```
出题人简要题解
T1
手玩或暴力打表打表就可以找到规律
T2
bfs+五位状压或五维数组 col[2][2][2][2][2]
T3
拆点+动态规划，dp[i][j] 表示休息 i-1 次后到达点j的最优解
T4
法一：ST表（或线段树）维护区间极差、差分数组的gcd，离散化判重
法二（其实是错解，但是出题人懒得卡了）：数据结构维护平方和
```

T1 全排列问题

不难发现，答案只由 n 决定，与具体的排列完全没关系。

若 $n \leq 3$，我们能得到全部 $n!$ 种不同的排列。

若 $n > 3$ 且 n 是 4 的倍数，我们能得到 4 种不同的排列：

1. 第一种排列：对原排列不进行任何操作；
2. 第二种排列：对原排列进行一次操作1；
3. 第三种排列：对原排列进行一次操作2；
4. 第四种排列：对原排列进行操作1和操作2各一次。

若 $n > 3$ 且 n 被 4 除的余数是 1，则案是 $2n$。第一个排列是原排列，记为 p_1；对 p_1 进行一次操作1，得到新的排列 p_2；对 p_2 进行一次操作2，得到新的排列 p_3。按这种方式操作下去就能得到 $2n$ 种不同的排列。找规律容易发现这个结论。大家可以想想有没有简单的证明方法。

若 $n > 3$ 且 n 被 4 除的余数是 2，则答案是 n。

若 $n > 3$ 且 n 被 4 除的余数是 3，则答案是 12。

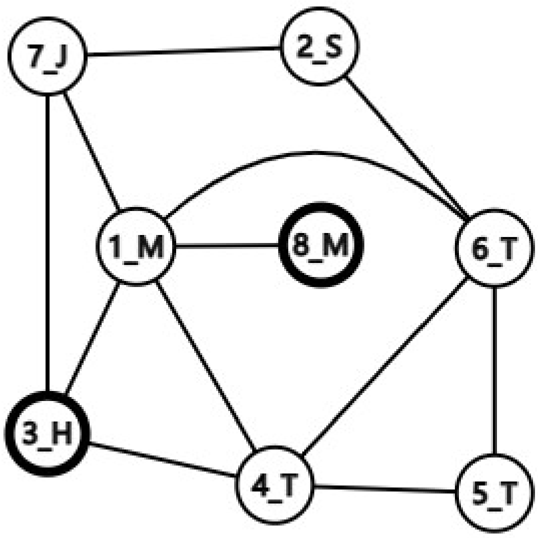
打表代码：

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

int n;

vector<int> op1(vector<int> &p) { // 对排列p进行操作1，返回操作后的排列
    auto res = p;
    for (int i = 0; i < n / 2; ++i)
        swap(res[i], res[(n + 1) / 2 + i]);
    return res;
}
vector<int> op2(vector<int> &p) { // 对排列p进行操作2，返回操作后的排列
    auto res = p;
    for (int i = 1; i < n; i += 2)
        swap(res[i], res[i - 1]);
    return res;
}

int main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
```

```
    for (n = 1; n <= 1000; ++n) {
        // bfs
        queue<vector<int>> q;
        map<vector<int>, bool> vis;
        vector<int> a(n);
        iota(a.begin(), a.end(), 1); // 初始排列为a=[1,2,3,...,n]
        q.push(a);
        vis[a] = true;
        int ans = 0;
        while (!q.empty()) {
            auto &u = q.front();
            ++ans;
            auto v = op1(u);
            if (!vis[v]) vis[v] = true, q.push(v);
            v = op2(u);
            if (!vis[v]) vis[v] = true, q.push(v);
            q.pop();
        }

