Problem 344: Reverse String

Problem Information

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

Problem Description

Write a function that reverses a string. The input string is given as an array of characters

s

.

You must do this by modifying the input array

in-place

with

O(1)

extra memory.

Example 1:

Input:

s = ["h","e","l","l","o"]

Output:

["o","l","l","e","h"]

Example 2:

Input:

s = ["H","a","n","n","a","h"]

Output:

["h","a","n","n","a","H"]

Constraints:

1 <= s.length <= 10

5

s[i]

is a

printable ascii character

.

Code Snippets

C++:

```
class Solution {
public:
void reverseString(vector<char>& s) {

}
};
```

Java:

```
class Solution {
public void reverseString(char[] s) {
```

```
}
}
```

Python3:

```
class Solution:
def reverseString(self, s: List[str]) -> None:
"""
Do not return anything, modify s in-place instead.
"""
```

Python:

```
class Solution(object):
def reverseString(self, s):
"""
:type s: List[str]
:rtype: None Do not return anything, modify s in-place instead.
"""
```

JavaScript:

```
/**
* @param {character[]} s
* @return {void} Do not return anything, modify s in-place instead.
*/
var reverseString = function(s) {

};
```

TypeScript:

```
/**
Do not return anything, modify s in-place instead.
*/
function reverseString(s: string[]): void {

};
```

C#:

```
public class Solution {
public void ReverseString(char[] s) {

}
}
```

C:

```
void reverseString(char* s, int sSize) {

}
```

Go:

```
func reverseString(s []byte) {

}
```

Kotlin:

```
class Solution {
fun reverseString(s: CharArray): Unit {

}
}
```

Swift:

```
class Solution {
func reverseString(_ s: inout [Character]) {

}
}
```

Rust:

```
impl Solution {
pub fn reverse_string(s: &mut Vec<char>) {

}
}
```

Ruby:

```
# @param {Character[]} s
# @return {Void} Do not return anything, modify s in-place instead.
def reverse_string(s)

end
```

PHP:

```
class Solution {

/**
* @param String[] $s
* @return NULL
*/
function reverseString(&$s) {

}
}
```

Dart:

```
class Solution {
void reverseString(List<String> s) {

}
}
```

Scala:

```
object Solution {
def reverseString(s: Array[Char]): Unit = {

}
}
```

Solutions

C++ Solution:

```
/*
* Problem: Reverse String
```

```
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public:
void reverseString(vector<char>& s) {

}
};
```

Java Solution:

```
/**
* Problem: Reverse String
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public void reverseString(char[] s) {

}
}
```

Python3 Solution:

```
"""
Problem: Reverse String
Difficulty: Easy
Tags: array, string

Approach: Use two pointers or sliding window technique
```

```
Time Complexity: O(n) or O(n log n)
Space Complexity: O(1) to O(n) depending on approach
"""

class Solution:
def reverseString(self, s: List[str]) -> None:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def reverseString(self, s):
"""
:type s: List[str]
:rtype: None Do not return anything, modify s in-place instead.
"""
```

JavaScript Solution:

```
/**
* Problem: Reverse String
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* @param {character[]} s
* @return {void} Do not return anything, modify s in-place instead.
*/
var reverseString = function(s) {

};
```

TypeScript Solution:

```
/**
* Problem: Reverse String
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
Do not return anything, modify s in-place instead.
*/
function reverseString(s: string[]): void {

};
```

C# Solution:

```
/*
* Problem: Reverse String
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

public class Solution {
public void ReverseString(char[] s) {

}
}
```

C Solution:

```
/*
* Problem: Reverse String
* Difficulty: Easy
* Tags: array, string
*
```

```
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

void reverseString(char* s, int sSize) {

}
```

Go Solution:

```
// Problem: Reverse String
// Difficulty: Easy
// Tags: array, string
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

func reverseString(s []byte) {

}
```

Kotlin Solution:

```
class Solution {
fun reverseString(s: CharArray): Unit {

}
}
```

Swift Solution:

```
class Solution {
func reverseString(_ s: inout [Character]) {

}
}
```

Rust Solution:

```
// Problem: Reverse String
// Difficulty: Easy
// Tags: array, string
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

impl Solution {
pub fn reverse_string(s: &mut Vec<char>) {

}
}
```

Ruby Solution:

```
# @param {Character[]} s
# @return {Void} Do not return anything, modify s in-place instead.
def reverse_string(s)

end
```

PHP Solution:

```
class Solution {

/**
* @param String[] $s
* @return NULL
*/
function reverseString(&$s) {

}
}
```

Dart Solution:

```
class Solution {
void reverseString(List<String> s) {

}
}
```

Scala Solution:

```
object Solution {
def reverseString(s: Array[Char]): Unit = {

}
}
```