1. 合法方案计数:先想怎么判定,再想怎么 dp ;找原问题的充要条件,可以先找必要条件再考虑构造;
2. dp 优化:二分决策位置,单调性,

值域和某一维(状态数比值域大)互换,

随机交换物品顺序来使期望所要用的状态数较少,

要把多个数组背包起来，可以两两合并

3. 分治(有些维大小可能不变),考虑特殊情况再考虑一般情况能不能转为特殊情况,把某些东西(下标+值$(i, f[i])$)转化为点放到平面上(答案也许与凸包有关)

交换判定的顺序来让问题有序

4. 算概率(实数)时概率很小的情况不管或用其他东西(一次函数，指数函数)去拟合
5. 前缀和优化,可以试试直接用前缀和数组 dp ,前缀和数组的前缀和转移(算恰好等于x的方案也可以变为算 $<= x$ 的方案)
6. 观察有没有单调性,考虑能不能贪心,有一些转移一定不优就不用
7. 观察题目有没有特殊性质,可能某些地方单独特殊处理能优化复杂度
8. 转移要枚举两个参数时可以考虑分开转移
9. 规定操作顺序
10. dp 状态维,记录一些有双重意义的状态会比较优(既是...的值,又是...的值)
11. 考虑倒着 dp ,状态换一些值(特别是期望题)
12. 一些常见转化: $个数^2$ =选两个点的方案数
13. 合并两个状态复杂度高但可以算两个状态的贡献:点分治
14. 二维 dp 中转移可以看成对于函数 dp_i 做平移,加一个函数(打tag),和另一个函数做闵可夫斯基和(插入)之类的操作(然后可能发现有凸性(归纳)之类的或平衡树维护每一段函数)
15. 选一些条件使得可以转移

16.有的状态维可以不要，然后再减去不合法的方案数

17.如果 dp 结果可以表示成多项式，可以插值

18.排除不可能成为最优决策点的点，可以利用单调性和凸性

USACO18DEC Banlance Beam

现在有一条纸带，纸带上有 n 个格子，第 i 个格子上写有数 v ，初始将棋子放进第 k 格，玩一个游戏，规则是：每一轮可以选择结束游戏或者继续游戏 如果选择结束游戏，得分为当前棋子所在格子的数 如果选择继续游戏，则棋子会以相同概率走到左侧或右侧相邻格子 如果棋子走出了纸带，游戏立刻结束，得分为 0 对于$1, 2, \ldots\ldots, k$，求最优策略下，最大期望得分，要求绝对或相对误差不超过 10^{-3}。

$n \leq 10^5$

列出动态规划的式子，可以用左右的平均值更新当前位置。

转化成把$(i, f[i])$放到平面上，每次松弛都是把一个点变为左右两个点的中点，求上凸壳即可。

黄金矿工

假设矿工位于数轴原点，初始时有 n 块金子，且每块金子都位于数轴的正半轴上。第 i 块金子的坐标为x_i，价值为v_i。保证序列 x **严格**递增。获得第 i 块金子所需的时间为 $x_i * v_i$ 。

小 T 想知道在时限内能获得的金子价值之和最大是多少。但关卡有很多，每次通关后，金子以及时限会发生一些变化。共有m次操作，操作有以下两种。

1. 删除第 y 块金子，保证该金子之前没被删除过。
2. 询问在有 k个单位时间时，获得金子的价值之和最大是多少,每块金子每次询问只能获得一次，且询问之间独立

你能帮小 T 在每个关卡都获得理论最高的价值吗?

$1 \le n \le k_{\max} \le 2 \times 10^6$，$1 \le x_i \cdot v_i \le k_{\max}$，$1 \le x_1 < x_2 < \cdots < x_n \le k_{\max}$，$1 \le m \le 5000$，$1 \le k \le k_{\max}$。

以下简记 $K = k_{\max}$。

首先，可以看出很多金子其实是根本不可能用到的。具体来说，以 b 为分界线，那么 $x_i \le b$ 的金子只有至多 b 个。$x_i > b$ 时，对每个 $c \in [1, \frac{K}{b}]$ 考虑 $v_i = c$ 的所有金子。不难发现，我们一定是按照 x_i 从小到大的顺序去选它们，而一块金子至少会花费 bc 的时间，于是可能有用的金子数只有 $\frac{K}{bc}$，对 c 求和可得至多 $O(\frac{K}{b}\log n)$ 块金子。取 $b = O(\sqrt{K \log K})$ 可知有用的金子的数量只有 $O(\sqrt{K \log K})$。