        cout << "n = " << n << ", ans = " << ans << '\n';
    }
}
```

T2 甜蜜蜜

某个下发样例图示：



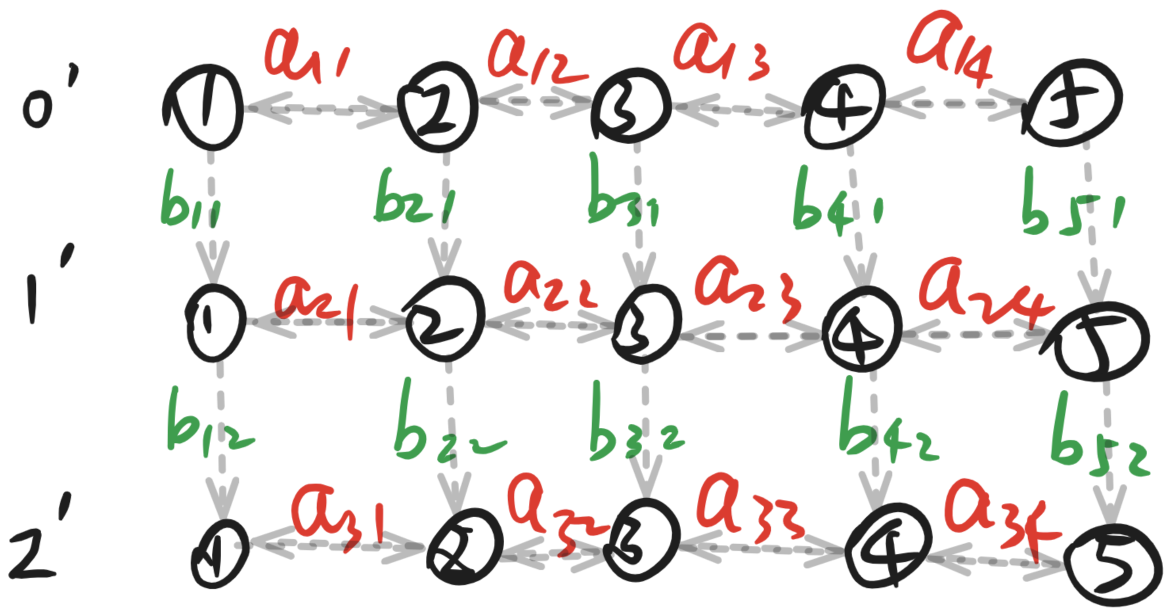
分别考虑每个人。对第 i 个人，bfs计算 `dis[u][S]` 表示第 i 个人走到点 u，最后一个周期中走过的点的属性为 S 的最短时间。S 是一个 5 位二进制非负整数，S 的第 0 位为 1 表示最后一个周期中走过了一个属性为金的点，S 的第 1 位为 1 表示最后一个周期中走过了一个属性为木的点，以此类推。若 $S=(11111)_2=31$，则下一步可以开始一个新的周期。

复杂度 $O(Tkn\times 2^5)$，代码细节比较多，记得清空数组。

T3 逃离魔爪

分层图dp。

题中的下标关系让人头晕。不妨画个图。下图为 $n=5, m=3$ 的情况，第 i 行表示休息了 i 次，$0 \le i \le m-1$.



题意变为：有 m 行 n 列共 mn 个点，相邻点之间有带权边，问从第 0 行的点 1 出发，每次可以向下或向左或向右走一步，走到任意一行的点 n 的最短距离。

$f(i,j)$ 表示走到第 i 行的点 j 的最短距离。相邻行之间的边是单向边，因此有 $f(i,j) = f(i-1,j) + b_{j,i}$

同一行内，相邻列之间的边是双向的，但边权非负。这意味着若一条边被重复走多次，答案不会更优。也就是说路径在一行内要么只从左往右走，要么只从右往左走，不会改变方向。从左往右递推一遍，再从右往左递推一遍即可。

数组可以只开一维，时间复杂度 $O(nm)$.

T4 意义

本题强制在线，必须顺序回答每个询问。

在什么情况下，区间 $[l,r]$ 中的数能排成公差为 k 的等差数列?

首先，区间中的数的差分的 gcd 必须为 k.

> 设区间中最小的数为 a_0，则区间中的数可以写成 $a_0 + b_1 k, a_0 + b_2 k, \cdots$，其中 b_i 是非负整数。
>
> b_i 的最小值是 0.

然后，若公差 k 不为 0，则区间中的数必须两两不同。

> 这就是说，b_i 两两不同。

最后，区间极差（最大值减去最小值）必须等于公差 k 的 $r-l$ 倍。

> 这就是说，b_i 的最大值是 $r-l$.
>
> 因此，b_i 取遍 $0,1,2,\cdots,r-l+1$ 中的每一个数。原数组是公差为 k 的等差数列。

以上信息都能用ST表维护。公差为 0 的情况可能需要特殊判断。时间复杂度 $O(n \log n + q)$.

这题的加强版：[P5278 算术天才⑨与等差数列]