is "4532015112830366", the output should be true, but if the input string is "1234567812345678", the output should be false because it does not pass the Luhn algorithm or start with 4, 5, or 6.

Your task is to find and fix the bug in the `buggyIsValidCreditCard` function.

Function description

Constraints

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static boolean buggyIsValidCreditCard(String cardNumber) {
        // Fix the code here
        if (cardNumber.length() != 16) {
            return false;
        }

        char startDigit = cardNumber.charAt(0);
        if (startDigit != '4' && startDigit != '5' && startDigit != '6') {
            return false;
        }

        for (char c : cardNumber.toCharArray()) {
            if (!Character.isDigit(c)) {
                return false;
            }
        }

        // Luhn algorithm
        int sum = 0;
        for (int i = cardNumber.length() - 1; i >= 0; i--) {
            int digit = cardNumber.charAt(i) - '0';
            if (i % 2 == cardNumber.length() % 2) {
                digit *= 2;
                if (digit > 9) {
                    digit -= 9;
                }
            }
            sum += digit;
        }

        return sum % 10 == 0;
    }
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

2414 min left

8. Fix the code to determine if a given string contains all unique characters - 2

Previous | Next

You are given a partially implemented code to determine if a given string contains all unique characters, excluding whitespace. The code reads a string and calls the function `buggyHasUniqueNonWhitespaceCharacters` to check if the string, after removing all whitespace, has all unique characters.

However, the function contains a bug that causes it to produce incorrect results. Your task is to find and fix the bug in the `buggyHasUniqueNonWhitespaceCharacters` function.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
hello world	0	The answer is False, since 'l' and 'o' repeated. Hence we return 0.
123 456 789	1	The answer is True, since all characters are unique. Hence we return 1.
a b c d e f	1	The answer is True, since all characters are unique. Hence the answer is 1.

18/18 Answered

Submit

Java 8

Reset code A+ A-

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


public class Solution {
    public static boolean buggyHasUniqueNonWhitespaceCharacters(String str) {
        // Fix the code here
        str = str.replaceAll(" ", "");
        HashMap<Character, Integer> charCount = new HashMap<>();
        for (char c : str.toCharArray()) {
            if (charCount.containsKey(c)) {
                return false;
            }
            charCount.put(c, 1);
        }
        return true;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String str = scan.nextLine();

        boolean result = buggyHasUniqueNonWhitespaceCharacters(str);

        System.out.println(result? "1": "0");
    }
}
```

Console | Custom Test Case

Run code | Submit code

All Test cases passed

2414 min left

7. Fix the code to find number of words in a sentence that start with a specified character - 2

Previous | Next

Jill has a string `str`. She wants to find the number of words in `str` that both start and end with a specified character `c`.

Jill has written a function `buggyCountWordsStartingAndEndingWithCharacter` that accepts `str` and `c` in its arguments. However, there are some bugs in the function logic. Fix the issues in `buggyCountWordsStartingAndEndingWithCharacter` to help Jill complete her task.

Notes:

- It is guaranteed that all inputs are provided in lowercase.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
oreo world h	0	No word starts and end with 'h'. Hence answer is 0.
cat bat matm cat m	1	matm starts and ends with 'm'. Thus, answer is 1.
apple applea apple a	1	applea starts and ends with 'a'. Hence answer is 1.

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggyCountWordsStartingAndEndingWithCharacter(String str, char c) {
        // Fix the code here
        int count = 0;
        String[] words = str.split(" ");
        for(String word : words) {
            if(word.startsWith(Character.toString(c)) && word.endsWith(Character.toString(c))) {
                count++;
            }
        }
        return count;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String str = scan.nextLine();

        char c = scan.nextLine().charAt(0);

        int result = buggyCountWordsStartingAndEndingWithCharacter(str, c);

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

Console | Custom Test Case | Run code | Submit code

All Test cases passed

2414 min left

6. Fix the code to determine if a given string is a valid email address - 2

Previous | Next

You are given a partially implemented code to determine if a given string is a valid email address. The code reads a string and calls the function `buggyIsValidEmail` to check if the string is a valid email address.

However, the function contains an issue. Your task is to find and fix the bug in the `buggyIsValidEmail` function.

Notes:
A valid email address for this task should:

1. Consist of a local part followed by an "@" symbol, and a domain part.
2. The local part may contain alphanumeric characters (both uppercase and lowercase), periods (.), and plus signs (+), but it should not start or end with a period or plus sign.
3. The domain part may contain alphanumeric characters (both uppercase and lowercase) and hyphens (-), but it must start and end with an alphanumeric character.
4. The domain part should also contain at least one period (.) symbol, indicating a domain suffix like .com, .org, etc.

Thus, if the input string is "john.doe@example.com", the output should be true, but if the input string is "john..doe@example.com" or ".john@example.com", the output should be false because two consecutive periods are not allowed and the local part shouldn't start with a period.

Function description

Constraints

Input format for debugging

Sample Testcases

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static boolean buggyIsValidEmail(String emailStr) {
        int atPos = emailStr.indexOf('@');

        if (atPos <= 0|| atPos==emailStr.length()-1) {
            return false;
        }

        String local = emailStr.substring(0, atPos);
        String domain = emailStr.substring(atPos + 1);

        if (local.charAt(0) == '.' || local.charAt(0) == '+' || local.charAt(local.length() - 1)
        == '.' || local.charAt(local.length() - 1) == '+' || local.contains("..")) {
            return false;
        }

        if (domain.charAt(0) == '-' || domain.charAt(0) == '.' || domain.charAt(domain.length() - 1)
        == '-' || domain.charAt(domain.length() - 1) == '.' || domain.contains("..")) {
            return false;
        }

        if (!domain.contains(".")) {
            return false;
        }

        return true;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String emailStr = scan.nextLine();
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

2414 min left

5. Fix the code to count the number of strings in a list that have a specified length - 2

Previous | Next

Jake has a string array `arr` of size `n`, an integer value `length`, and a character `targetChar`. He wants to count the total number of strings in `arr` whose length is equal to the given integer value and that also start with the `targetChar`.

Jake has written a function `buggyCountStringsOfTargetLengthAndChar` that accepts `arr`, `length`, and `targetChar` in its arguments. However, there are some bugs in the function logic.

Your task is to fix the issues in `buggyCountStringsOfTargetLengthAndChar` to help Jake complete his task.

- Function description
- Constraints
- Input format for debugging
- Sample Testcases

Input	Output	Output Description
4 apple pear banana apricot 5 a	1	Only apple starts with 'a' and has length of 5.
3 cat dog		cat and cow both start with 'c' and

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggyCountStringsOfTargetLengthAndChar(int n, List<String> arr, int length, char tar
        // Fix the code here
        int count = 0;
        for (int i = 0; i < n; i++) {
            if (arr.get(i).length() == length && arr.get(i).charAt(0) == targetChar) {
                count++;
            }
        }
        return count;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<String> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(scan.nextLine());
        }

        int length = Integer.parseInt(scan.nextLine().trim());

        char targetChar = scan.nextLine().charAt(0);

        int result = buggyCountStringsOfTargetLengthAndChar(n, arr, length, targetChar);

        System.out.println(result);
    }

