2744. Find Maximum Number of String Pairs(LEETCODE)

Method 1 — O(n^2) brute force

```
for (int i=0; i<arr.size(); i++) {
    string rev = arr[i];
    reverse(rev.begin(), rev.end()); // current word ka reverse bana
    for (int j=i+1; j<arr.size(); j++) {
        if (rev == arr[j]) count++; // agar koi match mila to count++
    }
}
return count;
```

Logic

- Har word ka reverse banao
- Baaki saare words ke saath compare karo
- Agar reverse match milta hai → ek pair ban gaya
- i < j maintain hota hai kyunki loop j = i+1 se chal raha hai

Dry Run (Example: `["cd","ac","dc","ca","zz"]`)

- `i=0`: "cd" → reverse = "dc"
  - compare with index 1 → "ac" ❌
  - compare with index 2 → "dc" ✅ count = 1
  - compare with 3,4 → ❌
- `i=1`: "ac" → reverse = "ca"
  - compare with index 2 → "dc" ❌
  - compare with index 3 → "ca" ✅ count = 2

- compare with index 4 → ❌

- Remaining i=2,3,4 me koi naya match nahi milta

  Output → 2

Method 2 — Set + Erase

```
unordered_set<string> s;
for (int i=0; i<arr.size(); i++) {
    s.insert(arr[i]); // sab words set me daale
}
for (int i=0; i<arr.size(); i++) {
    string rev = arr[i];
    reverse(rev.begin(), rev.end());
    if (rev == arr[i]) continue; // agar palindromic word hai to skip
    if (s.find(rev) != s.end()) {
        count++;
        s.erase(arr[i]); // current ko hata diya taaki dobara count na ho
    }
}
```

Logic

- Pehle sab words ek set me store kar liye (O(1) lookup)
- Har word ka reverse check kiya
- Agar reverse set me mila → pair found
- Current word ko erase kiya taaki dobara na count ho
- Palindrome words skip hue kyunki unka reverse wahi hota hai

Dry Run (Example: `["cd","ac","dc","ca","zz"]`)

- Set: {"cd","ac","dc","ca","zz"}
- i=0: "cd" → reverse "dc" found ✅ count=1, erase "cd"
- i=1: "ac" → reverse "ca" found ✅ count=2, erase "ac"

- i=2: "dc" → reverse "cd" ❌ (kyunki erase ho chuka hai)
- i=3: "ca" → reverse "ac" ❌ (erase ho chuka hai)
- i=4: "zz" palindrome → skip

  Output → 2

Method 3 — Single Pass

```
unordered_set<string> s;
for (int i=0; i<arr.size(); i++) {
    string rev = arr[i];
    reverse(rev.begin(), rev.end());
    if (s.find(rev) != s.end()) {
        count++;
    }
    else {
        s.insert(arr[i]);
    }
}
```

Logic

- Ek pass me kaam ho jaata hai
- Jab current word ka reverse pehle se set me hai → pair found → count++
- Agar nahi hai → current word ko set me daal do future matches ke liye
- Har word max ek baar hi count hoga

Dry Run (Example: `["cd","ac","dc","ca","zz"]`)

- s = {}
- i=0: "cd" → reverse "dc" ❌ → insert "cd" → s={"cd"}
- i=1: "ac" → reverse "ca" ❌ → insert "ac" → s={"cd","ac"}
- i=2: "dc" → reverse "cd" ✅ count=1
- i=3: "ca" → reverse "ac" ✅ count=2

- i=4: "zz" → reverse "zz" ❌ (not in set yet) → insert "zz" → s={"cd","ac","zz"}

  Output → 2

Bhai summary:

- **Method 1:** Simple brute force, easy to understand, O(n^2)
- **Method 2:** Faster O(n), but 2 passes (insert + check)
- **Method 3:** Fastest and cleanest O(n), single pass