



$n$	$\log_{10} n$	$n!$	$C(n, n/2)$	$\text{LCM}(1 \dots n)$
2	0.30102999	2	2	2
3	0.47712125	6	3	6
4	0.60205999	24	6	12
5	0.69897000	120	10	60
6	0.77815125	720	20	60
7	0.84509804	5040	35	420
8	0.90308998	40320	70	840
9	0.95424251	362880	126	2520
10	1	3628800	252	2520
11	1.04139269	39916800	462	27720
12	1.07918125	479001600	924	27720
15	1.17609126	1.31e12	6435	360360
20	1.30103000	2.43e18	184756	232792560
25	1.39794001	1.55e25	5200300	26771144400
30	1.47712125	2.65e32	155117520	1.444e14

$n \leq$	10	100	1e3	1e4	1e5	1e6
$\max\{\omega(n)\}$	2	3	4	5	6	7
$\max\{d(n)\}$	4	12	32	64	128	240
$\pi(n)$	4	25	168	1229	9592	78498
$n \leq$	1e7	1e8	1e9	1e10	1e11	1e12
$\max\{\omega(n)\}$	8	8	9	10	10	11
$\max\{d(n)\}$	448	768	1344	2304	4032	6720
$\pi(n)$	664579	5761455	5.08e7	4.55e8	4.12e9	3.7e10
$n \leq$	1e13	1e14	1e15	1e16	1e17	1e18
$\max\{\omega(n)\}$	12	12	13	13	14	15
$\max\{d(n)\}$	10752	17280	26880	41472	64512	103680
$\pi(n)$	Prime number theorem: $\pi(x) \sim x/\log(x)$					

## 尼姆 (Nim) 游戏

Nim 游戏的规则是这样的：地上有  $n$  堆石子，甲、乙两人交替取石子。每人每次只能从任意一堆石子里面取，至少取 1 枚，不能不取。最后没有子可取的人就输了。假如甲是先手，且已知每堆石子的数量  $a_i$ ，问是否存在先手必胜的策略。

# 结论

- 若初态为**必胜态** ( $a_1 \wedge a_2 \wedge \cdots \wedge a_n \neq 0$ ) , 则**先手必胜**;
- 若初态为**必败态** ( $a_1 \wedge a_2 \wedge \cdots \wedge a_n = 0$ ) , 则**先手必败**。

## 定理证明

### 定理1：必胜态的后继状态至少存在一个必败态

证明：

设  $a_1 \wedge \cdots \wedge a_n = s \neq 0$ , 设  $s$  的二进制位为 1 的最高位是第  $k$  位, 则  $a_1, \cdots, a_n$  中一定有奇数个  $a_i$  的二进制的第  $k$  位为 1。

用  $a_i \wedge s$  去替换  $a_i$ , 则：

$$a_1 \wedge \cdots \wedge (a_i \wedge s) \wedge \cdots \wedge a_n = a_1 \wedge \cdots \wedge a_i \wedge s \wedge \cdots \wedge a_n = s \wedge s = 0$$

这是一种必败态。

同时, 因为  $a_i \wedge s < a_i$ , 是合法的替换。

### 定理2：必败态的后继状态均为必胜态

证明：

设  $a_1 \wedge \cdots \wedge a_n = 0$ , 则相同位上 1 的个数为偶数。此时, 无论减少哪个数, 都会使得异或和  $\neq 0$ 。

必胜态与必败态交替出现, 终态  $(0, 0, \dots, 0)$  是必败态。

终态:  $a_i = 0$ , 异或和为 0, 必败态。

$a_i \neq 0$ , 异或和  $\neq 0$ , 必胜态。

## 台阶型 Nim 游戏

有  $1 \sim n$  级台阶, 第  $i$  级台阶上摆放  $a_i$  个石子, 每次操作可将第  $k$  级台阶上的石子移一些到第  $k - 1$  级台阶上, 移到第 0 级台阶 (地面) 的石子不能再移动。

如果一个人没有石子可以移动, 他就输了, 问先手是否必胜。

# 结论

若奇数台阶上的石子数异或和不为 0，即  $a_1 \oplus a_3 \oplus a_5 \oplus \cdots \neq 0$ ，则先手必胜；否则先手必败。

