青少年软件编程（C 语言）等级考试试卷（三级）

分数：100 题数：5

一、编程题(共 5 题，共 100 分)

1. 分组均衡性

在上机实验课上，老师将所有学生排列为 n 排，每排坐 m 个学生。每个学生有左右两个邻座（除了这一排的左右两端）。每个人可以和自己的邻座互相帮助完成实验。除了每排左右两端的学生，中间的每个学生都可以同时与两个邻座分别协作。

由于每个学生的个人能力不同，假设协作产生的小组能力值是两个协作学生的能力值之和，老师希望知道，自己给出的座位安排在多大程度上是“均衡”的 —— 所谓**分组均衡性**，是指所有可能组成的协作小组的能力值的最大值与最小值之差。

给定一张座位安排表，请计算这个安排的分组均衡性。

时间限制：1000
内存限制：65536

输入
输入第一行给出 2 个正整数 n 和 m（2 ≤ n, m ≤ 100），依次为座位的排数和每排的人数。 随后 n 行，每行给出 m 个数字，代表对应座位上学生的能力值（为区间 [1, 100] 内的整数）。同行数字间以空格分隔。

输出
在一行中输出分组均衡性。

样例输入

```
3 5
10 80 30 95 60
79 55 63 84 41
98 23 72 85 58
```

样例输出

```
67
```

提示
样例解释：最强组合是第 3 排的 72+85=157；最弱组合是第 1 排的 10+80=90。因此两者之差为 67。

试题编号：20250308-3-01
试题类型：编程题

标准答案：
试题难度：一般
试题解析：
展示地址：点击浏览
考生答案：

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

int main() {
    int n, m, a[110], b = 0, s = 1000000, bt, st, c;
    cin >> n >> m;
    for (int i = 1; i <= n; i++) {
        for (int j = 0; j < m; j++) {
            cin >> a[j];
        }
        for (int j = 1; j < m ; j++) {
            c = a[j] + a[j - 1];
            if (c > b) {
                b = c;
            }
            if (c < s) {
                s = c;
            }
        }
    }
    cout << b - s;
    return 0;
}
```

考生得分：20
是否评分：已评分
评价描述：
2. 狼人杀简单版

以下文字摘自《灵机一动·好玩的数学》：“狼人杀”游戏分为狼人、好人两大阵营。在一局“狼人杀”游戏中，1 号玩家说：“2 号是狼人”，2 号玩家说：“3 号是好人”，3 号玩家说：“4 号是狼人”，4 号玩家说：“5 号是好人”，5 号玩家说：“4 号是好人”。已知这 5 名玩家中有 2 人扮演狼人角色，有 2 人说的不是实话，有狼人撒谎但并不是所有狼人都在撒谎。扮演狼人角色的是哪两号玩家？

本题是这个问题的升级版：已知 n 名玩家中有 2 人扮演狼人角色，有 2 人说的不是实话，有狼人撒谎但并不是所有狼人都在撒谎。要求你找出扮演狼人角色的是哪几号玩家？

时间限制：1000
内存限制：65536

输入

输入在第一行中给出一个正整数 n（5 ≤ n ≤ 100）。随后 n 行，第 i 行给出第 i 号玩家说的话（1 ≤ i ≤ n），即一个玩家编号，用正号表示好人，负号表示狼人。

输出

如果有解，在一行中按递增顺序输出 2 个狼人的编号，其间以空格分隔，行首尾不得有多余空格。如果解不唯一，则输出最小序列解 —— 即对于两个序列 A = { a[1], ..., a[M] } 和 B = { b[1], ..., b[M] }，若存在 0 ≤ k < M 使得 a[i]=b[i] （i ≤ k），且 a[k+1] < b[k+1]，则称序列 A 小于序列 B。若无解则输出 `No Solution`。

样例输入

样例#1：

```
5
-2
+3
-4
+5
+4
```

样例#2：

```
6
+6
+3
+1
-5
-2
+4
```

样例#3：

```
5
-2
-3
-4
-5
```

```
-1
```

样例输出

```
样例#1：

1 4


样例#2：

1 5


样例#3：

No Solution
```

试题编号：20250308-3-02
试题类型：编程题
标准答案：
试题难度：一般
试题解析：
展示地址：点击浏览
参考程序：

```
#include <iostream>
#include <vector>

// 检查当前假设的狼人组合是否满足条件
bool isValidCombination(const std::vector<int>& statements, int wolf1, int wolf2) {
    int n = statements.size();
    std::vector<int> identities(n + 1, 1);
    identities[wolf1] = identities[wolf2] = -1;
    int lieCount = 0;
    int wolfLies = 0;

    for (int i = 1; i <= n; ++i) {
        int target = std::abs(statements[i - 1]);
        int claim = statements[i - 1] > 0 ? 1 : -1;

        if (claim != identities[target]) {
            ++lieCount;
```

```
            if (i == wolf1 || i == wolf2) {
                ++wolfLies;
            }
        }
    }

    return lieCount == 2 && wolfLies > 0 && wolfLies < 2;
}

int main() {
    int n;
    std::cin >> n;
    std::vector<int> statements(n);

