☑ PAT 乙级题目讲解：1008《数组元素循环右移问题》

🧩 题目简介

本题要求将一个长度为 N 的整数数组循环右移 M 位。所谓“循环右移”，即将数组后 M 个元素移至最前面，其余元素顺序后移。**题目特别要求不能使用额外数组空间（即就地操作）**，这使得我们必须在原数组上完成操作，且尽量减少数据移动次数。

🧪 样例分析

输入：

```
6 2
1 2 3 4 5 6
```

分析过程：

- 原数组：[1, 2, 3, 4, 5, 6]
- 循环右移 2 位后变为：[5, 6, 1, 2, 3, 4]

输出为：

```
5 6 1 2 3 4
```

🔍 解题思路

📌 变量说明

变量名	含义
`n`	数组长度
`m`	右移位数（注意需 `m %= n`）
`a[i]`	第 `i` 个数组元素

☑方法一：暴力法（不推荐）

每次循环将最后一个元素插入最前面，其余元素依次后移，重复 M 次。时间复杂度为 $O(MN)$，效率极低，题目已明确不推荐此法。

☑方法二：高效正解 —— 三次翻转法

📌 核心思想：

循环右移 M 位等价于：

- 把数组分成两段：前 $n-m$ 个元素，后 m 个元素；
- 将这两段分别翻转；
- 最后整体再翻转一次，即可达到目标。

🧠 推导过程：

以 `n = 6, m = 2` 为例：

原数组：`[1 2 3 4 5 6]`

分段：

- 前段 A = `[1 2 3 4]`
- 后段 B = `[5 6]`

执行三次翻转：

1. 翻转前段：`[4 3 2 1][5 6]`
2. 翻转后段：`[4 3 2 1][6 5]`
3. 整体翻转：`[5 6][1 2 3 4]` ☑

☑ 解法一：手写 `reverse` 函数

📌 函数设计

我们手动实现一个区间翻转函数：

```
void reverse(int l, int r){
    for(int i = l, j = r; i < j; i++, j--){
        swap(a[i], a[j]);
    }
}
```

注意：

- 区间为闭区间 `[l, r]`；
- 使用双指针向中间逼近、两两交换。

☑ 完整代码（手写翻转函数）

```
#include <bits/stdc++.h>
using namespace std;

int n, m, a[105];

void reverse(int l, int r){ // [l, r]
    for(int i = l, j = r; i < j; i++, j--){
        swap(a[i], a[j]);
    }
}
```

```cpp
int main(){
    cin >> n >> m;
    m %= n; // 防止 m >= n 越界

    for(int i = 0; i < n; i++){
        cin >> a[i];
    }

    // 三次翻转
    reverse(0, n - m - 1);      // 翻转前 n-m 段
    reverse(n - m, n - 1);      // 翻转后 m 段
    reverse(0, n - 1);          // 整体翻转

    for(int i = 0; i < n; i++){
        cout << a[i] << (i < n - 1 ? " " : "");
    }
    return 0;
}
```

☑ 解法二：使用 STL `reverse` 函数

📌 参数注意事项

- `reverse(a, a + n)` 使用的是地址指针；
- 其参数是**左闭右开区间**，即 `[first, last)`；
- 想翻转 `[L, R]`，实际写法是 `reverse(a + L, a + R + 1)`。

☑ 完整代码（调用标准库）

```cpp
#include <bits/stdc++.h>
using namespace std;

int n, m, a[105];

int main(){
    cin >> n >> m;
    m %= n;

    for(int i = 0; i < n; i++){
        cin >> a[i];
    }

    // 三次翻转（STL）
    reverse(a, a + n - m);        // 前段 [0, n-m-1]
    reverse(a + n - m, a + n);    // 后段 [n-m, n-1]
    reverse(a, a + n);            // 整体翻转

    for(int i = 0; i < n; i++){
        cout << a[i] << (i < n - 1 ? " " : "");
    }
    return 0;
}
```

常见错误提醒

错误类型	具体表现
未对 `m` 取模	可能导致下标越界（`m %= n` 必不可少）
输出末尾多空格	需通过 `i < n - 1` 控制空格输出
STL 参数写错	`reverse(a, a + n)` 是地址区间，不是下标
自定义 reverse 范围错误	注意是否为闭区间 `[l, r]`，双指针写法容易 off-by-one

总结归纳

- 本题核心是理解**循环右移 = 三段翻转**；
- 推荐熟练掌握两种写法（手写 + STL）；
- 此技巧广泛用于数组与字符串的旋转操作；
- 时间复杂度 $O(n)$，空间复杂度 $O(1)$，高效且符合题目要求。

思维拓展

- 类似方法可处理 **循环左移** 问题，翻转顺序不同；
- 三次翻转思想也可应用于字符串、链表、矩阵行列等结构；
- 思考如何将该算法推广到部分旋转、滑动窗口等问题。