

全真模拟赛精讲稿 · Day 3

上午全真模拟 · 下午逐题精讲 (14:00–17:00)

配套主题：线性 DP · 背包 DP · 区间 DP
极客少儿编程教研组 · 2026 年 10 月

本场概览

题号	题目	文件名	考点	难度	暴力分	满分算法
T1	程老师的加密器	cipher	模拟 / 字符串	红 ★☆	100	按规则一遍扫描
T2	程老师的洗牌机	shuffle	过程模拟	橙 ★★	100	数组模拟 $O(nk)$
T3	程老师的值日表	duty	组合计数 / 杨辉三角	黄 ★★★	30	插空法 + 递推取模
T4	程老师的回文招牌	sign	区间 DP / 方案恢复	绿 ★★★	30	LPS + 字典序记忆化恢复

暴力分合计 260 分：T1 规则模拟、T2 过程模拟，两道模拟题即满分，T3 子集枚举 30 分，T4 子序列枚举 30 分。本场前两题守住基本盘，后两题正面对撞下午专题——组合计数里的递推取模、区间 DP，全是 DP 主线。

T1 程老师的加密器 (cipher.cpp / cipher.in / cipher.out)

考点	难度	暴力分	时限 / 内存
模拟 / 字符串	红题 ★☆	100	1000ms / 512MB

题意复述

程老师有一台“数字加密器”，加密规则只有两道，全部写在题面里。请你替他把关：给定一个学号（纯数字串，可能以 0 开头），加密器会输出什么？

规则只有两道：① 每位数字 d 变成 $9-d$ ；② 把变换后产生的所有 0 删掉，删光了就输出 0。这题不考算法，考严谨——规则的内容和先后，一个都不能错。

输入输出与数据范围

一行一个数字串 s ($1 \leq |s| \leq 10^5$ ，仅含数字 0~9)；输出一行结果串。

规模估算：一遍扫描 $O(|s|)$ ， 10^5

长度毫无压力——唯一要做的就是严格按题面规则执行，一道工序都不能漏。

样例与解释

样例输入	样例输出
123	876
909	9
999	0

样例 2: $9 \rightarrow 0$ 、 $0 \rightarrow 9$ 、 $9 \rightarrow 0$ ，得到 090，删去两个 0 剩 9，输出 9。

样例 3: 999 全部变为 0，删光后一个字符不剩，按规则输出 0——这是本题唯一的特判。

部分分策略（先保底，再冲刺）

按题面规则自己实现即可——规则简单，写对就是 100 分。本题暴力与正解同一条路，实测最大点 1.8ms。

教学点：规则模拟题的正确姿势：先把两道规则各读两遍、手工把样例 2 ($909 \rightarrow 090 \rightarrow 9$) 推一遍，确认两道工序的先后再动手——先删 0 再变换，和先变换再删 0，是完全不同的两个程序。

满分思路

两道工序可以合并成一遍扫描：对每个字符 c 算出 $9 - (c - '0')$ ，结果不为 0 就拼入答案串；全部处理完若答案为空则输出 0。

注意删除的对象是“变换后的 0”，也就是原串里的 9——原串的 0 变换后是 9，永远删不掉。想反了就是错解警示里的第一类。

参考代码（C++，NOI Linux 下 -O2 编译通过）

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

int main() {
    freopen("cipher.in", "r", stdin);
    freopen("cipher.out", "w", stdout);
    ios::sync_with_stdio(false);
    cin.tie(0); cout.tie(0);

    string s;
    cin >> s;

    // 工序1: 对每个数位做变换, 9 - 原数字
    string t;
    for (char c : s) {
        int d = c - '0';
        int nd = 9 - d;
        if (nd != 0) {
            t += char('0' + nd);
        }
    }

