题目大意

给定两个 k 维字符立方体 A, B。
求 B 作为连续子立方体在 A 中的出现次数。
$k \le 5, 1 \le m_i \le n_i \le 8$。

题目分析：子任务 1

$k = 1$ 时，原问题为字符串匹配问题。

[redacted]

```
for (int i = 0; i <= n[0] - m[0]; i++) {
    match = True;
    for (int j = 0; j < m[0]; j++)
        match &= A[i + j] == B[j];
    ans += match;
}
```

题目分析：子任务 2

$k = 2$ 时，多加一维循环即可解决。

[redacted]

```
for (int i0 = 0; i0 <= n[0] - m[0]; i0++)
    for (int i1 = 0; i1 <= n[1] - m[1]; i1++) {
        match = True;
        for (int j0 = 0; j0 < m[0]; j0++)
            for (int j1 = 0; j1 < m[1]; j1++)
                match &= A[(i0 + j0) * n[1] + (i1 + j1)] == B[j0 * m[1] + j1];
                // a[i0 + j0][i1 + j1] == b[j0][j1]
        ans += match;
    }
```

题目分析

$k \leq 5$ 时，暴力枚举复杂度为 $O(n^k m^k)$，可以通过。
写两个五维循环？

[redacted]

```
for (int i0 = 0; i0 <= n[0] - m[0]; i0++)
    ... {
        match = True;
        for (int j0 = 0; j0 < m[0]; j0++)
            ...
                match &= A[((i0 + j0) * n[1] + ...) + (ik + jk)] == B[(j0 * m[1] + ...) + jk
                    ];
                // a[i0 + j0]...[ik + jk] == b[j0]...[jk]
        ans += match;
    }
```

题目分析

处理高维问题时，可以考虑 dfs 实现。

[redacted]

```
void check(int now, int x, int y) {
    if (now == k) return match &= A[x] == B[y], void();
    for (int i = 0; i < m[now]; i++) {
        q[now] = i, check(now + 1, x * n[now] + (p[now] + q[now]), y * m[now] + q[now]);
        if (!match) return;
    }
}

void dfs(int now) {
    if (now == k) return match = 1, check(0, 0, 0), ans += match, void();
    for (int i = 0; i <= n[now] - m[now]; i++)
        p[now] = i, dfs(now + 1);
}
```

题目大意

给定带边权简单有向图，满足：

- 每个点至多存在于一个简单环内；
- 任意两个点间至多存在一条简单路径。

求起点 S 到终点 T 的 k 短路长度。

$n \le 50, 1 \le w \le 9, 1 \le k \le 10^{12}$。

题目分析：子任务 1

原问题无解当且仅当以下两者之一发生：

- S 无法到达 T;
- S 到 T 恰有一条路径，且 $k > 1$。

该子任务可以考虑的实现方式：

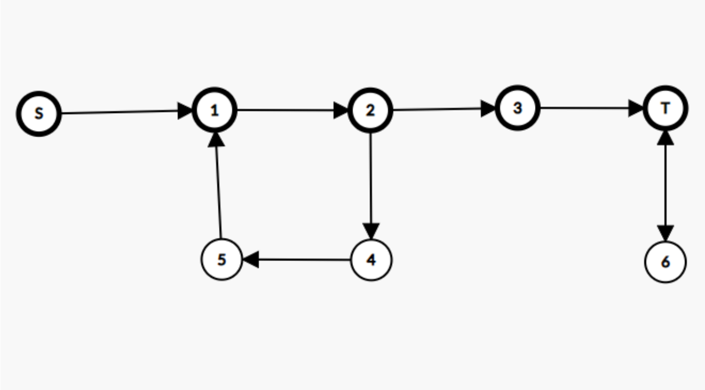
- dijkstra，每次从边权和最短的路径出发移动一步；
- 二分答案，搜索剪枝；
- 随机游走；
- …

