Problem 1641: Count Sorted Vowel Strings

Problem Information

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

Problem Description

Given an integer

n

, return

the number of strings of length

n

that consist only of vowels (

a

,

e

,

i

,

o

,

u

) and are

lexicographically sorted

.

A string

s

is

lexicographically sorted

if for all valid

i

,

s[i]

is the same as or comes before

s[i+1]

in the alphabet.

Example 1:

Input:

n = 1

Output:

5

Explanation:

The 5 sorted strings that consist of vowels only are

["a","e","i","o","u"].

Example 2:

Input:

n = 2

Output:

15

Explanation:

The 15 sorted strings that consist of vowels only are ["aa","ae","ai","ao","au","ee","ei","eo","eu","ii","io","iu","oo","ou","uu"]. Note that "ea" is not a valid string since 'e' comes after 'a' in the alphabet.

Example 3:

Input:

n = 33

Output:

66045

Constraints:

1 <= n <= 50

Code Snippets

C++:

```
class Solution {
public:
int countVowelStrings(int n) {

}
};
```

Java:

```
class Solution {
public int countVowelStrings(int n) {

}
}
```

Python3:

```
class Solution:
def countVowelStrings(self, n: int) -> int:
```

Python:

```
class Solution(object):
def countVowelStrings(self, n):
"""
:type n: int
:rtype: int
"""
```

JavaScript:

```
/**
* @param {number} n
* @return {number}
*/
var countVowelStrings = function(n) {

};
```

TypeScript:

```
function countVowelStrings(n: number): number {

};
```

C#:

```
public class Solution {
public int CountVowelStrings(int n) {

}
}
```

C:

```
int countVowelStrings(int n) {

}
```

Go:

```
func countVowelStrings(n int) int {

}
```

Kotlin:

```
class Solution {
fun countVowelStrings(n: Int): Int {

}
}
```

Swift:

```
class Solution {
func countVowelStrings(_ n: Int) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn count_vowel_strings(n: i32) -> i32 {

}
}
```

Ruby:

```
# @param {Integer} n
# @return {Integer}
def count_vowel_strings(n)

end
```

PHP:

```
class Solution {

/**
* @param Integer $n
* @return Integer
*/
function countVowelStrings($n) {

}
}
```

Dart:

```
class Solution {
int countVowelStrings(int n) {

}
}
```

Scala:

```
object Solution {
def countVowelStrings(n: Int): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec count_vowel_strings(n :: integer) :: integer
def count_vowel_strings(n) do

end
end
```

Erlang:

```
-spec count_vowel_strings(N :: integer()) -> integer().
count_vowel_strings(N) ->
.
```

Racket:

```
(define/contract (count-vowel-strings n)
(-> exact-integer? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Count Sorted Vowel Strings
* Difficulty: Medium
* Tags: string, graph, dp, math, sort
*
* 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 countVowelStrings(int n) {

}
};
```

Java Solution:

```
/**
* Problem: Count Sorted Vowel Strings
* Difficulty: Medium
* Tags: string, graph, dp, math, sort
*
* 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 countVowelStrings(int n) {

}
}
```

Python3 Solution:

```
"""
Problem: Count Sorted Vowel Strings
Difficulty: Medium
Tags: string, graph, dp, math, sort

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 countVowelStrings(self, n: int) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def countVowelStrings(self, n):
"""
:type n: int
:rtype: int
```

```
"""
```

JavaScript Solution:

```
/**
* Problem: Count Sorted Vowel Strings
* Difficulty: Medium
* Tags: string, graph, dp, math, sort
*
* 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} n
* @return {number}
*/
var countVowelStrings = function(n) {

};
```

TypeScript Solution:

```
/**
* Problem: Count Sorted Vowel Strings
* Difficulty: Medium
* Tags: string, graph, dp, math, sort
*
* 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 countVowelStrings(n: number): number {

};
```

C# Solution:

```
/*
* Problem: Count Sorted Vowel Strings
* Difficulty: Medium
* Tags: string, graph, dp, math, sort
*
* 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 CountVowelStrings(int n) {

}
}
```

C Solution:

```
/*
* Problem: Count Sorted Vowel Strings
* Difficulty: Medium
* Tags: string, graph, dp, math, sort
*
* 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 countVowelStrings(int n) {

}
```

Go Solution:

```
// Problem: Count Sorted Vowel Strings
// Difficulty: Medium
// Tags: string, graph, dp, math, sort
//
// 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 countVowelStrings(n int) int {

}
```

Kotlin Solution:

```
class Solution {
fun countVowelStrings(n: Int): Int {

}
}
```

Swift Solution:

```
class Solution {
func countVowelStrings(_ n: Int) -> Int {

}
}
```

Rust Solution:

```
// Problem: Count Sorted Vowel Strings
// Difficulty: Medium
// Tags: string, graph, dp, math, sort
//
// 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 count_vowel_strings(n: i32) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer} n
# @return {Integer}
def count_vowel_strings(n)
```

```

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer $n
* @return Integer
*/
function countVowelStrings($n) {

}
}
```

Dart Solution:

```
class Solution {
int countVowelStrings(int n) {

}
}
```

Scala Solution:

```
object Solution {
def countVowelStrings(n: Int): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec count_vowel_strings(n :: integer) :: integer
def count_vowel_strings(n) do

end
end
```

Erlang Solution:

```
-spec count_vowel_strings(N :: integer()) -> integer().
count_vowel_strings(N) ->
  .
```

Racket Solution:

```
(define/contract (count-vowel-strings n)
  (-> exact-integer? exact-integer?)
  )
```