```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

2415 min left

4. Fix the code to find sum of squares of all odd numbers - 2

Previous | Next

John has an array `arr` of size `n`. He wants to find the sum of squares of all odd numbers in `arr` that are also perfect squares.

John has written a function `buggySumOfOddPerfectSquareSquares` that accepts `n` and `arr` as a list of integers in its arguments. However, there are some bugs in the function logic.

Your task is to fix the issues in `buggySumOfOddPerfectSquareSquares` to help John complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
5 1 4 9 16 25	707	The answer here is 9*9 + 25*25 + 1*1 = 707.
5 2 4 6 8 10	0	There are no odd numbers in the array. Hence, the answer is 0.
5 1		

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggySumOfOddPerfectSquareSquares(int n, List<Integer> arr) {
        // Fix the code here
        int sum = 0;
        for (int i = 0; i < n; i++) {
            int root = (int) Math.sqrt(arr.get(i));
            if (arr.get(i) % 2 != 0 && root * root == arr.get(i)) {
                sum += arr.get(i) * arr.get(i);
            }
        }
        return sum;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<Integer> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(Integer.parseInt(scan.nextLine().trim()));
        }

        int result = buggySumOfOddPerfectSquareSquares(n, arr);

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

Console | Custom Test Case | Run code | Submit code

All Test cases passed

2415 min left

3. Fix the code to check if a given string is a palindrome - 2

Previous | Next

You are given a partially implemented code to check if a given string is a specific type of palindrome known as a **"mirrored string"**. If the given string is a **mirrored string**, return 1; else return 0.

A **mirrored string** is a string that appears the same when looked at in a mirror (ignoring spaces, punctuation, and capitalization). This means that each character in the string must be symmetrical along the vertical axis and the string should read the same backwards as forwards. For instance, the string "Aibohphobia" is a mirrored string as each character appears the same in a mirror and the string reads the same backwards.

The code reads a string and calls the function `buggyIsMirroredString` to determine if the string is a **mirrored string**.

However, the function contains a bug that causes it to produce incorrect results. Your task is to find and fix the bug in the `buggyIsMirroredString` function.

18/18 Answered

- Function description
- Constraints
- Input format for debugging
- Sample Testcases

Input	Output	Output Description
Aibohphobia	1	This is a mirrored string.
WOW	1	This is a mirrored string.
Hello	0	This is not a mirrored string.

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggyIsMirroredString(String str) {
        // Fix the code here
        String mirrored_chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        str = str.toUpperCase();
        return (str.equals(new StringBuilder(str).reverse().toString()) &&
                str.chars().allMatch(ch -> mirrored_chars.indexOf(ch) != -1)) ? 1 : 0;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String str = scan.nextLine();

        int result = buggyIsMirroredString(str);

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

Console | Custom Test Case | Run code | Submit code

All Test cases passed

Test Case #1	Accepted	memory 35488kb	runtime 0.220s
Test Case #2	Accepted	memory 35816kb	runtime 0.222s
Test Case #3	Accepted	memory 28884kb	runtime 0.113s
Test Case #4	Accepted	memory 35792kb	runtime 0.211s
Test Case #5	Accepted	memory 35616kb	runtime 0.210s
Test Case #6	Accepted	memory 35188kb	runtime 0.169s

2415 min left

2. Fix the code to determine if a given string is a valid URL - 2

Previous | Next

You are given a partially implemented code to determine if a given string is a valid URL with a specific domain suffix. The code reads a string and calls the function `buggyIsValidURLWithSpecificSuffix` to check if the string is a valid URL and ends with the domain suffix ".org".

However, the function contains a bug. Your task is to find and fix the bug in the `buggyIsValidURLWithSpecificSuffix` function.

Notes:
A valid URL should:

1. Start with a valid scheme (e.g., "http", "https", "ftp").
2. Followed by "://".
3. Then, a domain part, which may contain alphanumeric characters (both uppercase and lowercase), hyphens (-), and periods (.). The domain part must start and end with an alphanumeric character.
4. End with the domain suffix ".org".
5. Optionally, a path part after the domain, which may contain alphanumeric characters (both uppercase and lowercase), hyphens (-), slashes (/), and periods (.)

For example, if the input string is "https://www.example.org/test", the output should be true, but if the input string is "https://www.example.com/test" or "http:/www.example.org/test", the output should be false because the URL does not have a valid scheme or the correct domain suffix.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static boolean buggyIsValidURLWithSpecificSuffix(String url) {
        // Fix the code here
        String[] schemes = {"http://", "https://", "ftp://"};
        String[] parts = url.split("://");
        if (parts.length != 2 || !Arrays.asList(schemes).contains(parts[0]+"://")) {
            return false;
        }
        parts = parts[1].split("/");
        String domain = parts[0];
        if (!domain.endsWith(".org") || !Character.isLetterOrDigit(domain.charAt(0)) ||
         !Character.isLetterOrDigit(domain.charAt(domain.length() - 5))) {
            return false;
        }
        for (char c : domain.toCharArray()) {
            if (!Character.isLetterOrDigit(c) && c != '-' && c != '.') {
                return false;
            }
        }
        for (String part : Arrays.copyOfRange(parts, 1, parts.length)) {
            for (char c : part.toCharArray()) {
                if (!Character.isLetterOrDigit(c) && c != '-' && c != '/' && c != '.') {
                    return false;
                }
            }
        }
        return true;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String url = scan.nextLine();
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

2415 min left

1. Fix the code to find sum of squares of all even numbers - 2

Previous | Next

John has an array `arr` of size `n`. He wants to find the sum of squares of all even numbers in `arr` that are at odd-indexed positions. It is given that arr is 0-indexed.