    // 工序2: 如果结果为空, 输出 "0"
    if (t.empty()) {
        t = "0";
    }

    cout << t << '\n';
    return 0;
}
```

错解警示（均为实测得分）

典型错解	错在哪里	实测得分
先删 0 再变换（工序反了）	090 应先变 909 再删 0 得 99；工序反了输出全错——两道工序的先后就是本题本身	0
变换后保留 0 不删	909 应输出 9，错解输出 090；全 9 输入时输出 10^5 个 0——漏掉整道工序	15

变式拓展

- 变换规则改成 $d \rightarrow (d+3) \bmod 10$ ：规则换了，"变换 + 删除"的框架一行不动。
- 思考题：给出输出能不能反推输入？——不能，被删掉的 0 无法还原；体会"不可逆变换"的含义。
- 对照 Day 2 T1 负二进制：两题都是"规则给你、照做就行"，区别只在规则一个写成数学语言、一个写成操作步骤——都要先手工跑样例再动手。

T2 程老师的洗牌机 (shuffle.cpp / shuffle.in / shuffle.out)

考点	难度	暴力分	时限 / 内存
过程模拟	橙题 ★★	100	1000ms / 512MB

题意复述

一叠牌 n 张，从上到下编号 $1 \sim n$ 。洗牌机每次工作两步：均分——从中间分成上半叠和下半叠， n 为奇数时上半叠多一张（上半叠 $n/2$ 张）；交错——先放上半叠第 1 张、再下半叠第 1 张、然后上半叠第 2 张……直到放完。

全新牌叠（ $1 \sim n$ 顺序）连续洗 k 次，输出最终从上到下的编号顺序。

两步规则各埋一个坑：奇数时哪半叠多一张；交错时谁先放。

输入输出与数据范围

一行两个整数 n, k ($\leq 10^3$)；输出一行 n 个整数。

$O(n \cdot k)$ 最多 10^6 次操作——直接模拟就是正解。

样例与解释

样例输入	样例输出
6 1	1 4 2 5 3 6
5 1	1 4 2 5 3

样例 1：上半叠 [1,2,3]、下半叠 [4,5,6]，交错得 1,4,2,5,3,6。

样例 3（奇数关键）：上半叠 [1,2,3]（多一张）、下半叠 [4,5]，交错得 1,4,2,5,3——最后一张来自上半叠。

部分分策略（先保底，再冲刺）

每轮洗牌：mid = (n+1)/2 为上半叠张数；依次把上半叠第 i 张 b[p++] = a[i]、若 i+mid < n 则下半叠第 i 张 b[p++] = a[i+mid]；最后把 b 拷回 a。重复 k 轮。

实测最慢 4ms, 100 分。

教学点：模拟题的"样例"

3"往往是命题人专门准备的奇偶/边界提示——写完代码务必把奇数样例走一遍，本题的 mid 写错只在 n 为奇数时暴露。

满分思路

正解与暴力完全一致，O(nk)。两个关键写法：

- ① mid = (n+1)/2——整数除法天然下取整，(n+1)/2 恰好是 ceil(n/2)；
- ② 交错循环只遍历上半叠下标 i < mid，下半叠牌 i+mid < n 才放——奇数时下半叠少一张，这个判断自动处理，不需要分支讨论。

参考代码（C++，NOI Linux 下 -O2 编译通过）

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

const int N = 1e3 + 10;
int a[N], b[N];

int main() {
    freopen("shuffle.in", "r", stdin);
    freopen("shuffle.out", "w", stdout);
    ios::sync_with_stdio(false);
    cin.tie(0); cout.tie(0);

    int n, k;
    cin >> n >> k;
    for (int i = 0; i < n; i++) a[i] = i + 1;

    while (k--) {
        int mid = (n + 1) / 2; // 上半叠的张数
        int p = 0;
        for (int i = 0; i < mid; i++) {
            b[p++] = a[i]; // 上半叠的第 i 张
            if (i + mid < n) // 下半叠还有牌
                b[p++] = a[i + mid]; // 下半叠的第 i 张
        }
        for (int i = 0; i < n; i++) a[i] = b[i];
    }

    for (int i = 0; i < n; i++) {
        cout << a[i];
        if (i < n - 1) cout << ' ';
    }
    cout << '\n';
    return 0;
}
```

错解警示（均为实测得分）

典型错解	错在哪里	实测得分
------	------	------

mid = n/2 (下半叠多一张)	奇数 n 时上下半叠颠倒, 结果全错; 偶数 n 时碰巧正确——得的分全是偶数点	25
交错时先放下半叠第一张	交错顺序搞反, 即使偶数 n 也全错——规则方向性错误是毁灭性的	0

