配套练习 by 懵哥

字符串哈希

一、给定字符串 $\mathtt{S}$，$\mathtt{T}$，问 $\mathtt{T}$ 在 $\mathtt{S}$ 中出现多少次。$|\mathtt{S}| \le 1 \times 10^5$，允许 3 次失配。

Source：https://www.luogu.com.cn/problem/P3763

法一：字符串哈希。记 $|\mathtt{S}| = m, |\mathtt{T}| = n$，考虑一个合法的匹配状态，一定是至多四段连续+三个中间间断点——$t[1, l_1], \mathtt{x}, t[l_1 + 2, l_2], \mathtt{y}, t[l_2 + 2, l_3], \mathtt{z}, t[l_3 + 2, n]$，其中 $\mathtt{x}$，$\mathtt{y}$，$\mathtt{z}$ 为失配字符。

记录 $\mathtt{S}$ 每一个前缀的哈希值，首先枚举合法的起始点，对于一个固定的起始点 i，二分答案找到当前 $s[i, m]$ 和 $t[j, n]$ 的 lcp，然后跳过第一个失配字符继续找，直到失配次数超过 3 次或者 $\mathtt{T}$ 匹配完成，其中 j 为 $\mathtt{T}$ 成功匹配的长度+1，初始 $j = 1$，二分的依据是 $\mathtt{S}$ 从 i 开始的长度为 l 的哈希值和 $\mathtt{T}$ 的前缀 l 哈希值相同则 l 增加，反之减少。复杂度 $O(km \log n)$，k 为失配数。此法适用于 k 较小而字符集较大的情况。

法二：字符串 FFT 匹配。通常失配类问题均可用此法，考虑每一个字符 i 对匹配成功的贡献，如果 $\mathtt{S}$，$\mathtt{T}$ 中有出现 i 则该位系数赋为 1 否则为 0，然后将 $\mathtt{T}$ 翻转之后做乘法卷积 FFT，第 j 项系数对应于 $\mathtt{S}$ 从第 j 位开始和 $\mathtt{T}$ 在该字符上匹配成功数。最后只需要将整个字符集 Σ 中对应系数加起来就得到了每一个位置和 $\mathtt{T}$ 的匹配成功数。复杂度 $O(\Sigma m \log n)$，适用于字符集 Σ 较小而失配数较多（或为任意）的情况。

二、给定字符串 $\mathtt{S}$，q 次询问，每次询问 $\mathtt{S}$ 的一个子串的循环节长度。
$n = |\mathtt{S}| \le 5 \times 10^5, q \le 2 \times 10^5$。

Source：https://www.luogu.com.cn/problem/P3538

解法：在 KMP 中我们提到了 border 的概念，并建立了 border 和循环节的关系——若 $n - \mathrm{border} | n$，则 $n - \mathrm{border}$ 为最短循环节长度。border 可以由 KMP next 数组导出，但是这里求的是子串，并且有特殊的要求——能够被整除，因而应该回归 border 的概念——最长真公共前后缀。因而可以使用字符串哈希来判断，其长度必然能被子串长度 m 整除。枚举 m 的每一个因数 x，观察其长度为 x 的前后缀哈希值是否相同，如果相同则 $m \leftarrow \dfrac{m}{x}$，然后继续。复杂度 $O(n + q \log n)$。

三、给定两个 n 维字符串 $\mathtt{S}$，$\mathtt{T}$，问 $\mathtt{T}$ 在 $\mathtt{S}$ 中出现的位置。匹配条件为存在一个位移偏移量 $\vec{p} = (p_1, p_2, \cdots p_n)$，使得 $\mathtt{T}$ 中每一个元素 $\vec{t} = (t_1, t_2, \cdots t_n)$，都有 $\mathtt{S}$ 中对应位置 $\vec{s} = \vec{p} + \vec{t} = (p_1 + t_1 - 1, p_2 + t_2 - 1, \cdots, p_n + t_n - 1)$ 与 $\mathtt{T}$ 相同。保证 $\vec{p}$ 存在，并输出一个字典序最小的 $\vec{p}$。$1 \le n \le 10, |\mathtt{S}| \le 1 \times 10^5$。