# 定理

**定理1：必胜态的后继状态至少存在一个必败态**

**定理2：必败态的后继状态均为必胜态**

# 证明

**情况1：** 若初态奇数台阶石子异或和不为 0（必胜态），甲先手：

1. 甲选择某个奇数台阶，将其部分石子移到下一级偶数台阶，使得所有奇数台阶石子数的异或和为 0
2. 如果乙将偶数台阶的石子移到奇数台阶，甲只需将乙移动的同等数量石子继续下移到下一级偶数台阶，保持奇数台阶异或和为 0
3. 如果乙将奇数台阶的石子移到偶数台阶，甲可以调整其他奇数台阶的石子，使得奇数台阶异或和重新为 0
4. 重复上述过程，最终甲将把第 1 级台阶的石子全部移到地面，获得胜利

**情况2：** 若初态奇数台阶石子异或和为 0（必败态），甲先手：

- 无论甲如何操作，都会破坏奇数台阶石子异或和为 0 的状态
- 乙总是可以恢复奇数台阶石子异或和为 0 的状态
- 最终乙将获得胜利

# 有向图游戏

给定一个有向无环图，图中只有一个起点，在起点上放一个棋子，两个玩家轮流沿着有向边推动棋子，每次走一步，不能走的玩家失败。

---

## mex 运算 (minimum exclusion)

$$\text{mex}(S)$$

为不属于集合  $S$  中的最小非负整数,

$$\text{mex}(S) = \min\{x\} \quad (x \in N, x \notin S)$$

例如:

- $\text{mex}(\{0, 1, 2\}) = 3$
- $\text{mex}(\{1, 2\}) = 0$

---

## SG 函数

设状态 (节点)  $x$  有  $k$  个后继状态 (子节点)  $y_1, y_2, \dots, y_k$ ,

$$SG(x) = \text{mex}(\{SG(y_1), SG(y_2), \dots, SG(y_k)\})$$

---

## SG 定理

由  $n$  个有向图游戏组成的组合游戏, 设起点分别为  $s_1, s_2, \dots, s_n$ , 当

$$SG(s_1) \wedge SG(s_2) \wedge \dots \wedge SG(s_n) \neq 0$$

时, 先手必胜; 反之, 先手必败。

# SG 定理证明

## 定理陈述

由  $n$  个有向图游戏组成的组合游戏，设起点分别为  $s_1, s_2, \dots, s_n$ ，当

$$SG(s_1) \wedge SG(s_2) \wedge \dots \wedge SG(s_n) \neq 0$$

时，先手必胜；反之，先手必败。

## 证明

### 基本思路

证明思路类似于 Nim 游戏的证明，通过分析必胜态和必败态的转移关系。

## 定义

- 令  $X = SG(s_1) \oplus SG(s_2) \oplus \dots \oplus SG(s_n)$
- 终态：所有游戏都处于无法操作的状态，此时  $X = 0$

## 证明步骤

### 1. 终态是必败态

当所有游戏都处于无法操作的状态时：

- 所有  $SG(s_i) = 0$  (根据 SG 函数定义)
- 因此  $X = 0 \oplus 0 \oplus \dots \oplus 0 = 0$
- 无法进行任何操作，当前玩家失败

## 2. 若 $X \neq 0$ , 存在操作使 $X = 0$

设  $X = k \neq 0$ , 则存在某个  $SG(s_i)$  使得  $SG(s_i) \oplus k < SG(s_i)$

理由:

- 设  $k$  的最高位为第  $m$  位
- 存在  $SG(s_i)$  的第  $m$  位为 1
- 则  $SG(s_i) \oplus k$  的第  $m$  位变为 0, 且更高位不变
- 因此  $SG(s_i) \oplus k < SG(s_i)$

操作:

- 选择该游戏  $i$
- 将其状态从  $s_i$  移动到某个后继状态  $t$ , 使得  $SG(t) = SG(s_i) \oplus k$
- 这样的  $t$  一定存在, 因为根据 SG 函数定义,  $SG(s_i)$  是其后继状态 SG 值的 mex
- 这意味着 0 到  $SG(s_i) - 1$  的所有值都会出现在后继状态的 SG 值中

结果:

- 操作后, 新的异或和为:

$$X' = SG(s_1) \oplus \cdots \oplus SG(t) \oplus \cdots \oplus SG(s_n)$$

$$= SG(s_1) \oplus \cdots \oplus (SG(s_i) \oplus k) \oplus \cdots \oplus SG(s_n)$$

$$= (SG(s_1) \oplus \cdots \oplus SG(s_i) \oplus \cdots \oplus SG(s_n)) \oplus k$$

$$= k \oplus k = 0$$

## 3. 若 $X = 0$ , 任何操作都会使 $X \neq 0$

假设当前  $X = 0$ , 玩家选择游戏  $i$  并将其状态从  $s_i$  移动到  $t$

分析:

