2026年8月 SACCC 四级题解

苏州市人工智能学会 · 青少年人工智能核心算法素养考核（SACCC）

级别：四级 期次：2026年8月

题目清单

顺序	题目	主要考点
1	小组积分看板	二维数组、行列统计
2	营员排行榜	结构体、多关键字排序
3	展板数字旋转	二维数组、循环移动
4	爬楼梯	递推
5	学员记录校验	记录校验、去重、结构体排序
6	队伍顺序检查	逆序对、冒泡排序交换计数

第 1 题 小组积分看板

主要考点： 二维数组、行列统计

题意概括

给定一张 n 行 m 列的积分表，需要分别找出总分最高的小组和总分最高的项目；若总分相同，选择编号较小者。

核心思路

读入每个积分时，同时累加它所在行的总分和所在列的总分。之后分别扫描行和列寻找最大值。更新答案时只使用严格大于号，这样出现并列时不会覆盖先找到的较小编号。

参考代码

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

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n, m;
    cin >> n >> m;
    vector<vector<int>> a(n, vector<int>(m));
    vector<int> row(n, 0), col(m, 0);
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            cin >> a[i][j];
            // 一个积分同时计入对应小组和对应项目。
            row[i] += a[i][j];
            col[j] += a[i][j];
        }
    }
    int bestRow = 0, bestCol = 0;
    // 只在严格更大时更新，并列时自然保留编号较小者。
    for (int i = 1; i < n; ++i) if (row[i] > row[bestRow]) bestRow = i;
    for (int j = 1; j < m; ++j) if (col[j] > col[bestCol]) bestCol = j;
    cout << bestRow + 1 << ' ' << row[bestRow] << '\n';
    cout << bestCol + 1 << ' ' << col[bestCol] << '\n';
    return 0;
}
```

第 2 题 营员排行榜

主要考点： 结构体、多关键字排序

题意概括

根据成绩生成营员排行榜：分数高者靠前，分数相同时用时少者靠前，二者仍相同时保持原报名顺序。

核心思路

用结构体同时保存姓名、分数、用时和输入顺序。排序比较函数按三个关键字依次判断：分数降序、用时升序、输入编号升序。排序完成后依次输出姓名即可。

参考代码

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

struct Player {
    string name;
    int score;
    int usedTime;
    int id;
};

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n;
    cin >> n;
    vector<Player> a(n);
    for (int i = 0; i < n; ++i) {
        cin >> a[i].name >> a[i].score >> a[i].usedTime;
        a[i].id = i;
    }
    sort(a.begin(), a.end(), [](const Player &x, const Player &y) {
        // 依次比较分数、用时和报名顺序。
        if (x.score != y.score) return x.score > y.score;
        if (x.usedTime != y.usedTime) return x.usedTime < y.usedTime;
        return x.id < y.id;
    });
    for (const auto &p : a) cout << p.name << '\n';
    return 0;
}
```

第 3 题 展板数字旋转

主要考点： 二维数组、循环移动

题意概括

给定一个 n × n 方阵，第 i 行需要向右循环移动 i × k 个位置，输出所有行移动后的方阵。

核心思路

每一行的移动互不影响，可以单独处理。右移次数先对行长度取模；把原行复制一份后，原下标 i 的元素应放到新下标 (i+shift)%n，这样能直接得到循环移动结果。

参考代码

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

void rotateRight(vector<int> &row, int shift) {
    int n = row.size();
    // 整行右移 n 次后恢复原状，只需保留余数。
    shift %= n;
    vector<int> old = row;
    // 原位置 i 右移 shift 后到达 (i + shift) % n。
    for (int i = 0; i < n; ++i) row[(i + shift) % n] = old[i];
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n, k;
    cin >> n >> k;
    vector<vector<int>> a(n, vector<int>(n));
    for (int i = 0; i < n; ++i)
        for (int j = 0; j < n; ++j) cin >> a[i][j];
    for (int i = 0; i < n; ++i) rotateRight(a[i], (i + 1) * k);
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            if (j) cout << ' ';
            cout << a[i][j];
        }
        cout << '\n';
    }
    return 0;
}
```

第 4 题 爬楼梯

主要考点： 递推

题意概括

小明每次可以走 1 级或 2 级台阶，需要求从地面走到第 n 级台阶的不同走法数量，并对 1000000007 取模。

核心思路

到达第 i 级的最后一步只可能来自第 i-1 级或第 i-2 级，所以有递推式 f[i]=f[i-1]+f[i-2]。初始值是 f[1]=1、f[2]=2，计算时只保留前两项即可把额外空间降为常数。

参考代码

```
#include <bits/stdc++.h>
using namespace std;
const long long MOD = 1000000007LL;

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n;
    cin >> n;
    if (n == 1) {
        cout << 1 << '\n';
        return 0;
    }
    long long x = 1, y = 2;
    for (int i = 3; i <= n; ++i) {
        // x、y 分别保存 f[i-2] 和 f[i-1]。
        long long z = (x + y) % MOD;
        x = y;
        y = z;
    }
    cout << y << '\n';
    return 0;
}
```

第 5 题 学员记录校验

主要考点： 记录校验、去重、结构体排序

题意概括

依次校验导入的学员记录，过滤编号或成绩不合法以及有效编号重复的记录；随后按成绩降序、编号升序输出有效记录中的前 k 名。

核心思路

按照输入顺序处理记录，用布尔数组标记已经出现过的有效编号。只有编号、成绩和去重条件全部满足时才保存记录并登记编号，否则计入异常数。最后对有效记录按排行榜规则排序，并输出不超过 k 条结果；没有有效记录时输出 NONE。

参考代码

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

struct Student {
    int id;
    string name;
    int score;
};

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    int n, k;
    cin >> n >> k;

    vector<Student> students;
    vector<bool> used(10000, false);
    int invalidCount = 0;

    for (int i = 0; i < n; ++i) {
        int id, score;
        string name;
        cin >> id >> name >> score;

        // 短路判断保证只有合法范围内的 id 才会访问 used[id]。
        if (id < 1000 || id > 9999 || score < 0 || score > 100 || used[id]) {
            ++invalidCount;
            continue;
        }

        used[id] = true;
        students.push_back({id, name, score});
    }

    sort(students.begin(), students.end(), [](const Student &a, const Student &b) {
        // 成绩降序；同分时编号升序。
        if (a.score != b.score) return a.score > b.score;
        return a.id < b.id;
    });

    cout << students.size() << ' ' << invalidCount << '\n';
    if (students.empty()) {
        cout << "NONE\n";
        return 0;
    }

    int limit = min(k, (int)students.size());
    for (int i = 0; i < limit; ++i) {
        cout << students[i].id << ' ' << students[i].name << ' ' << students[i].score << '\n';
    }
    return 0;
}
```

第 6 题 队伍顺序检查

主要考点： 逆序对、冒泡排序交换计数

题意概括

给定当前队伍编号顺序，统计满足“前面编号大于后面编号”的下标对数量，也就是整个序列的逆序对数。

核心思路

冒泡排序每次交换一对相邻且顺序错误的元素，恰好会使逆序对数量减少 1；排序结束时逆序对数变为 0。因此，完成冒泡排序所做的相邻交换总次数就是原序列的逆序对数。答案可能较大，需要使用 long long。

参考代码

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

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

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

    long long ans = 0;
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j + 1 < n - i; ++j) {
            if (a[j] > a[j + 1]) {
                // 每次相邻逆序交换恰好消去一个逆序对。
                swap(a[j], a[j + 1]);
                ++ans;
            }
        }
    }

    cout << ans << '\n';
    return 0;
}
```