🔷 Problem Explanation (Simple Words)

You are given a **string s**.
Your task is to **reverse only the vowels** in the string and keep all other characters **in the same position**.

👉 Vowels = a e i o u A E I O U

🔷 Example

Input:

s = "hello"

Output:

"holle"

Only vowels e and o are reversed.

🔷 Dry Run (Example: "leetcode")

Original string: l e e t c o d e

Vowels found : e e o e

Positions of vowels:

index: 1, 2, 5, 7

Reverse vowels order:

e o e e

Final string:

l e o t c e d e

✅ Output: "leotcede"

✅ Approach 1: Using ArrayList (Collect & Replace)

🔷 Idea

1. Traverse the string and **store all vowels** in an ArrayList.
2. Reverse the ArrayList.
3. Traverse the string again and **replace vowels** using the reversed list.

🔷 Steps

1. Convert string → char array
2. Store vowels in ArrayList
3. Reverse the list
4. Replace vowels in original string

🔷 **Time & Space Complexity**

- **Time:** O(n)
- **Space:** O(n) (for ArrayList)

🔷 **C++ Code (ArrayList / Vector)**

```
class Solution {
public:
    bool isVowel(char c) {
        return string("aeiouAEIOU").find(c) != string::npos;
    }

    string reverseVowels(string s) {
        vector<char> vowels;

        for (char c : s) {
            if (isVowel(c)) vowels.push_back(c);
        }

        reverse(vowels.begin(), vowels.end());

        int idx = 0;
        for (char &c : s) {
            if (isVowel(c)) {
                c = vowels[idx++];
            }
        }
```

```
        return s;
    }
};
```

◈ Java Code (ArrayList)

```
class Solution {
    public boolean isVowel(char c) {
        return "aeiouAEIOU".indexOf(c) != -1;
    }

    public String reverseVowels(String s) {
        ArrayList<Character> list = new ArrayList<>();

        for (char c : s.toCharArray()) {
            if (isVowel(c)) list.add(c);
        }

        Collections.reverse(list);

        char[] arr = s.toCharArray();
        int idx = 0;
        for (int i = 0; i < arr.length; i++) {
            if (isVowel(arr[i])) {
                arr[i] = list.get(idx++);
            }
        }

        return new String(arr);
    }
}
```

🔷 Python Code (List)

```
class Solution:
    def reverseVowels(self, s: str) -> str:
        vowels = set("aeiouAEIOU")
        lst = [c for c in s if c in vowels]

        lst.reverse()
        res = []
        idx = 0

        for c in s:
            if c in vowels:
                res.append(lst[idx])
                idx += 1
            else:
                res.append(c)

        return "".join(res)
```

✅ Approach 2: Using Character Array (128) + Two Pointers (Optimized)

🔷 Idea

- Use a **boolean array of size 128** to mark vowels.
- Use **two pointers (l, r)**
- Swap vowels when both pointers point to vowels.

👉 This avoids extra space like ArrayList.

🔷 Steps

1. Mark vowels in char[128]
2. Convert string → char array
3. Use two pointers:
    - Move left until vowel

- Move right until vowel
- Swap

🔷 **Time & Space Complexity**

- **Time:** O(n)
- **Space:** O(1) ✅ (constant)

🔷 **C++ Code (Two Pointers)**

```
class Solution {
public:
    string reverseVowels(string s) {
        vector<bool> isVowel(128, false);
        string v = "aeiouAEIOU";
        for (char c : v) isVowel[c] = true;

        int l = 0, r = s.size() - 1;

        while (l < r) {
            while (l < r && !isVowel[s[l]]) l++;
            while (l < r && !isVowel[s[r]]) r--;

            swap(s[l], s[r]);
            l++; r--;
        }
        return s;
    }
};
```

🔷 **Java Code (Two Pointers)**

```
class Solution {
    public String reverseVowels(String s) {
```

```
        boolean[] isVowel = new boolean[128];
        for (char c : "aeiouAEIOU".toCharArray()) {
            isVowel[c] = true;
        }

        char[] arr = s.toCharArray();
        int l = 0, r = arr.length - 1;

        while (l < r) {
            while (l < r && !isVowel[arr[l]]) l++;
            while (l < r && !isVowel[arr[r]]) r--;

            char temp = arr[l];
            arr[l] = arr[r];
            arr[r] = temp;

            l++; r--;
        }
        return new String(arr);
    }
}
```

🔷 Python Code (Two Pointers)

```
class Solution:
    def reverseVowels(self, s: str) -> str:
        vowels = [False] * 128
        for c in "aeiouAEIOU":
            vowels[ord(c)] = True

        s = list(s)
        l, r = 0, len(s) - 1
```

```
    while l < r:
        while l < r and not vowels[ord(s[l])]:
            l += 1
        while l < r and not vowels[ord(s[r])]:
            r -= 1

        s[l], s[r] = s[r], s[l]
        l += 1
        r -= 1

    return "".join(s)
```

🔥 **Final Comparison**

Approach	Time	Space	Interview Preferred
ArrayList	O(n)	O(n)	❌
Two Pointers + Array	O(n)	O(1)	✅ BEST