Source：http://poj.org/problem?id=2842

解法：考虑在一个 n 维矩阵中，会和多少个元素有相邻关系。对于一个一维的线性表，如果只考虑从前向后的相邻关系，则会产生一个邻接关系——第 i 位排在第 $i + 1$ 位的前面；对于二维矩阵，考虑位置 (i, j)，则会与 $(i + 1, j)$ 与 $(i, j + 1)$ 发生邻接关系。如果每一个点的邻接关系都和原矩阵相同，则可以认为这两个矩阵相同。因而，枚举维数 k，每一轮将每个元素考虑一次，将一个新的邻接关系纳入当前位置的哈希值中，进行递推。总复杂度 $O(nk)$。

四、Cache 存储机制是 LRU 算法：最常使用的放在最前面，如果 cache 满了则踢出最不常使用的内存块。给定一个长度为 n 的操作序列，q 次询问，每次给定 Cache 的大小，并给出一个可能的 Cache 状态，问是否可能出现。$n \leq 5 \times 10^3$，$q \leq 2 \times 10^3$。

Source：https://codeforces.com/gym/102394/problem/L

解法：std 的做法是哈希但是有点卡常。考虑一个无限大的 Cache，暴力的模拟每一次访问后 Cache 的状态，并记录每一个前缀状态对应的哈希值，因为容易发现，限制大小的 Cache 是对应的无限大 Cache 状态的某一个前缀。然后每一次询问均 $O(1)$ 判断即可。注意哈希碰撞问题，推荐使用双哈希。

五、现在给定一个正整数 $n(n \leq 2 \times 10^3)$ 和一个 k，求对于 $\forall x, y \in [1, n]$，x, y 的二进制表示中存在 k 位相同的有序数对 (x, y) 个数。

Source：https://ac.nowcoder.com/acm/contest/11217/C

解法：将每个数的二进制转化为对应的字符串，那么原问题转化为多少个长度大于等于 k 的子串在多个字符串中出现。由于此题 n 较小因而可以考虑暴力哈希——枚举待匹配数 i，将其全部子串用哈希记录下来，对于 $\forall j > i$，考察 j 中子串是否在哈希表中出现过。整体复杂度 $O(n^2 \log n)$。

七、给出一个长度为 n 的串 S，求满足下面条件的最大的 l：S的长度为 l 的前缀和 S 的长度为 l 的后缀是循环相同的。$n \leq 1 \times 10^6$。

Source：https://www.luogu.com.cn/problem/P3546

解法：考虑循环同构的本质——将字符串写为 $ABCBA$ 的形式，则答案为 AB 部分的长度，即是去掉一段公共前后缀后新的 border 长度。记去掉左右外侧各 i 个字符，内部的 border 长度为 f_i。显然 f_i 不具有单调性，但是容易发现，每多一个外侧字符内部的 border 长度至多为上一个+2，即 $f_{i-1} \leq f_i + 2$，因而可以暴力枚举。

八、如果一个字符串可以被拆分为 $AABB$ 的形式，其中 A, B 是任意**非空**字符串，则我们称该字符串的这种拆分是优秀的。现在给出一个长度为 n 的字符串 S，我们需要求出，在它所有子串的所有拆分方式中，优秀拆分的总个数。$|\mathtt{S}| \leq 3 \times 10^4$。

Source：https://www.luogu.com.cn/problem/P1117

解法：考虑 f_i, g_i 分别表示以第 i 个字符结尾的、以第 i 个字符开头的，形如 AA 的子串个数。则答案为 $\sum_{i=1}^{|\mathtt{S}|-1} f_i g_{i+1}$。考虑如何计算 f_i 与 g_i。

显然，如果固定位点，去找当前位点有多少个形如 AA 子串的个数复杂度为 $O(n^2)$。考虑枚举长度，记 l 为 A 的长度，反过来找在哪些点有这样的 AA 串。将全串划块——每隔 l 划一个锚定点（调和级数的复杂度），则一个合法的 AA 串要想出现，必然越过两个相邻的锚定点，并且对应位置全部相同。