- 如果  $SG(t) = SG(s_i)$ , 则异或和不变, 仍为 0
- 但根据 SG 函数定义,  $SG(s_i)$  是其后继状态 SG 值的 mex
- 这意味着  $SG(s_i)$  不会出现在后继状态的 SG 值中
- 因此  $SG(t) \neq SG(s_i)$

结果:

- 操作后, 新的异或和为:

$$X' = SG(s_1) \oplus \cdots \oplus SG(t) \oplus \cdots \oplus SG(s_n)$$

$$= (SG(s_1) \oplus \cdots \oplus SG(s_i) \oplus \cdots \oplus SG(s_n)) \oplus SG(s_i) \oplus SG(t)$$

$$= 0 \oplus SG(s_i) \oplus SG(t)$$

$$= SG(s_i) \oplus SG(t) \neq 0$$

## 结论

- 从  $X \neq 0$  的状态, 玩家总可以移动到  $X = 0$  的状态
- 从  $X = 0$  的状态, 玩家只能移动到  $X \neq 0$  的状态
- 终态  $X = 0$  是必败态

因此, 当且仅当  $X \neq 0$  时, 先手必胜。

证毕。

## Trick

有时候想想 这个状态是否完备? 状态数是不是可以通过某些手段缩减?

多位dp的时候能把小的项放在前面 会快一点(?)

状压子集枚举  $3^n$

st的子集 for(int i=st;i=(i-1)&st)

st的超集 for(int i=st;i<(1<<n);i=(i+1)|st)

有些dp状态可以奇偶项不同 (

$dp[len][j][k]$  表示长度len满足尾巴是j,k  $j < k$  的序列数

$dp[i][j]$  表示将整数j划分为i个不同的正整数的方案数

$\Leftrightarrow$  等效命题: 存在划分  $l_1, l_2, l_3, l_4 \dots l_k$  满足  $l_1 + l_2 + l_3 + l_4 + \dots + l_k = j, l_1 < l_2 < l_3 < \dots < l_k$

两者是双射

转移 考虑一个j的划分  $\{l_1, l_2, l_3 \dots l_i\}$

case1> 划分中不存在1  $dp[i][j] \leftarrow dp[i][j-i]$  整体-1

case2> 划分中存在1  $dp[i][j] \leftarrow dp[i-1][j-i]$  整体-1,且划分中1的个数-1

$dp[i][k]$  表示合法的数量为i的前缀某排列子集末尾的rk为这个前缀排列中rk=k的方案数

-->解决某类值域上的相邻约束排列计数问题

树上背包经典状态  $dp[u][i]$  表示以u为根的子树中, 背包大小为i的...

复杂度是经典的on方 为啥??

因为背包合并本质笛卡尔积

每一对节点  $(x, y)$  只在lca(x, y) 处被计算一次, 所以复杂度是  $O(n^2)$  的

一类异或=0的dp

$dp[i]$  表示元素异或=0的方案数 可以钦定i=x利用容斥来转移

具体来说 考虑 $dp[i-2]$   $dp[i-1]$ 的含义 然后再kalu

以下为记忆化搜索函数 dfs 的常设定的形参

- **pos:** int 型变量, 表示当前枚举的位置, 一般从高到低
- **limit:** bool 型变量, 表示枚举的第 pos 位是否受到限制
  - 为 true 表示取的数不能大于  $a[pos]$ , 而只有在  $[pos+1, len]$  的位置上填写的数都等于  $a[]$  时该位才为 true
  - 否则表示当前位没有限制, 可以取到  $[0, R-1]$ , 因为 R 进制的数中数位最多能取到的就是 R-1
- **last:** int 型变量, 表示上一位 (第 pos+1 位) 填写的值



- 往往用于约束相邻数位之间的关系的问題
- **lead0**: `bool` 型变量, 表示是否有前导零, 即在 `len → (pos+1)` 这些位置是不是都是前导零
  - 基于常识, 我们往往默认一个数没有前导零, 也就是最高位不能为 0, 即不会写为 `000123`, 而是写为 `123`
  - 只有没有前导零的时候, 才能计算贡献
  - 那么前导零可用时答案有关?
    - 统计 `0` 出现次数
    - 相邻数位的差值
    - 以后问题为起点确定的奇偶位
- **sum**: `int` 型变量, 表示当前 `len → (pos+1)` 的数位和
- **r**: `int` 型变量, 表示整数 `x` 取模某个数 `m` 的余数
  - 该参数一般会固定在: 约束中出现了“能被 `m` 整除”
  - 当然也可以拓展成取数权取模的结果
- **st**: `int` 型变量, 用于状态压缩
  - 对一个集合的数位权上的出现次数的奇偶性有要求时, 其二进制形式就可以表示每个数出现的奇偶性