题目分析

找到 S 到 T 的唯一简单路径 p。
假设存在路径 p 之外的，可以被 S 到达，且可以到达 T 的环 c'。考察离 S, T 最近的 c' 中的点 s' 和 t'。
若 S 到 s' 的路径与 t' 到 T 的路径有交，则存在包含交点、s' 和 t' 的环，从而 s' 处在至少两个环中，与题设矛盾；否则，$S \to s' \to t' \to T$ 是一条简单路径，从而 S 到 T 存在两条简单路径，与题设矛盾。
故所有可以被 S 到达，且可以到达 T 的环，都与路径 p 有交。

题目分析

故整张图有用的部分，由路径 p 和其上挂的若干条互不相交的环组成。



题目分析

对于一条 S 到 T 的路径，考虑其经过路径 p 上的每个环的次数。
由于访问每个环的顺序是固定的，因此每个环经过次数构成的序列与 S 到 T 的路径一一对应。
预处理求出路径 p 上所有环的边权之和，问题转化为：

- 给定 $m \leq 25$ 个物品，所有物品的权值和 ≤ 450，求第 k 小的完全背包的权值。

题目分析：子任务 2

子任务特殊限制等价于 $m = 1$。
第 k 小的背包权值即为 $(k-1) \times$ 物品权值。

题目分析：子任务 3

子任务特殊限制等价于 $m \leq 2$。不妨假设 $m = 2$。
设两个物品的权值分别为 $a > b$，答案即为可重集合 $\{ax + by | x, y \geq 0\}$ 的第 k 小的元素。
二分答案 mid，相当于求

$$\sum_{0 \leq x,y} [ax + by \leq mid] = \sum_{0 \leq x \leq \lfloor \frac{mid}{a} \rfloor} \lfloor \frac{mid - ax}{b} \rfloor$$

注意到权值 $\leq mid$ 的路径至少有 $\binom{\lfloor \frac{mid}{a} \rfloor + m}{m}$ 条。
故答案有上界 $2a\sqrt{k}$，直接枚举物品 a 的出现次数即可。

题目分析：子任务 4

子任务特殊限制等价于 $m \geq 3$。
进一步地，设所有物品权值和为 s，原问题的答案有上界 $s\sqrt[m]{k}$。
当 $m \geq 3$ 时，原问题答案有 4.5×10^6 的上界，可以直接背包 dp：

- 记 $f_{i,s}$ 为，考虑了前 i 个物品，权值和为 s 的背包方案数。
- 类似完全背包地转移即可。

[redacted]

```
for (int i = 1; i <= m; i++)
    for (int s = w[i]; s <= mid; s++)
        if ((f[s] += f[s - w[i]]) >= k) {
            mid = s;
            break;
        }
```

题目大意

有一个大小为 $n \times n$ 的方格图，左下角为 $(0,0)$ 右上角为 (n,n)。
要求从左下角开始每次都只能向右或者向上走一单位，不走到对角线 $y=x$ 上方（但可以触碰）的情况下到达右上角，且第 i 步的移动方向必须与第 p_i 步相同（$p_i \le i$）。
求合法路径条数。
$n \le 25$

题目分析：子任务 1

dfs 枚举每一步向右走还是向上走，时间复杂度 $O(2^{2n})$。

题目分析

若对于所有 i 都有 $p_i = i$，原问题为卡特兰数。
将节点 p_i 视作节点 i 的父亲，将 i 向 p_i 连边，可以得到一棵限制关系构成的树。
对这棵树路径压缩。原问题等价于，这 $2n$ 步被划分为了若干组，每组内的移动方向必须相同。
不妨假设第 i 步属于第 c_i 组。

题目分析：子任务 3

考虑一个新问题：假设某几步的移动方向已经固定了，剩下的移动方向没有限制，如何求合法路径条数。
我会动态规划！
记 $f_{i,j}$ 为，走了 i 步，走到了 $x-y=j$ 的格子 (x,y) 的方案数。
时间复杂度 $O(n^2)$。

题目分析：子任务 3

若第 c_i 组只有一个元素 i，则第 i 步的移动方向没有限制；
包含大于一个元素的组，至多只有 n 个。
暴力枚举包含大于一个元素的组对应的移动方向，动态规划计算合法路径条数，时间复杂度 $O(n^2 2^n)$。