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

    for (int i = 1; i <= n; ++i) {
        for (int j = i + 1; j <= n; ++j) {
            if (isValidCombination(statements, i, j)) {
                std::cout << i << " " << j << std::endl;
                return 0;
            }
        }
    }

    std::cout << "No Solution" << std::endl;
    return 0;
}
```

考生得分：0
是否评分：已评分
评价描述：

3. 增高垫

网传最大号的增高垫有 16 厘米，可以一下把 1.64 米的男孩变成一米八的大汉，不少对自己身高缺乏自信的男生女生会悄悄买来穿，特别是大家站成一排拍照的时候。

假设准备拍照的一排人中，只要有人看到身边紧挨着的人比自己高，就会忍不住穿增高垫，并且一定要比人家多穿一层。你的任务就是在看过这一排人的身高后，算出谁穿了最多层的增高垫。

时间限制：1000
内存限制：65536

输入
输入首先在第一行给出正整数 n（≤ 10⁴），为一排人的个数。随后一行给出 n 个正整数，表示 n 个人的身高（厘米）。每个数值是不超过 300 的正整数，数字间以空格分隔。

输出
在一行中输出穿了最多层增高垫的人的位置和穿的层数（位序从左到右，从 1 开始）。如果有并列，按从左到右的顺序，每个人的信息占一行。

样例输入

```
10
150 160 186 200 170 175 180 186 186 183
```

样例输出

```
1 3
5 3
```

试题编号：20250308-3-03
试题类型：编程题
标准答案：
试题难度：一般
试题解析：
展示地址：点击浏览
参考程序：

```
#include <iostream>
#include <vector>

int main() {
    int n;
    std::cin >> n;
    std::vector<int> heights(n);
    std::vector<int> mats(n, 0);

    // 读取每个人的身高
    for (int i = 0; i < n; ++i) {
        std::cin >> heights[i];
    }

    bool updated;
    do {
        updated = false;
        for (int i = 0; i < n; ++i) {
```

```
            int left = (i > 0) ? heights[i - 1] + mats[i - 1] : 0;
            int right = (i < n - 1) ? heights[i + 1] + mats[i + 1] : 0;
            int newMat = std::max(0, std::max(left, right) - heights[i] + 1);
            if (newMat > mats[i]) {
                mats[i] = newMat;
                updated = true;
            }
        }
    } while (updated);

    int maxMats = 0;
    for (int mat : mats) {
        if (mat > maxMats) {
            maxMats = mat;
        }
    }

    for (int i = 0; i < n; ++i) {
        if (mats[i] == maxMats) {
            std::cout << i + 1 << " " << maxMats << std::endl;
        }
    }

    return 0;
}
```

考生得分：0
是否评分：已评分
评价描述：
4. 三元组的离心力

三元组 (a,b,c) 的“离心力”定义为 |a-b|+|b-c|+|c-a|。现给定三个非空的整数集合 S_1、S_2、S_3，请你找出所有跨这三个集合的三元组 (a,b,c)（即 a∈ S_1，b∈ S_2，c∈ S_3）的**最小**离心力。

时间限制：1000
内存限制：65536

输入
输入第一行给出三个不超过 10^4 的正整数 n_1、n_2、n_3，依次为集合 S_1、S_2、S_3 中元素的个数（注意一个集合内没有相同的元素）。随后三行，顺次给出三个集合的元素，均为 [-10^4，10^4] 内的整数。同行数字间以空格分隔。

输出

在一行中输出 `LiXinLi(a, b, c) = d`，其中 `(a, b, c)` 是具有最小离心力的三元组，`d` 是对应的离心力。如果解不唯一，输出那个最大的解。 注意：$(a_1,a_2,a_3)>(b_1,b_2,b_3)$ 是指，存在 $1 \le k \le 3$ 使得 $a_i=b_i$ 对 $1 \le i < k$ 成立，并且 $a_k > b_k$。

样例输入