变式拓展

- 探究题: 对 $n = 6$ 动手跑一跑, 洗几次牌序回到 1~6? (周期的存在性——本题性质 B 的数据就是 "k 恰好是周期倍数", 答案原样输出。)
- 对照魔术术语: 本题是 out-shuffle 的一种变体; 若改成 "先放下半叠第一张" (in-shuffle), 周期会变——两种都实现一遍, 比较周期。
- 思考题: 若 $k \leq 10^9$ 怎么办? (找出洗牌周期 T, 只洗 $k \bmod T$ 次——模拟题的大数据出路往往是找周期。)

T3 程老师的值日表 (duty.cpp / duty.in / duty.out)

考点	难度	暴力分	时限 / 内存
组合计数 / 杨辉三角	黄题 ★★★	30	1000ms / 512MB

题意复述

学期共 n 天, 要选 k 天值日, 硬性规定: 任意两个值日不能相邻。求合法方案数, 对 $10^9 + 7$ 取模。 $k = 0$ 时方案数为 1。

一句话模型: 从 $1 \sim n$ 中选 k 个互不相邻的数, 方案数是多少?

输入输出与数据范围

一行两个整数 n, k ($n \leq 2000, 0 \leq k \leq n$); 输出方案数 $\bmod 10^9 + 7$ 。

看到 $10^9 + 7$ 的第一反应: 所有加法递推全程取模——组合数在 $n=2000$ 时高达数百位, long long 完全装不下。

样例与解释

样例输入	样例输出
5 2	6
4 3	0

样例 1: 合法选法 $\{1,3\} \{1,4\} \{1,5\} \{2,4\} \{2,5\} \{3,5\}$, 共 6 种。

样例 3: 4 天选 3 天还要互不相邻, 办不到——答案 0。边界要先会判。

部分分策略 (先保底, 再冲刺)

2^n 枚子集: 位掩码枚举每个子集, 检查 "恰好 k 个且互不相邻"。 $n \leq 20$ 时约 10^6

个子集, 轻松通过测试点 1~6, 收下 30 分; $n = 2000$ 时指数爆炸。

教学点: 组合计数题的暴力形态就是子集枚举——写得快、拿得到小数据分, 同时它的枚举过程能帮你在草稿纸上验证小样例 ($n=4$ 手列对照)。

满分思路

插空法建模：先把 $n-k$ 个休息日排成一排，它们之间及两端一共形成 $(n-k)+1$ 个空位；往空位里放 k 个值日，每个空位至多放一个 任意两个值日不相邻。所以答案 = $C(n-k+1, k)$ ；当 $k > n-k+1$ 时方案数为 0。组合数计算：用杨辉三角递推 $C(i, j) = (C(i-1, j-1) + C(i-1, j)) \bmod (10^9 + 7)$ ， $O(n^2)$ 预处理到 $n =$

2001——这本质上就是一张 DP 表：状态 $C(i, j)$ 、转移是两项相加。复杂度 $O(n^2) \approx 4 \times 10^6$ ，轻松满分。本题数据分档里 $n \leq 20$ 留给暴力、 $n = 2000$ 考验递推与取模。

参考代码（C++，NOI Linux 下 -O2 编译通过）

```
#include <bits/stdc++.h>
using namespace std;
#define int long long
const int N = 2e3 + 10;
const int MOD = 1e9 + 7;
int T = 1;

int c[N][N];

void solve() {
    int n, k;
    cin >> n >> k;
    memset(c, 0, sizeof(c));
    for (int i = 0; i <= n + 1; i++) {
        c[i][0] = c[i][i] = 1;
        for (int j = 1; j < i; j++)
            c[i][j] = (c[i - 1][j - 1] + c[i - 1][j]) % MOD;
    }
    int m = n - k + 1;
    if (k > m) cout << 0 << "\n";
    else cout << c[m][k] << "\n";
}

signed main() {
    freopen("duty.in", "r", stdin);
    freopen("duty.out", "w", stdout);
    ios::sync_with_stdio(false);
    cin.tie(0); cout.tie(0);
    while (T--) solve();
    return 0;
}
```

错解警示（均为实测得分）

典型错解	错在哪里	实测得分
无视不相邻约束直接 $C(n, k)$	多算了所有相邻方案： $n=5, k=2$ 时输出 10 而非 6；仅 $k=1$ 时碰巧与 $C(n-1+1, 1)$ 相等	10
建模正确但不取模	杨辉三角递推不取 mod， n 稍大 long long 溢出——小数据全对、大数据全错，最隐蔽的错法	30