John has written a function `buggySumOfEvenSquaresAtOddIndices` that accepts `n` and `arr` as a list of integers in its arguments. However, there are some bugs in the function logic.

Your task is to fix the issues in `buggySumOfEvenSquaresAtOddIndices` to help John complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
5 1 2 3 4 5	20	Only 4 (at index 3) is an even number at an odd-indexed position, 4^2 = 16, and 2 (at index 1) is also an even number at an odd-indexed position, 2^2 = 4, so total is 16 + 4 = 20.
3 2 4 6	16	Only 4 (at index 1) is an even number at an odd-indexed position.

18/18 Answered

Submit

Java 8

Reset code A+ A-

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


public class Solution {
    public static int buggySumOfEvenSquaresAtOddIndices(int n, List<Integer> arr) {
        // Fix the code here
        int sum = 0;
        for(int i = 1; i < n; i += 2) {
            if(arr.get(i) % 2 == 0) {
                sum += Math.pow(arr.get(i), 2);
            }
        }
        return sum;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<Integer> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(Integer.parseInt(scan.nextLine().trim()));
        }

        int result = buggySumOfEvenSquaresAtOddIndices(n, arr);

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

Console | Custom Test Case

Run code | Submit code

2415 min left

5. Fix the code to perform character replication in string - 2

Previous | Next

Bob has a string `S` of size `N`. He wants to create a transformed string by replicating the first vowel character in `S` across the entire length of `S`. If no vowel is present, he wants to replicate the first character of `S`.

Bob has written a function `ReplicateVowel` that accepts the integer `N` and string `S` in its arguments. However, there are some bugs in the function logic. Your task is to fix ReplicateVowel to help Bob complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
3 Bob	ooo	The length of bob is 3. The first vowel is 'o' so the output is "ooo".
5 sugar	uuuuu	The length of sugar is 5. The first vowel is 'u', so the output is "uuuuu".
5 Hello	eeeee	The length of hello is 5. The first vowel is 'e', so the output is "eeeee".

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static String ReplicateVowel(int N, String S) {
        //Fix the code here
        Set<Character> vowels = new HashSet<>(Arrays.asList('a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U'));
        for(int i = 0; i < N; i++) {
            char c = S.charAt(i);
            if(vowels.contains(c)) {
                char[] chars = new char[N];
                Arrays.fill(chars, c);
                return new String(chars);
            }
        }
        char[] chars = new char[N];
        Arrays.fill(chars, S.charAt(0));
        return new String(chars);
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int N = Integer.parseInt(scan.nextLine().trim());

        String S = scan.nextLine();

        String result = ReplicateVowel(N, S);

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

Console | Custom Test Case | Run code | Submit code

2416 min left

4. Fix the code to calculate remainder sum - 2

Previous | Next

Joe has two integers `a` and `b`. He wants to find the remainder when `a` is divided by `b` and do the following:

1. If the remainder is an even number greater than 2, then output the sum of `a`, `b`, and the remainder.
2. If the remainder is an odd number, then output the product of `a` and the remainder.
3. If the remainder is 0, 1, or 2, then output the sum of `a` and the remainder.

Joe has written a function `buggyRemainderOperation` that accepts integers `a` and `b` in its arguments. However, there are some bugs in the function logic. Your task is to fix buggyRemainderOperation to help Joe complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
55 594	3025	If you input a = 55 and b = 594, the remainder when a is divided by b is 55 because 55 is less than 594. The conditions in the function dictate that if the remainder is odd (which 55 is), the function should output the product of a and the remainder. Therefore, the output for the inputs 55 and 594 would be 55 * 55 = 3025.
625 779	390625	If you input a = 625 and b = 779, the remainder when a is divided by b is 625 because 625 is less than 779. The conditions in the function dictate that if the remainder is odd (which 625 is), the function should output the product of a and the remainder. Therefore, the output for the inputs

18/18 Answered

Submit

Java 8

Reset code A+ A-

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


public class Solution {
    public static int buggyRemainderOperation(int a, int b) {
        // Fix the code here
        int remainder = a % b;
        if (remainder > 2 && remainder % 2 == 0) {
            return a + b + remainder;
        } else if (remainder % 2 == 1) {
            return a * remainder;
        } else if (remainder >= 0 && remainder <= 2) {
            return a + remainder;
        } else {
            System.out.println("Unexpected case");
            return -1;  // Return some error code
        }
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int a = Integer.parseInt(scan.nextLine().trim());

        int b = Integer.parseInt(scan.nextLine().trim());

        int result = buggyRemainderOperation(a, b);

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

Console | Custom Test Case | Run code | Submit code

2416 min left

3. Fix the code to replace vowels in a string - 2

Previous | Next

Bob has a string `s`. He wants to replace all vowels in `s` with the number 3. However, if a vowel is surrounded by consonants on both sides, it should be replaced with the number 5 instead.

Bob has written a function `BuggyReplaceVowels` that accepts the string `s` in its arguments and returns the updated version of the string s. However, there are some bugs in the function logic.