`dp[st][i]` 表示经过图中某些点构成的集合为`st`,且最后一个点是`i`的某性质

我们在处理某些计数问题时 总是有一下思路

- 1.定1移1 考虑某个前缀对`r`的贡献
- 2.考虑某个点对区间的贡献

$\gcd(k,n)=1 \rightarrow \gcd(n-k,n)=1$   $\sum [\gcd(i,n)=1] \cdot i = \phi(n) \cdot n/2$

<https://ac.nowcoder.com/acm/contest/view-submission?submissionId=79094134&returnHomeType=1&uid=719203876>

原来 $a_j \cdot 2 > a_i$ 的时候 $a_j$ 一定不整除 $a_i$

<https://ac.nowcoder.com/acm/contest/view-submission?submissionId=79086887&returnHomeType=1&uid=719203876>

树上任意一点`v`, 距离其最远的点一定是该树某条直径的两个端点之一。  
考虑反证法即可

联通块

<https://ac.nowcoder.com/acm/contest/view->

[submission?submissionId=79644138&returnHomeType=1&uid=719203876](https://ac.nowcoder.com/acm/contest/view-submission?submissionId=79644138&returnHomeType=1&uid=719203876)

有时候想想菊花图呢

括号序列合法的充要是  $+1 -1$  trick  $\rightarrow$  所有前缀和  $\geq 0$   $\text{sum}=0$

$\rightarrow$  能推导出 这样的串里一定存在() 子串

sqrt 对longlong精度不够 要用sqrtl

某些区间问题 可以利用前后缀处理的思想

[https://ac.nowcoder.com/acm/contest/view-](https://ac.nowcoder.com/acm/contest/view-submission?submissionId=79677851&returnHomeType=1&uid=719203876)

[submission?submissionId=79677851&returnHomeType=1&uid=719203876](https://ac.nowcoder.com/acm/contest/view-submission?submissionId=79677851&returnHomeType=1&uid=719203876)

某些题可以转化成二进制来做

[https://ac.nowcoder.com/acm/contest/view-](https://ac.nowcoder.com/acm/contest/view-submission?submissionId=79675702&returnHomeType=1&uid=719203876)

[submission?submissionId=79675702&returnHomeType=1&uid=719203876](https://ac.nowcoder.com/acm/contest/view-submission?submissionId=79675702&returnHomeType=1&uid=719203876)

eg  $x \leftarrow \text{floor}(x/2) = x > 1$

位运算计数某些经典技巧是拆位, 每位独立贡献

[https://ac.nowcoder.com/acm/contest/view-](https://ac.nowcoder.com/acm/contest/view-submission?submissionId=79091623&returnHomeType=1&uid=719203876)

[submission?submissionId=79091623&returnHomeType=1&uid=719203876](https://ac.nowcoder.com/acm/contest/view-submission?submissionId=79091623&returnHomeType=1&uid=719203876)

xorhash

通常存在于某些2/1性质 奇偶性质的题里面

有些题有奇偶不变量/其他的不变量

压维转化 0-base  $x+my$  1base  $(x-1)+m(y-1)+1$

曼哈顿距离  $\text{dis}(x1,y1,x2,y2)=\text{abs}(x1-x2)+\text{abs}(y1-y2)$

切比雪夫距离  $\text{dis}(x1,y1,x2,y2)=\max(\text{abs}(x1-x2),\text{abs}(y1-y2))$

互转  $(x,y) \rightarrow ((x+y)/2,(x-y)/2)$  原坐标的曼哈顿距离等于新坐标的切比雪夫距离

$\max(a,b)=(a+b+\text{abs}(a-b))/2$

$\min(a,b)=(a+b-\text{abs}(a-b))/2$

dilworth定理

dilworth定理: 一个有向无环图的最小链划分等于其最大反链大小

hall定理

一个二分图是完美匹配的充要条件是，对于任意一个 $X$ 的子集，其邻居数大于等于该子集大小

$\sum a_i \cdot b_i$ 最大的充要条件为 即对于所有  $i, j$ , 若  $a_i < a_j$  则  $b_i < b_j$

排列诸结论

1. 交换两个数的顺序，会使排列的逆序对数 $\pm 1$  奇偶性变换

2. 一个长度为  $n$  的排列，其逆序对数的奇偶性等于 $n - c$   $c$ 是置换环数

置换环：下标 $[l, r]$ 包含了 $[l, r]$ 的所有元素，称为置换环

3. 交换大小为 $x, y$ 的两个块 对逆序对的贡献为 $x \cdot y - 2k$ ,  $k$ 为原来块间的逆序对数