```
4 4 6
0 9 -1 11
10 -25 11 -10
9 2 41 17 12 30
```

样例输出

```
LiXinLi(11, 11, 12) = 2
```

提示

注意到样例实际上有两组解，另外一组解是 (9, 10, 9)。因为 (11, 11, 12) 比较大，所以输出的是这组解。

试题编号：20250308-3-04
试题类型：编程题
标准答案：
试题难度：一般
试题解析：
展示地址：点击浏览
参考程序：

```
#include <iostream>
#include <vector>
#include <climits>

int main() {
    int n1, n2, n3;
    std::cin >> n1 >> n2 >> n3;

    std::vector<int> S1(n1), S2(n2), S3(n3);

    for (int i = 0; i < n1; ++i) {
        std::cin >> S1[i];
    }
    for (int i = 0; i < n2; ++i) {
        std::cin >> S2[i];
    }
```

```
    for (int i = 0; i < n3; ++i) {
        std::cin >> S3[i];
    }

    int minForce = INT_MAX;
    int bestA, bestB, bestC;

    for (int a : S1) {
        for (int b : S2) {
            for (int c : S3) {
                int force = std::abs(a - b) + std::abs(b - c) + std::abs(c - a);
                  if (force < minForce || (force == minForce && (a > bestA || (a ==
bestA && (b > bestB || (b == bestB && c > bestC)))))) {
                    minForce = force;
                    bestA = a;
                    bestB = b;
                    bestC = c;
                }
            }
        }
    }

    std::cout << "LiXinLi(" << bestA << ", " << bestB << ", " << bestC << ")
= " << minForce << std::endl;

    return 0;
}
```

考生得分：0
是否评分：已评分
评价描述：

5. 分玩具

已知 n 位小朋友对 m 件玩具的喜好（n ≤ m），现要将 m 件玩具分给 n 位小朋友，每位小朋友只能分到 1 件玩具，每件玩具也最多只能分给 1 位小朋友，并且还要求每位小朋友都能分到自己喜欢的玩具。

本题请你对任意 n 和 m 尝试列出所有满足要求的方案。

时间限制：5000
内存限制：65536

输入

输入第一行给出两个正整数 n 和 m（n ≤ m ≤ 8），即小朋友人数和玩具的数量。 随后 n 行，每行给出 m 个数字。其中第 i 行第 j 个数字为 1 表示第 i 位小朋友喜欢第 j 件玩具，为 0 则表示不喜欢。

输出

按升序列出所有满足要求的方案，格式为 $(s_1, \ldots, s_n)$。其中 s_i 表示第 i 位小朋友分到了第 s_i 件玩具。 注：方案 $(a_1, \ldots, a_n) < (b_1, \ldots, b_n)$ 是指存在 $1 \le k \le n$，使得 $a_i = b_i$ 对所有 $1 \le i < k$ 成立，并且有 $a_k < b_k$。

样例输入

```
4 5
0 1 0 0 1
1 1 0 1 0
1 0 1 1 0
0 0 0 1 1
```

样例输出

```
(2, 1, 3, 4)
(2, 1, 3, 5)
(2, 1, 4, 5)
(2, 4, 1, 5)
(2, 4, 3, 5)
(5, 1, 3, 4)
(5, 2, 1, 4)
(5, 2, 3, 4)
```

试题编号：20250308-3-05
试题类型：编程题
标准答案：
试题难度：一般
试题解析：
展示地址：点击浏览
参考程序：

```
#include <iostream>
#include <vector>
#include <algorithm>

int main() {
    int n, m;
```

```
    std::cin >> n >> m;

    std::vector<std::vector<int>> preferences(n, std::vector<int>(m));
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            std::cin >> preferences[i][j];
        }
    }

    std::vector<int> scheme;
    for (int i = 1; i <= m; ++i) {
        scheme.push_back(i);
    }

    do {
        bool valid = true;
        std::vector<bool> used(m + 1, false);
        for (int i = 0; i < n; ++i) {
            int toy = scheme[i];
            if (!preferences[i][toy - 1] || used[toy]) {
                valid = false;
                break;
            }
            used[toy] = true;
        }

        if (valid) {
            std::cout << "(";
            for (int i = 0; i < n; ++i) {
                if (i > 0) {
                    std::cout << ", ";
                }
                std::cout << scheme[i];
            }
            std::cout << ")" << std::endl;
        }
    } while (std::next_permutation(scheme.begin(), scheme.end()));

    return 0;
}
```

考生得分：0
是否评分：已评分
评价描述：