Your task is to fix the issues in BuggyReplaceVowels to fix the code.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
hello	h5ll3	'e' is surrounded by consonants. However, 'o' is not surrounded. Therefore the transformed string is h5ll3.
'o' is		

18/18 Answered

Submit

Java 8

Reset code A+ A-

```
1  mport java.io.*;
2  mport java.util.*;
3  mport java.lang.Math;
4
5
6  ublic class Solution {
7      public static String BuggyReplaceVowels(String s) {
8          // Fix the code here
9          StringBuilder sb = new StringBuilder();
10         int len = s.length();
11         for (int i = 0; i < len; i++) {
12             char ch = s.charAt(i);
13             if ("aeiouAEIOU".indexOf(ch) != -1) {
14                 if (i > 0 && i < len - 1 && "bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ".indexOf(s.charAt(i - 1)) !=
15                         && "bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ".indexOf(s.charAt(i + 1)) != -1) {
16                     sb.append('5');
17                 } else {
18                     sb.append('3');
19                 }
20             } else {
21                 sb.append(ch);
22             }
23         }
24         return sb.toString();
25
26     }
27
28     public static void main(String[] args) {
29         Scanner scan = new Scanner(System.in);
30
31         String s = scan.nextLine();
32
33         String result = BuggyReplaceVowels(s);
34
35         System.out.println(result);
36     }
37
```

Console | Custom Test Case

Run code | Submit code

All Test cases passed

2416 min left

2. Fix the code to count the occurrences of a target character - 2

Previous | Next

Jake has a string array `arr` of size `n` and two special characters, `target1` and `target2`. He wants to count the total number of occurrences of `target1` in all strings in `arr`, but only if `target1` is immediately followed by `target2` in the string.

Jake has written a function `buggyCountTargetCharacterPairs` that accepts `arr`, `target1`, and `target2` in its arguments. However, there are some bugs in the function logic. Your task is to fix the issues in `buggyCountTargetCharacterPairs` to help Jake complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
3 abc def gab a b	2	'a' followed by 'b' appears twice in abc and in gab. Thus, the answer is 2.
3 aaa aaa aaa a a	6	'a' followed by 'a' appears twice in aaa. We have 3 strings of "aaa" in the given array. Hence the answer is 3*2=6.

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggyCountTargetCharacterPairs(List<String> arr, char target1, char target2
        // Fix the code here
        int count = 0;
        for (String s : arr) {
            for (int i = 0; i < s.length() - 1; i++) {
                if (s.charAt(i) == target1 && s.charAt(i + 1) == target2) {
                    count++;
                }
            }
        }
        return count;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<String> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(scan.nextLine());
        }

        char target1 = scan.nextLine().charAt(0);

        char target2 = scan.nextLine().charAt(0);

        int result = buggyCountTargetCharacterPairs(arr, target1, target2);

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

Console | Custom Test Case | Run code | Submit code

2417 min left

1. Fix the code to find sum of even numbers - 2

Previous | Next

John has an array `arr` of size `n`. He wants to find the sum of squares of all even numbers in `arr` that are also divisible by 4.
John has written a function `buggySumOfEvenSquaresDivisibleByFour` that accepts `n` and `arr` as a list of integers in its arguments. However, there are some bugs in the function logic.

Your task is to fix the issues in buggySumOfEvenSquaresDivisibleByFour to help john complete the task.

- Function description
- Constraints
- Input format for debugging
- Sample Testcases

Input	Output	Output Description
10 1 2 3 4 5 6 7 8 9 10	80	4 and 8 satisfy the requirements. Hence, the answer is 16 + 64 = 80.
4 4 4 4	64	All of the elements in arr satisfy the given requirements. Hence, the answer is 16+16+16+16=64.

18/18 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggySumOfEvenSquaresDivisibleByFour(List<Integer> arr) {
        // Fix the code here
        int result = 0;

        for (int num : arr) {
            if (num % 2 == 0 && num % 4 == 0) {
                result += num * num;
            }
        }

        return result;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<Integer> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(Integer.parseInt(scan.nextLine().trim()));
        }

        int result = buggySumOfEvenSquaresDivisibleByFour(arr);

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

Console | Custom Test Case | Run code | Submit code

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1716 min left

5. Fix the code to determine if a given string is a valid phone number

Previous | Next

You are given a partially implemented code to determine if a given string is a valid phone number. The code reads a string and calls the function `buggyIsValidPhoneNumber` to check if the string is a valid phone number. However, the function contains a bug that causes it to produce incorrect results.

A valid phone number should meet the following requirements:

1. Be exactly 10 digits long, or 12 digits long if it includes dashes.
2. Contain only digits (0-9) and optionally dashes (-).
3. If dashes are included, they should appear after the 3rd and 6th digits (e.g., "123-456-7890").

Thus, if the given phone number is "1234567890", the output should be `true`, and if the given phone number is "123-456-7890", the output should also be `true`. However, if the given phone number is "123-4567-890", the output should be `false` because the dashes are not in the correct positions.

Your task is to find and fix the bug in the `buggyIsValidPhoneNumber` function.

- Function description
- Constraints
- Input format for debugging
- Sample Testcases