考虑两个锚定点向前的最长公共子串（以两锚定点为结尾的）长度 l_1 和向后的 lcp 长度 l_2。如果 $l_1 + l_2 < l$，则考虑从向前的最长公共子串开始的地方向后延申到 lcp 的结尾，也无法凑出一个长度为 l 且相邻的两段相同的串，因而不合法；而当 $l_1 + l_2 \geq l$ 时，此时前面一个锚定点的 lcp 延伸过了后一个锚定点向前的最长公共子串开始的地方。那么所有重叠的地方就都可以构成一个 AA 串中间的分界点。

中间的维护可以使用 ST 表 $O(1)$ 的维护，分界点的答案维护可以使用差分维护（区间加一），因而整体复杂度 $O(n \log n)$。

KMP 与扩展 KMP

一、给定一个长为 n 的串 S，找到 S 中循环节最大循环次数。$n \leq 1 \times 10^6$。

Source：http://poj.org/problem?id=2406

解法：border 的常用结论：$n - \mathrm{border_n}$ 即是循环节长度（不一定完整）。次数等于 $\dfrac{n}{n - \mathrm{border_n}}$。

二、求串 S 本质不同公共前后缀个数。

Source：http://poj.org/problem?id=2752

法一：S 的 border 存放在 next_n 中，border 的 border 即是 S 次长的 border 长度。因而不断跳 border 即可求出本质不同的 border。

法二：使用 Z 函数求出每一个后缀 i 和原串的 lcp 长度，若后缀 i 和原串 S 的 lcp 长度等于 $n - i$ 则证明当前的 lcp 延申到了最后，即这个后缀本身都可以和原串的某一个前缀完全匹配，则当前后缀为一个 border。

三、给定串 S，求每一个前缀对应的 num_i，其中 num_i 表示前缀 i 本质不同的、短于 $\left\lfloor \dfrac{i}{2} \right\rfloor$ 的 公共前后缀个数。$|\mathrm{S}| \leq 1 \times 10^6$。

Source：https://www.luogu.com.cn/problem/P2375

解法：记录一个 f 数组，f_i 表示前缀 i 的公共前后缀的个数（长于 $\left\lfloor \dfrac{i}{2} \right\rfloor$ 的也计入）。那么此数组可以在 KMP 过程中顺带计算出来——$f_i = f_{\mathrm{border}_i} + 1$，而 num_i 即是前缀 i 的短于 $\left\lfloor \dfrac{i}{2} \right\rfloor$ 的公共前后缀 j 的 f_j 值。可以暴力的跳 next 实现。

四、给定一个字符串 S，找到一个 S 的最短前缀，使得它可以覆盖整个 S，允许重复。$|\mathrm{S}| \leq 5 \times 10^5$。

Source：https://www.luogu.com.cn/problem/P3426

解法：考虑 DP。记 f_i 为前缀 i 的最短前缀满足可以完全覆盖前缀 i 的答案，则基础的 $f_i = i$——即原串。考虑能否变短——如果 next_i 足够的长，border 都能互相重叠那么答案可以从 f_{next_i} 转移；此外，如果 border 未覆盖段能够由 border 组成那么也可以，具体表示为 $\exists j \geq i - \mathrm{next}_i, f_j = f_{\mathrm{next}_i}$。使用一个桶记录一下每一个 x 对应有哪些 $f_i = x$ 即可（只需要维护最大的一个即可）。复杂度 $O(n)$。

五、给定一个 $n \times m$ 的字符矩阵，每个位置有一个权值，要求找到一个连续子矩阵是该大矩阵的循环节（即使用该子矩阵通过不重叠平移得到的无限大平面中出现有原矩阵），最小化该子矩阵中所有元素权值的最大值。

Source：https://ac.nowcoder.com/acm/contest/20323/K

类似题：https://codeforces.com/gym/103389/problem/F

解法：一个显然的贪心思想是找到最小的循环节。由于需要在二维上匹配，因而不能只满足于单字符的匹配，而应该考虑整行或整列的匹配。例如：$\texttt{aaabaaa}$，border 为 3，最短循环节为 4，$\texttt{aaaabaa}$，最短循环节为 5。此时不可以取其 lcm 也不是取它们中较大的那一个，而应该综和考虑一个串整体的性质——因为可能顾了横行纵行就满足不了了。

