

Problem 2019: The Score of Students Solving Math Expression

Problem Information

Difficulty: **Hard**

Acceptance Rate: 0.00%

Paid Only: No

Problem Description

You are given a string

s

that contains digits

0-9

, addition symbols

'+'

, and multiplication symbols

'*'

only

, representing a

valid

math expression of

single digit numbers

(e.g.,

$3+5*2$

). This expression was given to

n

elementary school students. The students were instructed to get the answer of the expression by following this

order of operations

:

Compute

multiplication

, reading from

left to right

; Then,

Compute

addition

, reading from

left to right

.

You are given an integer array

answers

of length

n

, which are the submitted answers of the students in no particular order. You are asked to grade the

answers

, by following these

rules

:

If an answer

equals

the correct answer of the expression, this student will be rewarded

5

points;

Otherwise, if the answer

could be interpreted

as if the student applied the operators

in the wrong order

but had

correct arithmetic

, this student will be rewarded

2

points;

Otherwise, this student will be rewarded

0

points.

Return

the sum of the points of the students

.

Example 1:

$$\begin{array}{rcll} & 7 & + & 3 * 1 * 2 \\ = & 7 & + & \underline{3} * 2 & \text{Compute multiplication, reading from left to right.} \\ = & 7 & + & \underline{6} & \text{Compute multiplication, reading from left to right.} \\ = & & & \underline{13} & \text{Compute addition, reading from left to right.} \end{array}$$

Input:

s = "7+3*1*2", answers = [20,13,42]

Output:

7

Explanation:

As illustrated above, the correct answer of the expression is 13, therefore one student is rewarded 5 points: [20,

13

,42] A student might have applied the operators in this wrong order: $((7+3)*1)*2 = 20$. Therefore one student is rewarded 2 points: [

20

,13,42] The points for the students are: [2,5,0]. The sum of the points is $2+5+0=7$.

Example 2:

Input:

s = "3+5*2", answers = [13,0,10,13,13,16,16]

Output:

19

Explanation:

The correct answer of the expression is 13, therefore three students are rewarded 5 points each: [

13

,0,10,

13

,

13

,16,16] A student might have applied the operators in this wrong order: $((3+5)*2 = 16$. Therefore two students are rewarded 2 points: [13,0,10,13,13,

16

,

16

] The points for the students are: [5,0,0,5,5,2,2]. The sum of the points is $5+0+0+5+5+2+2=19$.

Example 3:

Input:

`s = "6+0*1", answers = [12,9,6,4,8,6]`

Output:

10

Explanation:

The correct answer of the expression is 6. If a student had incorrectly done $(6+0)*1$, the answer would also be 6. By the rules of grading, the students will still be rewarded 5 points (as they got the correct answer), not 2 points. The points for the students are: [0,0,5,0,0,5]. The sum of the points is 10.

Constraints:

`3 <= s.length <= 31`

`s`

represents a valid expression that contains only digits

0-9

,

'+'

, and

'*'

only.

All the integer operands in the expression are in the

inclusive

range

[0, 9]

.

1 <=

The count of all operators (

'+'

and

'*')

) in the math expression

<= 15

Test data are generated such that the correct answer of the expression is in the range of

[0, 1000]

.

Test data are generated such that value never exceeds 10

9

in intermediate steps of multiplication.

n == answers.length

$1 \leq n \leq 10$

4

$0 \leq \text{answers}[i] \leq 1000$

Code Snippets

C++:

```
class Solution {
public:
    int scoreOfStudents(string s, vector<int>& answers) {

    }
};
```

Java:

```
class Solution {
    public int scoreOfStudents(String s, int[] answers) {

    }
}
```

Python3:

```
class Solution:
    def scoreOfStudents(self, s: str, answers: List[int]) -> int:
```

Python:

```
class Solution(object):
    def scoreOfStudents(self, s, answers):
        """
        :type s: str
        :type answers: List[int]
        :rtype: int
        """
```


JavaScript:

```
/**
 * @param {string} s
 * @param {number[]} answers
 * @return {number}
 */
var scoreOfStudents = function(s, answers) {

};
```

TypeScript:

```
function scoreOfStudents(s: string, answers: number[]): number {

};
```

C#:

```
public class Solution {
    public int ScoreOfStudents(string s, int[] answers) {

    }
}
```

C:

```
int scoreOfStudents(char* s, int* answers, int answersSize) {

}
```

Go:

```
func scoreOfStudents(s string, answers []int) int {

}
```

Kotlin:

```
class Solution {
    fun scoreOfStudents(s: String, answers: IntArray): Int {

    }
}
```

```
}
```

Swift:

```
class Solution {  
    func scoreOfStudents(_ s: String, _ answers: [Int]) -> Int {  
  
    }  
}
```

Rust:

```
impl Solution {  
    pub fn score_of_students(s: String, answers: Vec<i32>) -> i32 {  
  
    }  
}
```

Ruby:

```
# @param {String} s  
# @param {Integer[]} answers  
# @return {Integer}  
def score_of_students(s, answers)  
  
end
```