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

```java
import java.io.*;
import java.util.*;
import java.lang.Math;
import java.util.Scanner;
import java.util.regex.Pattern;
import java.util.regex.Matcher;
import java.util.regex.*;

public class Solution {
    public static boolean buggyIsValidPhoneNumber(String phoneNumber) {
        // Fix the code here
        String regex = "^(\\d{10}|\\d{3}-\\d{3}-\\d{4}$)";
        Pattern pattern = Pattern.compile(regex);
        Matcher matcher = pattern.matcher(phoneNumber);
        return matcher.matches();
        //
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String phoneNumber = scan.nextLine();

        boolean result = buggyIsValidPhoneNumber(phoneNumber);

        System.out.println(result? "true": "false");
    }
}
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:24 26-12-2024

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1716 min left

4. Fix the code to find product of prime numbers in a list

Previous | Next

Sam has an array **arr** containing **n** integers. He wants to calculate the product of all prime numbers in **arr**. Therefore, he has written a function buggyProductOfPrimeNumbers that accepts **n** and stores **arr** as a list of integers as its arguments.

However, there are some issues in the code for this function. Fix the issues in buggyProductOfPrimeNumbers to help Sam complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
1 7	7	This list has a single prime number 7, so the result is 7.
4 7 9 13 14	91	arr has two prime numbers: 7 and 13. Therefore the result is: 7 * 13 = 91
4		

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static long buggyProductOfPrimeNumbers(int n, List<Integer> arr) {
        // Fix the code here
        long product = 1;
        for (int i = 0; i < n; i++) {
            int num = arr.get(i);
            if (num < 2) {
                continue;
            }
            boolean isPrime = true;
            for (int j = 2; j <= Math.sqrt(num); j++) {
                if (num % j == 0) {
                    isPrime = false;
                    break;
                }
            }
            if (isPrime) {
                product *= num;
            }
        }
        return product;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:24 26-12-2024

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1716 min left

3. Fix the buggy program to find sum of the factorial of all odd numbers

Previous | Next

John has an array **arr** of size **n**. He wants to find the sum of the factorial of all odd numbers in **arr**.

John has written a function buggySumOfOddFactorials that accepts **n** and stores **arr** as a list of integers in its arguments. However, there are some bugs in the function logic. Fix the issues to help John complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
2 6 11	39916800	The only odd number in this list is 11. The factorial of 11 is 39916800. Hence the answer 39916800.
4 7 16 9 4	367920	This list has 2 odd numbers 7 and 9. The factorial of 7 is 5040 and for 9 is 362880. The sum of these is 367920

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static long buggySumOfOddFactorials(int n, List<Integer> arr) {
        // Fix the code here
        long sum = 0;
        for (int i = 0; i < arr.size(); i++) {
            if (arr.get(i) % 2 == 1) {
                long fact = 1;
                for (int j = 1; j <= arr.get(i); j++) {
                    fact *= j;
                }
                sum += fact;
            }
        }
        return sum;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<Integer> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(Integer.parseInt(scan.nextLine().trim()));
        }

```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:24 26-12-2024

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1717 min left

2. Fix the code to determine if a given string is a valid credit card number

Previous | Next

You are given a partially implemented code to determine if a given string is a valid credit card number. The code reads a string and calls the function `buggyIsValidCreditCard` to check if the string is a valid credit card number. However, the function contains a bug that causes it to produce incorrect results.

A valid credit card number should:

1. Contain exactly 16 digits.
2. Contain only digits (0-9).
3. Pass the Luhn algorithm (a simple checksum formula used to validate a variety of identification numbers, such as credit card numbers).

The Luhn algorithm works as follows:

1. Double the value of every second digit from the rightmost digit.
2. If doubling a number results in a two-digit number, add the two digits to get a single-digit number.
3. Take the sum of all the single-digit numbers obtained in the previous steps.
4. If the total modulo 10 is equal to 0 (if the total ends in zero), the number is valid according to the Luhn formula; otherwise, it is not valid.

For example, if the input string is "4532015112830366", the output should be `true`, but if the input string is "1234567812345678", the output should be `false` because it does not pass the Luhn algorithm.

Your task is to find and fix the bug in the `buggyIsValidCreditCard` function.

