矩阵

一个背包问题的变形，考虑状态 $dp[i][j][c][r]$ 表示当选到第 i 行第 j 列的元素、并且在第 i 行已经选取了 c 个元素时模 k 余 r 的最大和是多少。容易得到状态转移方程如下：

- 当不选择 a_{ij} 时，$dp[i][j+1][c][r] = max(dp[i][j][c][r], dp[i][j+1][c][r])$
- 当选择 a_{ij} 时，$dp[i][j+1][c+1][(r+a_{ij})\%k] = max(dp[i][j][c][r], dp[i][j+1][c+1][(r+a_{ij})\%k])$

以上方程只适用于在第 i 行内转移的情况，当从第 i 行第 m 列向第 $i+1$ 行第1列转移时，还要注意将 c 清空，为了方便实现此操作，可以在第 $i+1$ 行第0列设置一个虚拟状态来“继承”第 i 行第 m 列的状态，即 $dp[i+1][0][0][r] = max(dp[i][m][c][r])$ 。最终结果为 $max(0, dp[n][m][c][0])$，这里 $c = 0, 1, \cdots, m/2$。

排列

考虑这样一种构造排列的方法：假设已有一个长度为 i 排列 P_i 满足伴随序列 b，现要在其末尾加上一个数 $j(j = 1, 2, \cdots, i, i+1)$ ，使之成为 P_{i+1} ，我们需要把 P_i 中所有大于等于 j 的数全部+1，使之变成一个排列，并且前 i 个数的大小关系不变，依旧满足伴随序列 b。例如，对于一个 $P_4 = [2, 1, 4, 3]$ 满足 $b_1 = 1, b_2 = 0, b_3 = 1$，现在其末尾添上一个 3，得到 $P_5 = [2, 1, 5, 4, 3]$，前4项依旧满足伴随序列 b ，因此我们只需对新加入的 j 以及 b_i 的值进行讨论即可。

容易发现，按照以上构造序列的方法，只需比较 j 与 P_i 最后一个数的大小即可，不妨设最后一个数是 k ，当 $b_i = 0$ 时，需要 $k < j$，当 $b_i = 1$ 时，需要 $k \geq j$。设 $dp[i][j]$ 表示长度为 i、末尾数字为 j 的满足伴随序列 b 的排列有多少种，可以得到以下转移方程：

- 当 $b_i = 0$ 时，$dp[i+1][j] = \sum_{k=1}^{j-1} dp[i][k]$
- 当 $b_i = 1$ 时，$dp[i+1][j] = \sum_{k=j}^{i} dp[i][k]$

直接计算的话时间复杂度为 $O(n^3)$，这里应使用前缀和优化，这样可以在 $O(n^2)$ 的时间内完成。

树

一道经典的树形DP题目。

- 首先$k = 0$时答案显然为0
- 对于最终所求的连通块，要求不超过一个点的度大于k，其余点的度均小于等于k的树，采用树形dp。
- 首先将原树看成一颗有根树，根节点任意：
  - 我们假设$dp[x][0]$ 表示x为根节点，所有节点儿子数量均小于等于$k-1$的点的子连通图最大边权和。
  - $dp[x][1]$表示x点为根节点，有不超过1个节点儿子数量大于$k-1$，其余节点儿子个数等小于等于$k-1$的子连通图最大边权和。
  - 只要保证点儿子数量均不超过$k-1$，那么显然该点度不会超过k。
- 转移方程：
  - 假设 x 有c个儿子，表示为$s_1, s_2 \ldots . s_c$，我们首先将儿子节点按 $dp[s_i][0]$ 由大到小排序。
  - 可以得到$dp[x][0] = \sum_{i=1}^{k-1} dp[s_i][0] + w_{xs_i}$，其中$w_{xs_i}$表示$x$到$s_i$点的边的边权。
  - 而对于$dp[x][1]$，由于度大于k的不超过一个，那最多只能有1个点儿子数量大于$k-1$
    - 若该点为x点，则有

$$dp[x][1] = \sum_{i=1}^{c} dp[s_i][0] + w_{xs_i}$$

    - 而若该点为x点的子孙节点，则选取x的$k-2$个儿子的$dp[0]$以及1个儿子的$dp[1]$，可以先贪心的将最大的$k-2$个$dp[0]$取下，由于$dp[1]$只需要取一个，那么有两种取法：
        - 第一种为取前$k-1$个点中的某点的$dp[1]$，那么$dp[0]$就少了一个，需要将$dp[s_{k-1}][0]$一并选取，即有
$$dp[x][1]=\sum_{i=1}^{k-1}(dp[s_i][0]+w_{xs_i})+max(dp[s_j][1]-dp[s_j][0]),j\in[1,k-1]$$
        - 第二种为取$k-2$个节点后的某个节点，有
$$dp[x][1]=\sum_{i=1}^{k-2}(dp[s_i][0]+w_{xs_i})+max(dp[s_j][1],j\in[k-1,c])$$
- 最终方案显然是可以选取**某个点的$dp[x][1]$，但是存在一种情况，即选取了某一个节点和其k个子树，而没有将联通块和该节点的父亲节点相连**，这时候答案为该节点的$k-1$个儿子节点s_i的$dp[s_i][0]$以及一个儿子节点的$dp[s_i][1]$组成，做法和求该节点$dp[x][1]$相同，只是多选了一个儿子节点的$dp[0]$而已，在dfs的时候顺便求了即可。
- 时间复杂度为$O(n\log n)$

糖果

求最小的最大值，首先考虑二分答案。对于每次二分的答案 x ，可以用动态规划判断是否能够划分出 k 份使得其中的最大值小于 x， $dp[i]=t$ 表示前 i 个数最多能被划分成满足条件的 t 段，得到转移方程如下：

- $dp[i]=max(dp[j])+1$，其中 j 满足 $\sum_{k=j+1}^{i}a_i\le x$

对于这个方程，直接进行枚举的话可以在 $O(n^2)$ 的时间内完成，但显然我们可以做进一步的优化。令 $sum[i]$ 表示 a 的前缀和，原 j 需要满足的条件就变成了 $sum[i]-sum[j]\le x$ ，考虑建立权值线段树，每次转移时查询一个区间最大值，并更新线段树，那么单次转移的复杂度就变成了 $O(logn)$。总的复杂度为 $O(n\times log(n)\times log(a_i))$