Problem 984: String Without AAA or BBB

Problem Information

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

Problem Description

Given two integers

a

and

b

, return

any

string

s

such that:

s

has length

a + b

and contains exactly

a

'a'

letters, and exactly

b

'b'

letters,

The substring

'aaa'

does not occur in

s

, and

The substring

'bbb'

does not occur in

s

.

Example 1:

Input:

a = 1, b = 2

Output:

"abb"

Explanation:

"abb", "bab" and "bba" are all correct answers.

Example 2:

Input:

a = 4, b = 1

Output:

"aabaa"

Constraints:

0 <= a, b <= 100

It is guaranteed such an

s

exists for the given

a

and

b

.

Code Snippets

C++:

```
class Solution {
public:
string strWithout3a3b(int a, int b) {

}
};
```

Java:

```
class Solution {
public String strWithout3a3b(int a, int b) {

}
}
```

Python3:

```
class Solution:
def strWithout3a3b(self, a: int, b: int) -> str:
```

Python:

```
class Solution(object):
def strWithout3a3b(self, a, b):
"""
:type a: int
:type b: int
:rtype: str
"""
```

JavaScript:

```
/**
* @param {number} a
* @param {number} b
* @return {string}
*/
var strWithout3a3b = function(a, b) {

};
```

TypeScript:

```
function strWithout3a3b(a: number, b: number): string {

};
```

C#:

```
public class Solution {
public string StrWithout3a3b(int a, int b) {

}
}
```

C:

```


char * strWithout3a3b(int a, int b){

}
```

Go:

```
func strWithout3a3b(a int, b int) string {

}
```

Kotlin:

```
class Solution {
fun strWithout3a3b(a: Int, b: Int): String {

}
}
```

Swift:

```
class Solution {
func strWithout3a3b(_ a: Int, _ b: Int) -> String {

}
}
```

Rust:

```
impl Solution {
pub fn str_without3a3b(a: i32, b: i32) -> String {

}
}
```

Ruby:

```
# @param {Integer} a
# @param {Integer} b
# @return {String}
def str_without3a3b(a, b)

end
```

PHP:

```
class Solution {

/**
* @param Integer $a
* @param Integer $b
* @return String
*/
function strWithout3a3b($a, $b) {

}
}
```

Scala:

```
object Solution {
def strWithout3a3b(a: Int, b: Int): String = {

}
}
```

Solutions

C++ Solution:

```
/*
* Problem: String Without AAA or BBB
* Difficulty: Medium
* Tags: string, tree, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

class Solution {
public:
string strWithout3a3b(int a, int b) {

}
};
```

Java Solution:

```
/**
* Problem: String Without AAA or BBB
* Difficulty: Medium
* Tags: string, tree, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

class Solution {
public String strWithout3a3b(int a, int b) {

}
}
```

Python3 Solution:

```
"""
Problem: String Without AAA or BBB
Difficulty: Medium
Tags: string, tree, greedy
```

```
Approach: String manipulation with hash map or two pointers
Time Complexity: O(n) or O(n log n)
Space Complexity: O(h) for recursion stack where h is height
"""

class Solution:
def strWithout3a3b(self, a: int, b: int) -> str:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def strWithout3a3b(self, a, b):
"""
:type a: int
:type b: int
:rtype: str
"""
```

JavaScript Solution:

```
/**
* Problem: String Without AAA or BBB
* Difficulty: Medium
* Tags: string, tree, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

/**
* @param {number} a
* @param {number} b
* @return {string}
*/
var strWithout3a3b = function(a, b) {

};
```

TypeScript Solution:

```
/**
* Problem: String Without AAA or BBB
* Difficulty: Medium
* Tags: string, tree, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

function strWithout3a3b(a: number, b: number): string {

};
```

C# Solution:

```
/*
* Problem: String Without AAA or BBB
* Difficulty: Medium
* Tags: string, tree, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

public class Solution {
public string StrWithout3a3b(int a, int b) {

}
}
```

C Solution:

```
/*
* Problem: String Without AAA or BBB
* Difficulty: Medium
* Tags: string, tree, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
```

```
* Space Complexity: O(h) for recursion stack where h is height
*/



char * strWithout3a3b(int a, int b){

}
```

Go Solution:

```
// Problem: String Without AAA or BBB
// Difficulty: Medium
// Tags: string, tree, greedy
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height

func strWithout3a3b(a int, b int) string {

}
```

Kotlin Solution:

```
class Solution {
fun strWithout3a3b(a: Int, b: Int): String {

}
}
```

Swift Solution:

```
class Solution {
func strWithout3a3b(_ a: Int, _ b: Int) -> String {

}
}
```

Rust Solution:

```
// Problem: String Without AAA or BBB
// Difficulty: Medium
// Tags: string, tree, greedy
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height

impl Solution {
pub fn str_without3a3b(a: i32, b: i32) -> String {

}
}
```

Ruby Solution:

```
# @param {Integer} a
# @param {Integer} b
# @return {String}
def str_without3a3b(a, b)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer $a
* @param Integer $b
* @return String
*/
function strWithout3a3b($a, $b) {

}
}
```

Scala Solution:

```
object Solution {
def strWithout3a3b(a: Int, b: Int): String = {
```

```
}
}
```