Function description

20/20 Answered

Submit

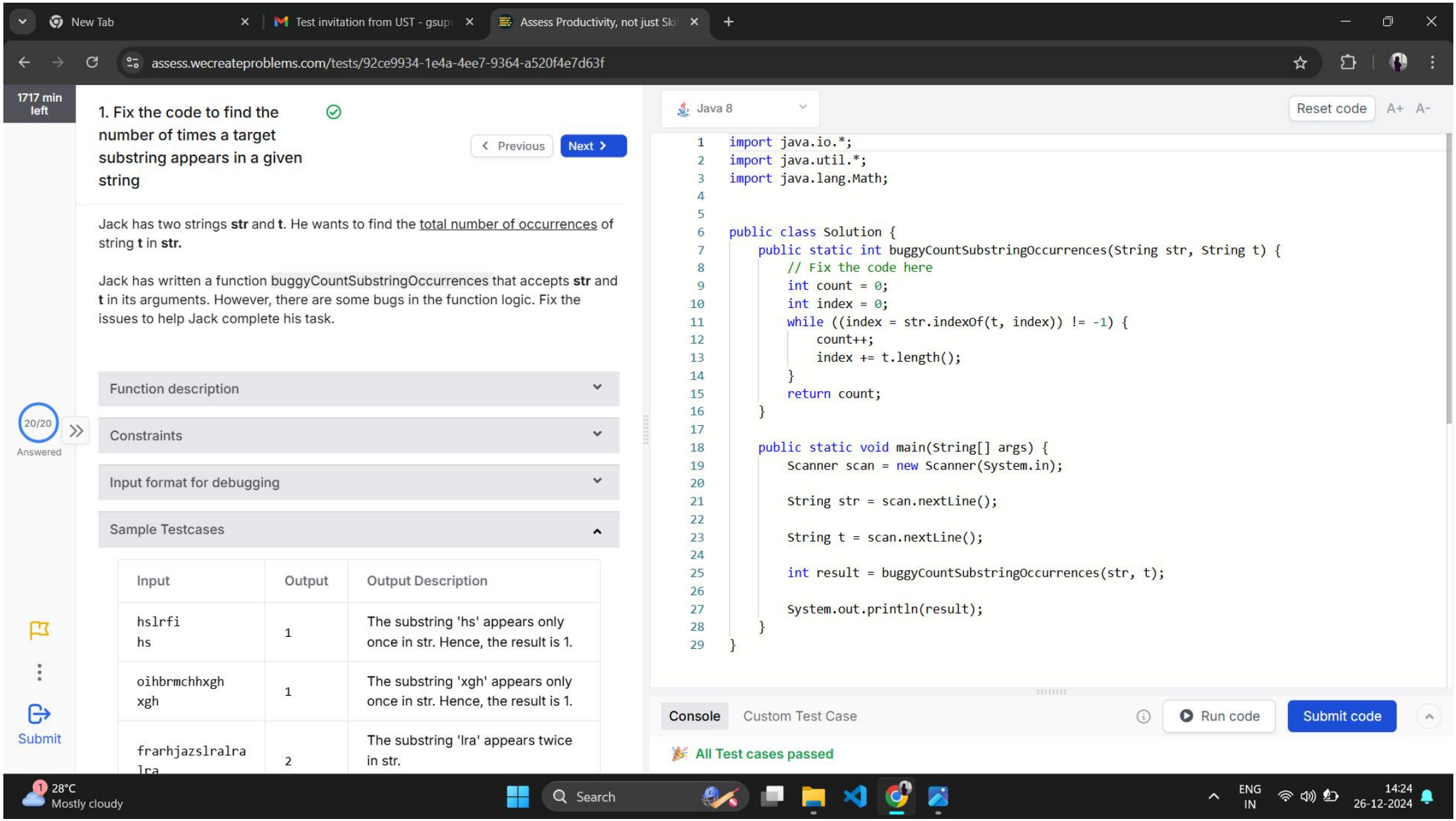
Java 8 | Reset code | A+ | A-

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


public class Solution {
    public static boolean buggyIsValidCreditCard(String number) {
        // Fix the code here
        int len = number.length();
        if (len != 16) return false;

        int sum = 0;
        boolean doubleNext = false;
        for (int i = len-1; i >= 0; i--) {
            char c = number.charAt(i);
            if (c < '0' || c > '9') return false;

            int digit = c - '0';
            if (doubleNext) {
                digit *= 2;
                if (digit > 9) digit = digit/10 + digit%10;
            }
            sum += digit;
            doubleNext = !doubleNext;
        }

        return sum % 10 == 0;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:24 26-12-2024


New Tab
Test invitation from UST - gsup
Assess Productivity, not just Ski
assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f
1717 min left
1. Fix the code to find the number of times a target substring appears in a given string
Previous
Next
Jack has two strings str and t. He wants to find the total number of occurrences of string t in str.
Jack has written a function buggyCountSubstringOccurrences that accepts str and t in its arguments. However, there are some bugs in the function logic. Fix the issues to help Jack complete his task.
Function description
Constraints
Input format for debugging
Sample Testcases
Input
Output
Output Description
hslrfi
hs
1
The substring 'hs' appears only once in str. Hence, the result is 1.
oihbrmchhxgh
xgh
1
The substring 'xgh' appears only once in str. Hence, the result is 1.
frarhjazslralra
lra
2
The substring 'lra' appears twice in str.
20/20
Answered
Submit
Java 8
Reset code
A+
A-
1 import java.io.*;
2 import java.util.*;
3 import java.lang.Math;
4
5
6 public class Solution {
7 public static int buggyCountSubstringOccurrences(String str, String t) {
8 // Fix the code here
9 int count = 0;
10 int index = 0;
11 while ((index = str.indexOf(t, index)) != -1) {
12 count++;
13 index += t.length();
14 }
15 return count;
16 }
17
18 public static void main(String[] args) {
19 Scanner scan = new Scanner(System.in);
20
21 String str = scan.nextLine();
22
23 String t = scan.nextLine();
24
25 int result = buggyCountSubstringOccurrences(str, t);
26
27 System.out.println(result);
28 }
29 }
Console
Custom Test Case
Run code
Submit code
All Test cases passed
28°C
Mostly cloudy
Search
ENG
IN
14:24
26-12-2024

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1717 min left

10. Fix the code to find sum of squares of all odd numbers

Previous | Next

John has an array **arr** of size **n**. He wants to find the sum of squares of all odd numbers in **arr**.

John has written a function buggySumOfOddSquares that accepts **n** and stores **arr** as a list of integers in its arguments. However, there are some bugs in the function logic. Fix the issues to help John complete his task.

- Function description
- Constraints
- Input format for debugging
- Sample Testcases

Input	Output	Output Description
2 1 1	2	Here 1^2 + 1^2 = 2. Hence the answer is 2.
6 1 2 3 4 5	35	Here 1^2 + 3^2 +5^2 =35. Hence the answer is 35.

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {

    public static boolean isOdd(int num) {
        return num % 2 != 0;
    }

    public static int buggySumOfOddSquares(int n, List<Integer> arr) {
        // Fix the code here
        int sum = 0;
        for (int i = 0; i < n; i++) {
            if (isOdd(arr.get(i))) {
                sum += arr.get(i) * arr.get(i);
            }
        }
        return sum;

    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<Integer> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(Integer.parseInt(scan.nextLine().trim()));
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:23 26-12-2024

1717 min left

8. Fix the code to count the number of strings in a list that have a specified length

Previous | Next

Jake has a string array **arr** of size **n** and an integer value **length.** He wants to count the total number strings in **arr** whose length is equal to the given integer value.

Jake has written a function `buggyCountStringsOfTargetLength` that accepts **arr** and **length** in its arguments. However, there are some bugs in the function logic. Fix the issues to help Jake complete his task.

- Function description
- Constraints
- Input format for debugging
- Sample Testcases

Input	Output	Output Description
1 ejj 6	0	There are no strings in the array that have a length of 6. Hence the answer is 0.
8 geaiqrl mfypgzx bhdfsxdp bk	2	Out of the 8 strings in the input array, only two of them have a length of 10: "lidthpzexx" and "svdifclnxe".

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggyCountStringsOfTargetLength(int n, List<String> arr, int length) {
        // Fix the code here
        int count = 0;
        for (int i = 0; i < n; i++) {
            if (arr.get(i).length() == length) {
                count++;
            }
        }
        return count;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<String> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(scan.nextLine());
        }

        int length = Integer.parseInt(scan.nextLine().trim());

        int result = buggyCountStringsOfTargetLength(n, arr, length);
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

1718 min left

7. Fix the code to check if a given string is a palindrome

Previous | Next

You are given a partially implemented code to check if a given string is a palindrome. If the given string is a palindrome return 1 else return 0.

A palindrome is a word, phrase, number, or other sequences of characters that reads the same forward and backward (ignoring spaces, punctuation, and capitalization). The code reads a string and calls the function `buggyIsPalindrome` to determine if the string is a palindrome.

However, the function contains a bug that causes it to produce incorrect results. Your task is to find and fix the bug in the `buggyIsPalindrome` function.

