Reverse String

- Take 2 pointers.
- 1 at start and other at last.
- Swap both.
- Increase start pointer.
- Decrease last pointer.
- Continue till first is less than last.

Reverse string

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.

Examples

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.

Approach: Two-Pointer Swap

Use two pointers: one from the start (`i`) and one from the end (`j = len - i - 1`).
Swap characters at those indices.
Continue until the pointers meet or cross.

Dry Run:

Input: s = ['h', 'e', 'l', 'l', 'o']

Initial Setup: len = 5, halfLen = Math.floor(5 / 2) = 2
Loop from i = 0 to i < 2

Step-by-step Execution:

i = 0:
temp = s[0] = 'h'
s[0] = s[4] = 'o'
s[4] = temp = 'h'
Result: ['o', 'e', 'l', 'l', 'h']
i = 1:
temp = s[1] = 'e'
s[1] = s[3] = 'l'
s[3] = temp = 'e'
Result: ['o', 'l', 'l', 'e', 'h']

Final Output: ['o', 'l', 'l', 'e', 'h']

Time Complexity:

O(n) — The loop runs floor(n / 2) times, each doing a constant-time swap.

Space Complexity:

O(1) — In-place reversal using one temporary variable.

JavaScript | C++ | C | Java | Python

```javascript
var reverseString = function(s) {
  let len = s.length;
  let halfLen = Math.floor(len / 2);

  for (let i = 0; i < halfLen; i++) {
    let temp = s[i];
    s[i] = s[len - i - 1];
    s[len - i - 1] = temp;
  }
};
```

```javascript
var reverseString = function (s) {
    const n = s.length;
    let l = 0;
    let r = n - 1;

    while (l < r) {
        [s[l], s[r]] = [s[r], s[l]]; // JS Swapping
        l++;
        r--;
    }
};
```