PHP:

```
class Solution {  
  
    /**  
     * @param String $s  
     * @param Integer[] $answers  
     * @return Integer  
     */  
    function scoreOfStudents($s, $answers) {  
  
    }  
}
```

Dart:

```
class Solution {
  int scoreOfStudents(String s, List<int> answers) {

  }
}
```

Scala:

```
object Solution {
  def scoreOfStudents(s: String, answers: Array[Int]): Int = {

  }
}
```

Elixir:

```
defmodule Solution do
  @spec score_of_students(s :: String.t, answers :: [integer]) :: integer
  def score_of_students(s, answers) do

  end
end
```

Erlang:

```
-spec score_of_students(S :: unicode:unicode_binary(), Answers ::
[integer()]) -> integer().
score_of_students(S, Answers) ->
.
```

Racket:

```
(define/contract (score-of-students s answers)
  (-> string? (listof exact-integer?) exact-integer?)
)
```

Solutions

C++ Solution:

```

/*
 * Problem: The Score of Students Solving Math Expression
 * Difficulty: Hard
 * Tags: array, string, dp, math, hash, stack
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

class Solution {
public:
    int scoreOfStudents(string s, vector<int>& answers) {

    }
};

```

Java Solution:

```

/**
 * Problem: The Score of Students Solving Math Expression
 * Difficulty: Hard
 * Tags: array, string, dp, math, hash, stack
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

class Solution {
    public int scoreOfStudents(String s, int[] answers) {

    }
}

```

Python3 Solution:

```

"""
Problem: The Score of Students Solving Math Expression
Difficulty: Hard
Tags: array, string, dp, math, hash, stack

```

```

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) or O(n * m) for DP table
"""

class Solution:
    def scoreOfStudents(self, s: str, answers: List[int]) -> int:
        # TODO: Implement optimized solution
        pass

```

Python Solution:

```

class Solution(object):
    def scoreOfStudents(self, s, answers):
        """
        :type s: str
        :type answers: List[int]
        :rtype: int
        """

```

JavaScript Solution:

```

/**
 * Problem: The Score of Students Solving Math Expression
 * Difficulty: Hard
 * Tags: array, string, dp, math, hash, stack
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

/**
 * @param {string} s
 * @param {number[]} answers
 * @return {number}
 */
var scoreOfStudents = function(s, answers) {

};

```

TypeScript Solution:

```
/**
 * Problem: The Score of Students Solving Math Expression
 * Difficulty: Hard
 * Tags: array, string, dp, math, hash, stack
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

function scoreOfStudents(s: string, answers: number[]): number {

};
```

C# Solution:

```
/*
 * Problem: The Score of Students Solving Math Expression
 * Difficulty: Hard
 * Tags: array, string, dp, math, hash, stack
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

public class Solution {
    public int ScoreOfStudents(string s, int[] answers) {

    }
}
```

C Solution:

```
/*
 * Problem: The Score of Students Solving Math Expression
 * Difficulty: Hard
 * Tags: array, string, dp, math, hash, stack
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
```

```

* Space Complexity: O(n) or O(n * m) for DP table
*/

int scoreOfStudents(char* s, int* answers, int answersSize) {

}

```

Go Solution:

```

// Problem: The Score of Students Solving Math Expression
// Difficulty: Hard
// Tags: array, string, dp, math, hash, stack
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table

func scoreOfStudents(s string, answers []int) int {

}

```

Kotlin Solution:

```

class Solution {
    fun scoreOfStudents(s: String, answers: IntArray): Int {

    }
}

```

Swift Solution:

```

class Solution {
    func scoreOfStudents(_ s: String, _ answers: [Int]) -> Int {

    }
}

```

Rust Solution:

```

// Problem: The Score of Students Solving Math Expression
// Difficulty: Hard

```

```

// Tags: array, string, dp, math, hash, stack
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table

impl Solution {
    pub fn score_of_students(s: String, answers: Vec<i32>) -> i32 {

    }
}

```

Ruby Solution:

```

# @param {String} s
# @param {Integer[]} answers
# @return {Integer}
def score_of_students(s, answers)

end

```

PHP Solution:

```

class Solution {

    /**
     * @param String $s
     * @param Integer[] $answers
     * @return Integer
     */
    function scoreOfStudents($s, $answers) {

    }

}

```

Dart Solution:

```

class Solution {
    int scoreOfStudents(String s, List<int> answers) {

    }
}

```



```
}
```

Scala Solution:

```
object Solution {  
  def scoreOfStudents(s: String, answers: Array[Int]): Int = {  
  
  }  
}
```

Elixir Solution:

```
defmodule Solution do  
  @spec score_of_students(s :: String.t, answers :: [integer]) :: integer  
  def score_of_students(s, answers) do  
  
  end  
end
```

Erlang Solution:

```
-spec score_of_students(S :: unicode:unicode_binary(), Answers ::  
[integer()]) -> integer().  
score_of_students(S, Answers) ->  
.
```

Racket Solution:

```
(define/contract (score-of-students s answers)  
  (-> string? (listof exact-integer?) exact-integer?)  
)
```