- Function description
- Constraints
- Input format for debugging
- Sample Testcases

Input	Output	Output Description
Not a palindrome	0	This text in not the same when read forward and backward, so it's not a palindrome. The output is 0 (false).
		This text is the same when read forward and backward. It is a palindrome.

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static boolean buggyIsPalindrome(String s) {
        // Fix the code here
        String cleanedS = s.replaceAll("[^A-Za-z0-9]+", "").toLowerCase();
        for(int i = 0; i < cleanedS.length() / 2; i++) {
            if(cleanedS.charAt(i) != cleanedS.charAt(cleanedS.length() - 1- i)) {
                return false;
            }
        }
        return true;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String s = scan.nextLine();

        boolean result = buggyIsPalindrome(s);

        System.out.println(result? "1": "0");
    }
}
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1718 min left

6. Fix the code to determine if a given string is a valid URL

Previous | Next

You are given a partially implemented code to determine if a given string is a valid URL. The code reads a string and calls the function `buggyIsValidURL` to check if the string is a valid URL.
However, the function contains a bug that causes it to produce incorrect results.

A valid URL should:

1. Start with a valid scheme (e.g., "http", "https", "ftp").
2. Followed by "://".
3. Then, a domain part, which may contain alphanumeric characters (both uppercase and lowercase), hyphens (-), and periods (.). The domain part must start and end with an alphanumeric character.
4. Optionally, a path part after the domain, which may contain alphanumeric characters (both uppercase and lowercase), hyphens (-), slashes (/), and periods (.).

For example, if the input string is "https://www.example.com/test", the output should be `true`, but if the input string is "http:/www.example.com/test", the output should be `false` because the URL does not have a valid scheme.

Your task is to find and fix the bug in the `buggyIsValidURL` function.

Function description

Constraints

Input format for debugging

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

```java
import java.io.*;
import java.util.*;
import java.lang.Math;
import java.util.regex.Pattern;
import java.util.regex.Matcher;

public class Solution {
    public static boolean buggyIsValidURL(String url) {
        // Fix the code here
        String regex = "^(http?|ftp|https)://([a-zA-Z0-9-]+\\.)+[a-zA-Z0-9]+(/[a-zA-Z0-9-./]*)?$";
        Pattern pattern = Pattern.compile(regex, Pattern.CASE_INSENSITIVE);
        Matcher matcher = pattern.matcher(url);
        return matcher.matches();
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String url = scan.nextLine();

        boolean result = buggyIsValidURL(url);

        System.out.println(result? "true": "false");
    }
}
```

Console | Custom Test Case | Run code | Submit code

2/8 Test cases failed

28°C Mostly cloudy | Search | ENG IN | 14:22 26-12-2024

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1718 min left

5. Fix the buggy program to find sum of squares of all even numbers

Previous | Next

John has an array **arr** of size **n**. He wants to find the **sum of squares** of all even numbers in **arr**.

John has written a function buggySumOfEvenSquares that accepts **n** and stores **arr** as a list of integers in its arguments. However, there are some bugs in the function logic. Fix the issues to help John complete his task.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
4 7 11 17 12	144	arr has a single even number 12, the square of 12 is 144. Hence, the answer is 144
6 16 9 19	256	arr has 1 even number 16. The square of 16 is 256. Hence, the answer is 256.

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggySumOfEvenSquares(int n, List<Integer> arr) {
        // Fix the bug here
        int sum = 0;
        for (int i = 0; i < n; i++) {
            int num = arr.get(i);
            if (num % 2 == 0) {
                sum += num*num;
            }
        }
        return sum;

    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<Integer> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(Integer.parseInt(scan.nextLine().trim()));
        }

        int result = buggySumOfEvenSquares(n, arr);

```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:22 26-12-2024

1719 min left

3. Fix the code to find number of words in a sentence that start with a specified character

Previous | Next

Jill has a string **str**. She wants to find the number of words in **str** that start with a specified character **c**.

Jill has written a function `buggyCountWordsStartingWithCharacter` that accepts **str** and **c** in its arguments. However, there are some bugs in the function logic. Fix the issues to help Jill complete his task.

Notes:

- It is guaranteed that all inputs are provided in lowercase.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description
s this is a simple example	1	Only one word starts with the letter s, so the answer is 1.
h here is another example. here.	2	Here, we have two words starting with the letter h, so the answer is 2.

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggyCountWordsStartingWithCharacter(char c, String str) {
        // Fix the code here
        int count = 0;
        String[] words = str.split(" ");
        for (String word : words) {
            if (word.length() > 0 && word.charAt(0) == c) {
                count++;
            }
        }
        return count;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        char c = scan.nextLine().charAt(0);

        String str = scan.nextLine();

        int result = buggyCountWordsStartingWithCharacter(c, str);

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

Console | Custom Test Case | Run code | Submit code

All Test cases passed

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1719 min left

2. Fix the code to determine if a given string contains all unique characters

Previous | Next

You are given a partially implemented code to determine if a given string contains all unique characters. The code reads a string and calls the function `buggyHasUniqueCharacters` to check if the string has all unique characters. However, the function contains a bug that causes it to produce incorrect results.

Your task is to find and fix the bug in the `buggyHasUniqueCharacters` function.

- Function description
- Constraints
- Input format for debugging
- Sample Testcases

Input	Output	Output Description
abcd	1	All unique characters are present in the given string so output 1 (true).
notunique	0	Characters n and u are repeated. The output is 0 (false).
almostunique	0	Character u is repeated. The output is 0 (false).

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static boolean buggyHasUniqueCharacters(String str) {
        // Fix the code here.
        boolean[] charSet = new boolean[128];
        for (char c : str.toCharArray()) {
            if (charSet[c]) {
                return false;
            }
            charSet[c] = true;
        }
        return true;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String str = scan.nextLine();

        boolean result = buggyHasUniqueCharacters(str);

        System.out.println(result? "1": "0");
    }
}
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:21 26-12-2024

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1719 min left

1. Fix the code to determine if a given string is a valid email address

Previous | Next

You are given a partially implemented code to determine if a given string is a valid email address. The code reads a string and calls the function `buggyIsValidEmail` to check if the string is a valid email address.
However, the function contains a bug that causes it to produce incorrect results.

Your task is to find and fix the bug in the `buggyIsValidEmail` function.

A valid email address should:

1. Consist of a local part followed by an "@" symbol, and a domain part.
2. The local part may contain alphanumeric characters (both uppercase and lowercase), periods (.), and plus signs (+).
3. The domain part may contain alphanumeric characters (both uppercase and lowercase) and hyphens (-), but it must start and end with an alphanumeric character.

For example, if the input string is "john.doe@example.com", the output should be `true`, but if the input string is "john..doe@example.com", the output should be `false` because two consecutive periods are not allowed.

Function description

Constraints

Input format for debugging

Sample Testcases

Input	Output	Output Description

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