接着考虑类似根号分治的做法，取阈值 B，对 $x_i \le B$ 的金子暴力跑 01 背包。$x_i > B$ 时注意到此时 dp 值的取值范围要比下标的取值范围小，所以我们可以另设 g_i 表示价值为 i 时背包容量最小是多少，此时加入一个物品的复杂度变为了 $O(\frac{K}{B})$。

m 次操作如何处理？首先倒序做使得删除变为加入，而加入直接用上面提到的手段就可以完成了。对于一个询问，我们可以对 $x_i > B$ 部分的背包暴力扫，然后并上 $x_i \le B$ 部分的背包贡献答案。

最终复杂度为 $O\left(BK + \frac{K}{B}(\sqrt{K \log K} + m)\right)$，平衡复杂度可得最终复杂度为 $O\left(K\sqrt{\sqrt{K \log K} + m}\right)$，若看作 $m = O(\sqrt{K \log K})$ 可得 $O\left(K^{5/4} \log^{1/4} K\right)$。

毒假强

对于正整数 k，定义毒假强数 x 满足 $x * (10^k - 1)$ 的十进制表示不包含 9 的正整数，你需要求出第n 个毒假强数。

$k <= 18, n <= 10^{18}$

如果尝试直接求出第 n 个 x 满足 $x \times (10^k - 1)$ 不包含 9 并没有比较好的转化，但是可以考虑求出第 n 个 x 满足 x 的十进制表示不包含 9 并且 x 是 $10^k - 1$ 的倍数，答案即为 $\frac{x}{10^k - 1}$。

考虑逐位确定答案，答案的位数是 $O(k + \log n)$ 的，因此可以将问题转化为 $O(k + \log n)$ 次询问形如 $\overline{x_1 x_2 x_3 \cdots x_k ?? \cdots ?}$ 的数有多少种方案将 $?$ 替换为 $0 \sim 8$ 使得这个数是 $10^k - 1$ 的倍数。

一个数是 $10^k - 1$ 的倍数当且仅当从低往高位，每 k 位划为一段，所有段的和是 $10^k - 1$ 的倍数。

因为段数很少（设为 l），不妨先枚举这个和，即 $[i(10^k - 1)]_{i=0}^{l}$。接下来按照在段中的位置，从低往高数位 dp 即可。

时间复杂度：$O(\text{poly}(\log n + k))$，视实现而定。

```
#include<bits/stdc++.h>
using namespace std;
#define f(i,x,y) for(int i=x;i<=y;i++)
#define df(i,x,y) for(int i=x;i>=y;i--)
#define int __int128
const int N=400;
```

```cpp
long long m,n,B,v[N];//B借位最多可以借的位数
int k,z,f[N][N],g[N][N],pw[N],q[N],h,r,cnt,req,t,x,fg,as,s;
//g[i][j]:i位数(前导0也算)中,数位和为j的有多少个;f[i][j]:所有%m余0~i的位置,填好了,一共向i+1借了j位
int Q(int len)
{
    s=0;
    f(i,0,B-1)
    {
        f(j,0,29)f(k,0,29)f[j][k]=0;f[0][0]=1;
        t=req+i*z;//t:当前枚举的数位和
        f(j,0,m-1)f(k,0,B-1)if(f[j][k])f(l,0,B-1)//j:填所有使得i%m余j的位置i的值(i<len)
        if((x=l*10+t/pw[j]%10-k)>=0)f[j+1][l]+=f[j][k]*g[len/m+(len%m>j)][x];//k,l:上次/这次借位
        s+=f[m][t/pw[m]];
    }
    return s;
}
signed main()
{
    freopen("x.txt","r",stdin);
    scanf("%lld%lld",&m,&n);
    pw[0]=g[0][0]=1;f(i,1,m)pw[i]=pw[i-1]*10;
    f(i,0,21)f(j,0,i*8)f(k,0,8)g[i+1][j+k]+=g[i][j];
    z=pw[m]-1,B=20/m+2;
    df(i,B*m,0)for(;v[i]<=8;v[i]++)//枚举第i位填v[i]合不合法
    if(n<(cnt=Q(i)))break;
    else n-=cnt,req=(req+z-pw[i%m])%z;//req:前面填的数对数位和的贡献
    df(i,B*m,0)//答案为(v[B*m]v[B*m-1]...v[1]v[0])/z
    {
        as=as%z*10+v[i];
        if(as>=z)fg=1;
        if(fg)putchar(as/z+48);
    }
    return 0;
}
```

