Problem 2745: Construct the Longest New String

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

You are given three integers

x

,

y

, and

z

.

You have

x

strings equal to

"AA"

,

y

strings equal to

"BB"

, and

z

strings equal to

"AB"

. You want to choose some (possibly all or none) of these strings and concatenate them in some order to form a new string. This new string must not contain

"AAA"

or

"BBB"

as a substring.

Return

the maximum possible length of the new string

.

A

substring

is a contiguous

non-empty

sequence of characters within a string.

Example 1:

Input:

x = 2, y = 5, z = 1

Output:

12

Explanation:

We can concatenate the strings "BB", "AA", "BB", "AA", "BB", and "AB" in that order. Then, our new string is "BBAABBAABBAB". That string has length 12, and we can show that it is impossible to construct a string of longer length.

Example 2:

Input:

x = 3, y = 2, z = 2

Output:

14

Explanation:

We can concatenate the strings "AB", "AB", "AA", "BB", "AA", "BB", and "AA" in that order. Then, our new string is "ABABAABBAABBAA". That string has length 14, and we can show that it is impossible to construct a string of longer length.

Constraints:

1 <= x, y, z <= 50

Code Snippets

C++:

```
class Solution {
public:
int longestString(int x, int y, int z) {

}
};
```

Java:

```
class Solution {
public int longestString(int x, int y, int z) {

}
}
```

Python3:

```
class Solution:
def longestString(self, x: int, y: int, z: int) -> int:
```

Python:

```
class Solution(object):
def longestString(self, x, y, z):
"""
:type x: int
:type y: int
:type z: int
:rtype: int
"""
```

JavaScript:

```
/**
* @param {number} x
* @param {number} y
* @param {number} z
* @return {number}
*/
var longestString = function(x, y, z) {
```

```
};
```

TypeScript:

```
function longestString(x: number, y: number, z: number): number {

};
```

C#:

```
public class Solution {
public int LongestString(int x, int y, int z) {

}
}
```

C:

```
int longestString(int x, int y, int z) {

}
```

Go:

```
func longestString(x int, y int, z int) int {

}
```

Kotlin:

```
class Solution {
fun longestString(x: Int, y: Int, z: Int): Int {

}
}
```

Swift:

```
class Solution {
func longestString(_ x: Int, _ y: Int, _ z: Int) -> Int {

}
```

```
}
```

Rust:

```
impl Solution {
pub fn longest_string(x: i32, y: i32, z: i32) -> i32 {

}
}
```

Ruby:

```
# @param {Integer} x
# @param {Integer} y
# @param {Integer} z
# @return {Integer}
def longest_string(x, y, z)

end
```

PHP:

```
class Solution {

/**
* @param Integer $x
* @param Integer $y
* @param Integer $z
* @return Integer
*/
function longestString($x, $y, $z) {

}
}
```

Dart:

```
class Solution {
int longestString(int x, int y, int z) {

}
}
```

Scala:

```
object Solution {
def longestString(x: Int, y: Int, z: Int): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec longest_string(x :: integer, y :: integer, z :: integer) :: integer
def longest_string(x, y, z) do

end
end
```

Erlang:

```
-spec longest_string(X :: integer(), Y :: integer(), Z :: integer()) ->
integer().
longest_string(X, Y, Z) ->
.
```

Racket:

```
(define/contract (longest-string x y z)
(-> exact-integer? exact-integer? exact-integer? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Construct the Longest New String
* Difficulty: Medium
* Tags: string, tree, dp, greedy, math
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
```

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

class Solution {
public:
int longestString(int x, int y, int z) {

}
};
```

Java Solution:

```
/**
* Problem: Construct the Longest New String
* Difficulty: Medium
* Tags: string, tree, dp, greedy, math
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

class Solution {
public int longestString(int x, int y, int z) {

}
}
```

Python3 Solution:

```
"""
Problem: Construct the Longest New String
Difficulty: Medium
Tags: string, tree, dp, greedy, math

Approach: String manipulation with hash map or two pointers
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) or O(n * m) for DP table
"""

class Solution:
```

```
def longestString(self, x: int, y: int, z: int) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def longestString(self, x, y, z):
"""
:type x: int
:type y: int
:type z: int
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Construct the Longest New String
* Difficulty: Medium
* Tags: string, tree, dp, greedy, math
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

/**
* @param {number} x
* @param {number} y
* @param {number} z
* @return {number}
*/
var longestString = function(x, y, z) {

};
```

TypeScript Solution:

```
/**
* Problem: Construct the Longest New String
```

```
* Difficulty: Medium
* Tags: string, tree, dp, greedy, math
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

function longestString(x: number, y: number, z: number): number {

};
```

C# Solution:

```
/*
* Problem: Construct the Longest New String
* Difficulty: Medium
* Tags: string, tree, dp, greedy, math
*
* Approach: String manipulation with hash map or two pointers
* 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 LongestString(int x, int y, int z) {

}
}
```

C Solution:

```
/*
* Problem: Construct the Longest New String
* Difficulty: Medium
* Tags: string, tree, dp, greedy, math
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/
```

```
int longestString(int x, int y, int z) {

}
```

Go Solution:

```
// Problem: Construct the Longest New String
// Difficulty: Medium
// Tags: string, tree, dp, greedy, math
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table

func longestString(x int, y int, z int) int {

}
```

Kotlin Solution:

```
class Solution {
fun longestString(x: Int, y: Int, z: Int): Int {

}
}
```

Swift Solution:

```
class Solution {
func longestString(_ x: Int, _ y: Int, _ z: Int) -> Int {

}
}
```

Rust Solution:

```
// Problem: Construct the Longest New String
// Difficulty: Medium
// Tags: string, tree, dp, greedy, math
//
```

```
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table

impl Solution {
pub fn longest_string(x: i32, y: i32, z: i32) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer} x
# @param {Integer} y
# @param {Integer} z
# @return {Integer}
def longest_string(x, y, z)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer $x
* @param Integer $y
* @param Integer $z
* @return Integer
*/
function longestString($x, $y, $z) {

}
}
```

Dart Solution:

```
class Solution {
int longestString(int x, int y, int z) {

}
```

```
}
```

Scala Solution:

```
object Solution {
def longestString(x: Int, y: Int, z: Int): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec longest_string(x :: integer, y :: integer, z :: integer) :: integer
def longest_string(x, y, z) do

end
end
```

Erlang Solution:

```
-spec longest_string(X :: integer(), Y :: integer(), Z :: integer()) ->
integer().
longest_string(X, Y, Z) ->
.
```

Racket Solution:

```
(define/contract (longest-string x y z)
(-> exact-integer? exact-integer? exact-integer? exact-integer?)
)
```