题目分析

折半搜索，将前 n 步和后 n 步分开考虑：

- dfs 枚举前 n 步的移动方向，存储得到的结果；后 n 步同理。
- 考虑如何合并前 n 步和后 n 步的结果。

对于一条前 n 步对应路径 p 和一条后 n 步对应路径 q，
p 和 q 能组合得到一条合法路径当且仅当：

- 路径 p 对应的终点和路径 q 对应的起点相同；
- 对于所有同时出现在前 n 步和后 n 步的“组”，其在 p 和 q 中对应的移动方向的相同。

题目分析

由于这样的组至多只有 n 个，且移动方向仅有 `UR` 两种，
可以用一个长度至多为 n 的二进制数 S_p 描述路径 p 中每个这样的组的移动方向，路径 q 同理。
记路径 p 和 q 对应的终点和起点分别是 r_p, r_q，答案即为

$$\sum_{p,q}[(r_p, S_p) == (r_q, S_q)]$$

枚举 r_p，将所有 S_p 存进值域 2^n 的桶里，然后枚举所有 $r_q = r_p$ 的 S_q 计算答案即可。
时间复杂度 $O(2^n)$。

[redacted]

```
for (p : ALL_p) L[r_p].push_back(S_p);
for (q : ALL_q) R[r_q].push_back(S_q);
for (r : ALL_r) {
    for (S_p : L[r]) cnt[S_p]++;
    for (S_q : R[r]) ans += cnt[S_q];
    for (S_p : L[r]) cnt[S_p]--;
}
```

题目大意

对长为 n 的序列 a，构造长为 $n+1$ 的序列 $b=\{b_0,\cdots,b_n\}$ 使得 $b_i=\lfloor\frac{\sum_{j=1}^{i}a_j}{10}\rfloor$。
对可重集 S，重排 S 中元素得到序列 a，记对应的序列 b 的不同元素个数的最大值为 $f(S)$。
给定可重集合 A，对其所有本质不同的子集 B，求 $f(B)$ 之和对 998244353 取模后的值。
$n\le 2\times 10^3$，$1\le a_i\le 11$

题目分析：子任务 1

对于集合 A 的每个本质不同子集 B，$f(B)$ 的取值可以通过枚举 a_B 的最后一个数，递归求出。
这个过程可以记忆化，时间复杂度 $O(n^2 3^{\frac{n}{3}})$。

题目分析

先考虑如何求 $f(S)$。
首先，答案有上界 $1+|S|$。
将问题看作最大化前缀和进位次数，易得 $s=1+\lfloor\frac{\sum x\in S}{10}\rfloor$ 是答案的上界。
我们期望 $1,\cdots,s$ 中作为前缀和除以 10 下取整被取到的数的个数尽量多，等价于希望避免出现前缀和的十位的增量 >1 的情况，即避免在前缀和的个位数为 9 时将 11 作为 a 中的下一个数。

题目分析

考虑以下依次确定 a 每一位的贪心策略：

- 若前缀和个位不是 9，且 11 尚未用完，将 11 作为 a 的下一个数；
- 否则，任选一个不是 11 的数作为 a 的下一个数。若仅剩 11，则结束过程。

若过程结束后 11 用完了，则答案取到了上界 $1+s$。否则，序列 a 任意位置的前缀和都进了位，故答案取到了另一个上界 $1+|S|$。

故我们有 $f(S)=1+\min\{|S|,\lfloor\frac{\sum x\in S}{10}\rfloor\}$。

题目分析

故对于集合 S，我们只关心其元素个数 $|S|$ 和所有元素的和 sum。
记 $g(i,j)$ 为，A 的本质不同子集中，有多少个大小为 i，且元素和为 j。
依次枚举每种元素，背包递推即可得到 g。
答案即为

$$\sum_{i,j} g(i,j)(1+\min(i,\lfloor\frac{j}{10}\rfloor))$$

[redacted]

```
for (v : [1, 11])
    c = cnt[v]
    tot += c, sum += v * c
    for (i : [1, cnt]) for (j : [1, sum])
        g[i][j] += g[i - 1][j - v]
    for (i : [cnt, 1]) for (j : sum, 1)
        g[i][j] -= g[i - (c + 1)][j - v * (c + 1)]
```