发怒

有一棵 n 个点的树，树上每个点有一个正整数权值 a_i。定义点集的一个子集 S 是连通的，当且仅当在树上仅保留 S 内的点以及它们之间的边后，这张图是连通的，定义 S 的权值为其包含的所有节点的权值之积。问这棵树上有多少非空的连通的且权值 $\leq m$ 的子集 S，答案对 10^9+7 取模。

首先我们有一个非常显然的 $\Theta(nm)$ 的做法：设 $dp_{u,j}$ 表示有多少个以 u 为根的树上连通块，满足连通块中所有 a_i 的乘积恰好为 j，树上背包以下然后前缀和优化转移即可，一脸过不去的样子。

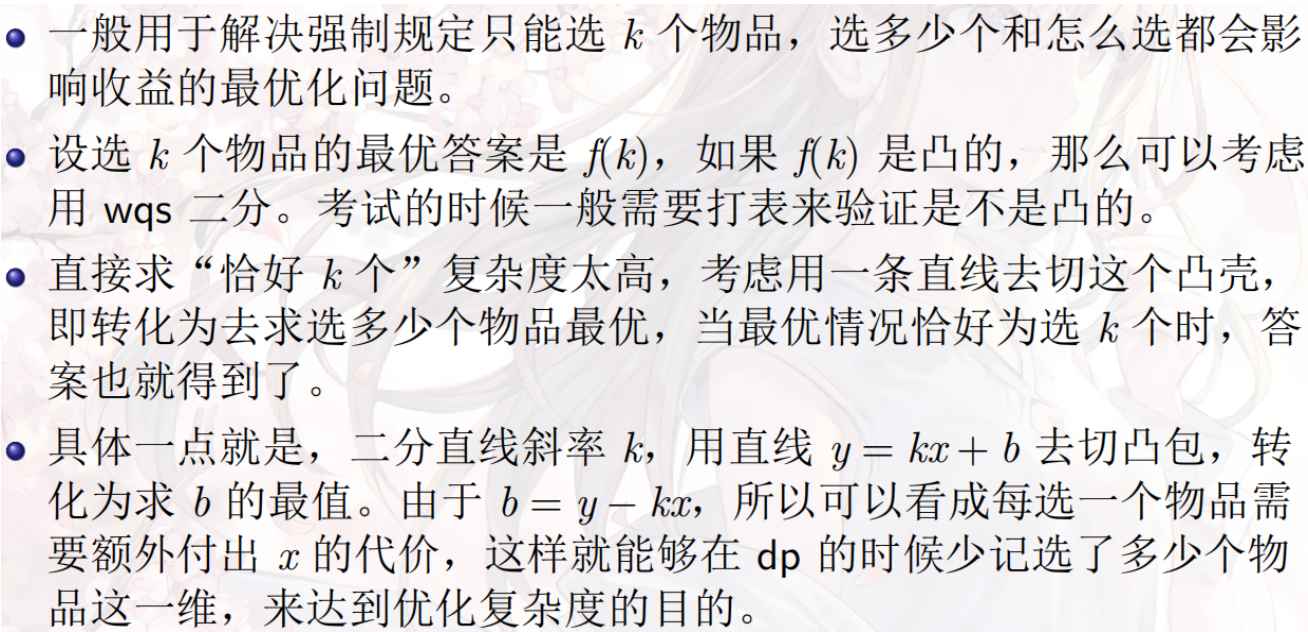
考虑优化。注意到我们在 DP 状态中专门记录一个目前连通块中所有元素的乘积，其实是很浪费的。因为我们只用关心 j 再乘个多少之后就能超过 m，即 $\lfloor\frac{m}{j}\rfloor$，因此我们考虑换一个 DP 状态的定义：$dp_{u,j}$ 表示有多少个以 u 为根的树上连通块，满足 m 除以树上连通块中所有点的乘积等于 j，这样根据整除分块那套理论，复杂度可以降到 $\Theta(\sqrt{m})$，但是这样再跑树上背包就真的可以了吗？非也。因为我们知道 $x=\lfloor\frac{m}{i}\rfloor$，$y=\lfloor\frac{m}{j}\rfloor$，并不能直接推出 $\lfloor\frac{m}{ij}\rfloor$。因此无法合并两个背包，但是我们可以支持加入一个单点，这就导致我们直接树形背包变得相当棘手，因此我们只能另辟蹊径。