变式拓展

- 推广练习：规定改成“任意两个值日至少隔 2 天”——每个值日生后面跟 2 个休息日再插空，答案变成 $C(n-2(k-1), k)$ 的形式——自己动手推一遍插空法的参数变化。

- 思考题：若 $n \leq 10^5$ ，杨辉三角 $O(n^2)$ 不行——需要阶乘 + 费马小定理求逆元（提高组内容，只点名）；但插空建模的结论 $C(n-k+1, k)$ 不变，建模与计算是两件事。
- 环形变式： n 天围成一圈选 k 天不相邻（首尾也算相邻）——用“第 1 天选/不选”分两类拆开，各化归成链上的插空。

T4 程老师的回文招牌 (sign.cpp / sign.in / sign.out)

考点	难度	暴力分	时限 / 内存
区间 DP / 方案恢复	绿题 ★★★	30	1000ms / 512MB

题意复述

给定小写字母串

s ，可以删掉若干字母（剩下的保持原先后顺序），使剩下的串正读反读相同（回文）。求：① 最少删几个字母；② 在删最少的所有方案中，输出字典序最小的回文结果。

一句话模型：删最少字母留回文 求最长回文子序列（LPS）——删除数 = $n - \text{LPS}$ 长度；第二问要求恢复字典序最小的那条 LPS。

输入输出与数据范围

一行一个字符串 s ($1 \leq |s| \leq 300$)；输出两行：最少删除数、字典序最小的回文串。 $|s| \leq 20$ 的前 6 个点留给子集枚举； $|s| \leq 300$ 时 $O(n^2)$ 甚至 $O(n^3)$ 都能过。

样例与解释

样例输入	样例输出
abca	1 aba
abcba	0 abcba

样例 1：删 c 得 aba ，删 b 得 aca ——都只删 1 个； aba 与 aca 首位相同，第二位 $b < c$ ，故输出 aba 。字典序比较发生在“同样最长”的方案之间。

样例 2：原串本身回文，删 0 个，原样输出。

部分分策略（先保底，再冲刺）

2^n 枚举所有子序列（位掩码），逐个检查是否回文，记录最长且字典序最小者。

$|s| \leq 20$ 时约 10^6 个子序列，通过测试点 1~6，收下 30 分。

教学点：这个暴力同时还解决了“方案恢复”——它在枚举中天然拿到了具体串。对比正解：DP 求长度容易、恢复方案要另写一遍，考场上如果 $|s|$ 小，暴力反而更省事。

满分思路

第一步：区间 DP 求长度。设 $dp[l][r]$ = 子串 $s[l..r]$ 内的最长回文子序列长度：

- ① 若 $s[l] == s[r]$ ： $dp[l][r] = dp[l+1][r-1] + 2$ （两端同时保留）；

② 否则: $dp[l][r] = \max(dp[l+1][r], dp[l][r-1])$ (必删其一, 取较优)。按区间长度从小到大递推 ($len = 2..n$ 外层循环), $O(n^2)$ 。最少删除数 $= n - dp[1][n]$ 。

第二步: 字典序最小的方案恢复。记忆化递归 $rec(l, r)$ 返回子区间 $s[l..r]$

内"最长且字典序最小"的回文子序列: 三个候选——两端相等时配对 ($s[l] + rec(l+1, r-1) + s[r]$)、删左端 $rec(l+1, r)$ 、删右端 $rec(l, r-1)$, 按"先比长度、同长比字典序"取最优。每个子区间构造一次字符串, $O(n^3) \approx 2.7 \times 10^7$, $|s| \leq 300$ 时完全可行。

与区间 DP 专题的衔接: 这就是区间 DP 的标准骨架——定义在区间 $[l, r]$ 上的状态、按区间长度递推。石子合并 (P1880) 是同骨架的另一道题, 吃透本题再去看它, 只剩"环形"一层窗户纸。

参考代码 (C++, NOI Linux 下 -O2 编译通过)

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

const int N = 305;
int dp[N][N];
string dp2[N][N];
bool vis[N][N];
string s;
int n;