```java
import java.io.*;
import java.util.*;
import java.lang.Math;
import java.util.regex.Matcher;
import java.util.regex.Pattern;


public class Solution {
    public static boolean buggyIsValidEmail(String email) {
        String regex = "(^[a-zA-Z0-9._+]).+(@[a-zA-Z0-9-]+\\.[a-zA-Z0-9-.]+$)";
        Pattern pattern = Pattern.compile(regex);
        Matcher matcher = pattern.matcher(email);
        return matcher.matches();

    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        String email = scan.nextLine();

        boolean result = buggyIsValidEmail(email);

        System.out.println(result? "true" : "false");
    }
}
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:21 26-12-2024

1720 min left

5. Fix the code to count the occurrences of a target character

Previous | Next

Jake has a string array **arr** of size **n** and a special character **target.** He wants to count the total number of occurrences of **target** in all strings in **arr.**

Jake has written a function `buggyCountTargetCharacter` that accepts **arr** and **target** in its arguments. However, there are some bugs in the function logic. Fix the issues to help Jake complete his task.

Function description

Constraints

- 0 ≤ **n** ≤ 10^5
- 1 ≤ **len(arr[i])** ≤ 10^5

Input format for debugging

Sample Testcases

Input	Output	Output Description
0 r	0	The input consists of zero strings in arr and therefore there is no string to match with the target character "r". Hence, the answer is 0.
1		The input consists of a single string "amzq" and the target character is "y".

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggyCountTargetCharacter(int n, List<String> arr, char target) {
        // Fix the code here
        int count = 0;
        for (int i = 0; i < n; i++) {
            String str = arr.get(i);
            for (int j = 0; j < str.length(); j++) {
                if (str.charAt(j) == target) {
                    count++;
                }
            }
        }
        return count;

    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<String> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(scan.nextLine());
        }
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1720 min left

4. Fix the code to find sum of even numbers

Previous | Next

John has an array **arr** of size **n**. He wants to find the sum of squares of all even numbers in **arr**.

John has written a function buggySumOfEvenNumbers that accepts **n** and stores **arr** as a list of integers in its arguments. However, there are some bugs in the function logic. Fix the issues to help John complete his task.

Function description

Constraints

- 1 ≤ **n** ≤ 10^5
- -10^5 ≤ **arr[i]** ≤ 10^5

Input format for debugging

Sample Testcases

Input	Output	Output Description
3 0 1 2	2	Here, 0 and 2 are the even numbers, and their sum is 2.
2 -1 -2	-2	Here, Only -2 is an even number. Hence its sum is -2 itself.

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

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


public class Solution {
    public static int buggySumOfEvenNumbers(int n, List<Integer> arr) {
        // Fix the code here
        int sum = 0;
        for (int i = 0; i < n; i++) {
            if (arr.get(i) % 2 == 0) {
                sum += arr.get(i);
            }
        }
         return sum;

    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = Integer.parseInt(scan.nextLine().trim());

        List<Integer> arr = new ArrayList<>(n);
        for(int j=0; j<n; j++) {
            arr.add(Integer.parseInt(scan.nextLine().trim()));
        }

        int result = buggySumOfEvenNumbers(n, arr);

        System.out.println(result);
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:21 26-12-2024

New Tab | Test invitation from UST - gsup | Assess Productivity, not just Ski

assess.wecreateproblems.com/tests/92ce9934-1e4a-4ee7-9364-a520f4e7d63f

1720 min left

3. Fix the code to replace vowels in a string

Previous | Next

Bob has a string **S**. He wants to replace all vowels in **S** with the number 3.

Bob has written a function ReplaceVowels that accepts the string **S** in its arguments. However, there are some bugs in the function logic. Fix the issues to help Bob complete his task.

Function description

Constraints

- 1 ≤ **len(S)** ≤ 10^5

Input format for debugging

Sample Testcases

Input	Output	Output Description
Yellow	Y3ll3w	The vowels in string Yellow are e and o. Hence, after replacing it with 3 we get Y3ll3w.
Sunshine	S3nsh3n3	The vowels in string Sunshine are u, i and e. Hence, after replacing it with 3 we get S3nsh3n3.
World	W3rld	The vowel in string World is o. Hence, after replacing it with 3 we get W3rld.

20/20 Answered

Submit

Java 8 | Reset code | A+ A-

```
1  import java.io.*;
2  import java.util.*;
3  import java.lang.Math;
4
5
6  public class Solution {
7      public static String ReplaceVowels(String S) {
8          // Write your code here
9          String out="";
10         for(int i = 0; i < S.length(); i++)
11         {
12             char ch = S.charAt(i);
13            if (ch=='A'||ch=='E'||ch=='I'||ch=='O'||ch=='U'||ch=='a'||ch=='e'||ch=='i'||c
14                 {
15                     out=out+'3';
16
17                 }
18             else
19             {
20                 out=out+ch;
21             }
22         }
23         return out.toString();
24     }
25
26     public static void main(String[] args) {
27         Scanner scan = new Scanner(System.in);
28
29         String S = scan.nextLine();
30
31         String result = ReplaceVowels(S);
```

Console | Custom Test Case | Run code | Submit code

All Test cases passed

28°C Mostly cloudy | Search | ENG IN | 14:20 26-12-2024