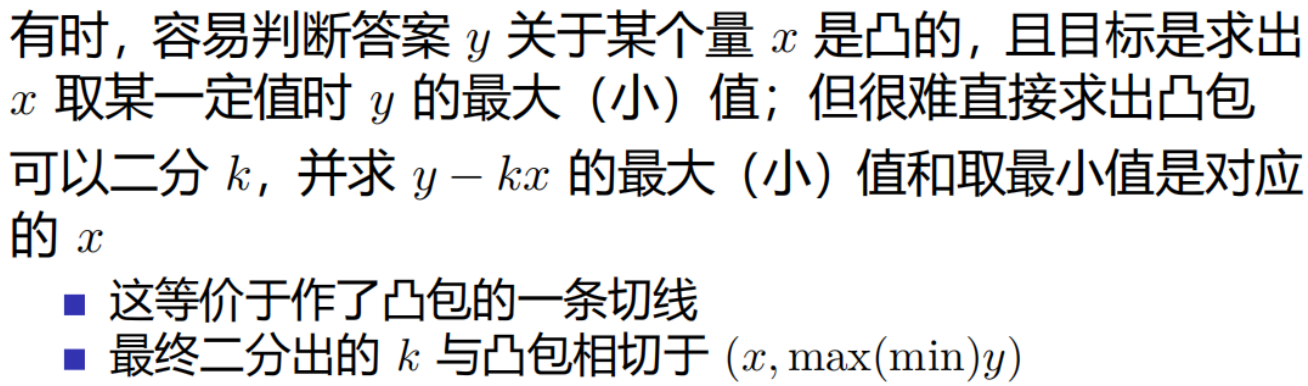
怎么办呢？这里有一个可能有些套路化的科技：考虑**点分治**（树上连通块计数也能点分治？想不到吧）。那么我们分经过当前分治中心的连通块和不经过当前分治中心的连通块处理。对于前者，由于是树上连通块，因此一个点如果被选入树上连通块中，那么它的父亲肯定要被选择，因此我们考虑一个类树形背包的 dp，我们考虑设 $dp_{i,j}$ 表示考虑了 DFS 序 $\le i$ 的节点，它们的乘积 x 满足 $\lfloor\frac{m}{x}\rfloor=j$，有多少个满足条件的连通块（这个 **DFS 序**可能略有点难理解，其实我们可以视作已经考虑过的点的 DP 状态），考虑如何转移，我们 DFS 到一个节点 x 并遍历它的一个儿子 y 时，我们将在当前背包中加入 a_y 后的背包传给它的儿子 y，然后 y 带着这个 DP 值到 y 的子树内荡一圈，上推的时候，再将 y 的 DP 值传给 x，这样我们的 DP 只有添加节点的操作，因此不会出现上面的问题。而对于后者，我们显然可以在后续的分治过程中考虑到，因此继续分治下去即可。

总结：

- 对于状态中带下取整的 DP，可以考虑整除分块优化。
- 对于那种合并不太行，但支持插入的树上连通块类 DP，可以考虑用点分治搞掉合并。
- 对于父亲推给儿子式的 DP，可以考虑按 DFS 序处理。

时间复杂度 $\Theta(n\sqrt{m}\log n)$。





gym102331H Honorable Mention

给定一个长度为 n 的序列 a，有 q 次询问，每次询问给定三个参数 l,r,k，求出对于区间 $[l,r]$，你将其划分为若干个子区间，然后取其中的 k 个，最大化取出来的所有元素的和。即：最大 k 子段和。

$1\le n,q,|a_i|\le 35000$

对于固定的 l, r 答案关于 k 应该是凸的，可以 wqs 二分。

先考虑暴力，我们用线段树维护一个 $2 * 2$ 的矩阵分别表示两端有没有强制选答案关于 k 的凸包,现在求一个答案需要把 $O(log)$ 个凸包闵可夫斯基和 (max ,+卷积) 然后求第 k 项。

但凸包太大,不好求凸包的闵可夫斯基和,但凸包的切线好求,并且凸包的闵可夫斯基和的切线可以由每个凸包分别的切线的和求到。

对于一个询问，二分一个 k ，易求出所有凸包的斜率为k的切线， $2 * 2$ 的矩阵每一项放凸包的切点纵坐标,直接 dp 合并 $O(log)$ 个矩阵。

对于所有询问可以整体 wqs 二分。

每次二分把询问按斜率从大到小排序,每个区间放一个指针指向当前最优解，扫一遍。