// 记忆化重构: 返回 s[l..r] 内最长回文子序列中字典序最小的一个
string rec(int l, int r) {
    if (l > r) return "";
    if (l == r) return string(1, s[l]);
    if (vis[l][r]) return dp2[l][r];
    vis[l][r] = true;

    string bst = "";

    // 选法一: s[l] == s[r] 时两端配对
    if (s[l] == s[r]) {
        string mid = rec(l + 1, r - 1);
        bst = s[l] + mid + s[r];
    }

    // 选法二: 跳过左端
    {
        string cnd = rec(l + 1, r);
        if (cnd.size() > bst.size() ||
            (cnd.size() == bst.size() && cnd < bst))
            bst = cnd;
    }

    // 选法三: 跳过右端
    {
        string cnd = rec(l, r - 1);
        if (cnd.size() > bst.size() ||
            (cnd.size() == bst.size() && cnd < bst))
            bst = cnd;
    }

    return dp2[l][r] = bst;
}
```

```

int main() {
    freopen("sign.in", "r", stdin);
    freopen("sign.out", "w", stdout);
    ios::sync_with_stdio(false);
    cin.tie(0); cout.tie(0);

    cin >> s;
    n = s.size();
    s = " " + s; // 下标从 1 开始

    // 区间 DP: dp[l][r] = s[l..r] 内最长回文子序列长度
    for (int i = 1; i <= n; i++) dp[i][i] = 1;
    for (int i = 1; i < n; i++) {
        dp[i][i + 1] = (s[i] == s[i + 1]) ? 2 : 1;
    }

    for (int len = 3; len <= n; len++) {
        for (int l = 1; l + len - 1 <= n; l++) {
            int r = l + len - 1;
            if (s[l] == s[r]) {
                dp[l][r] = dp[l + 1][r - 1] + 2;
            } else {
                dp[l][r] = max(dp[l + 1][r], dp[l][r - 1]);
            }
        }
    }

    int del = n - dp[1][n];
    cout << del << "\n";
    cout << rec(1, n) << "\n";

    return 0;
}

```

错解警示（均为实测得分）

典型错解	错在哪里	实测得分
双指针贪心	两端相等则保留、不等则固定删右端——删哪端需要全局信息，贪心没有；小数据与回文多的点碰巧得分	35
DP 长度对、方案恢复错	恢复时固定"删左优先"而不比较字典序——长度全对、串半错，评分按点判，串错的点一分没有	30

变式拓展

- 对照题 P1880 石子合并：环形区间 DP——破坏成链（复制一倍）后与本题同骨架，也是 DP 专题的重点题。
- 辨析：最长回文子串（连续）与最长回文子序列（可跳跃）是两个不同问题，前者可以中心扩展，后者必须区间 DP——两种各写一遍，体会差别。
- 删题练习：若只问最少删除数（不要求输出方案），代码能砍掉一半——体会"计数/最优值"与"方案恢复"在 DP 里是两层工作。

本场得分策略小结

- 基本盘 260 分：T1 (100) + T2 (100) 靠模拟基本功，T3 子集枚举 (30) + T4 子序列枚举 (30)。
- 考场时间建议：T1 用 15 分钟（写完先测样例 2 的 909 和全 9 边界）；T2 用 20 分钟（奇数样例必走一遍）；T3 先写 2^n 枚举保底，再推插空公式；T4 先写子序列枚举保底 30，再分别攻"长度 DP"与"方案恢复"——两问可以分开拿分。
- 检查清单：T1 的两道工序顺序（先变换后删 0）与删光输出 0 的特判；T2 的 $\text{mid} = (n+1)/2$ ；T3 的取模与 $k > n-k+1$ 判 0；T4 输出是两行（删除数 + 回文串），别漏第一行。
- 与 DP 专题的衔接：T3 的杨辉三角就是一张递推 DP 表（计数型 DP）；T4 是区间 DP 的标准样板，直通石子合并与能量项链；背包主线则由 Day 1 的补给站、Day 4 的野餐篮共同呼应。

本场题眼：T1 工序顺序就是答案本身，先推样例再动手；T2 模拟题的正确性靠奇数样例背书；T3 "不相邻"翻译成"插空"，大组合数必取模；T4 区间 DP 按长度递推，方案恢复用记忆化搜索最省事。