而直接暴力匹配的复杂度过高，因而对整行/整列取哈希值然后做 KMP 匹配。分别对行、列做 KMP，以整行、列元素作为匹配对象进行匹配，此时得到的 border 才是真正的最短循环节大小。

六、（2021威海D）给定一个字符串 $\texttt{S}$，对于每一个字符，如果将其变成未出现的 $\#$，则字符串本质不同周期数目为多少，每次变化独立。

Source：https://codeforces.com/gym/103428/problem/D

解法：显然，一个周期的长度不能越过当前的 $\#$。设 $\#$ 号前有 l 个字符，其后有 r 个字符，则该点的答案为长度不超过 $\min\{l,r\}$ 的公共前后缀的答案。因而可以使用后缀和统计答案。

此题还可以使用扩展 KMP 解决。考虑对 $\#$ 后的全部后缀和原串进行匹配，如果能有匹配成功的，则答案增加一个。同样使用后缀和即可。

七、有一个超长小写字母串 $\texttt{S}$，将其中重复子串压缩成**一个**大写字母，保证压缩过程无环，再给定一个无压缩的子串 $\texttt{T}$，问 $\texttt{T}$ 在 $\texttt{S}$ 中出现过多少次。$|\texttt{S}| \le 1 \times 10^5$。

Source：BZOJ 2061

解法：如果 $\texttt{S}$ 不太长可以直接 KMP 处理。考虑到压缩的次数不会很多——只有 26 个，因而可以考虑每一个大写字母的匹配情况。显然，$\texttt{T}$ 的一次出现只会是在一个完整的字母中（视压缩后的 $\texttt{S}$ 串也为一个大写字母），因而只需要统计每个大写字母内部会包含几个 $\texttt{T}$。注意到 KMP 过程中还有一个指针的变化（对 $\texttt{T}$ 而言的），因而还需要知道每个字母中匹配到当前位置指针的移动位置。因而总复杂度为线性。

八、给定环形字符串 $\texttt{S}$，问能否取出一个长度为 k 的连续子串 $\texttt{T}$，使得环形字符串 $\texttt{T}$ 相邻字符均不相同，需要对每一个 $k \in [1, |\texttt{S}|)$ 进行判断。$|\texttt{S}| \le 1 \times 10^6$。

Source：https://codeforces.com/gym/100162/problem/B

解法：环形字符串通常都是倍长后破环为链。首先如果 $a_i = a_{i+1}$ 则必然不合法，考虑将其断开。因而整个字符串被拆分成为若干个相邻字符各不同的连续子串，并且留下来的部分只能是截取出来的某一段连续子串。

考虑对每一个这样的连续子串枚举其长度 l，看这一段中是否有一个长度为 l 的连续子串符合条件。注意这个长度不具有合法单调性。这些合法的连续子串中不合法的情况即为头尾的字符相同。在这一段中留下的长度记为 k，若 k 不合法则表示每一对相距为 k 的字符均相同——这两个字符在长度为 k 的情况下会放在一起。因而问题进一步转化——每一对相距为 k 的字符均相同，等价于其有一个长度为 $n-k$ 的 border。但是此处不要求是完整循环节，因而直接对子串线性的跑 KMP，得到 next 数组即可；或者使用字符串哈希，线性的枚举公共前后缀的长度依次比较其哈希值即可。复杂度 $O(n)$。

Trie 树与 AC 自动机

一、给定一个字典 $\texttt{T}_1, \texttt{T}_2, \cdots \texttt{T}_n$ 和 m 条文本，对于每一个文本，询问其最长前缀，满足能完整的被拆分成若干个字典中的单词。$n \le 20, m \le 50$，每条文本长度 $\le 2 \times 10^5$。

Source：https://www.luogu.com.cn/problem/P2292

解法：将字典建成 AC 自动机，可以认为是 m 次询问。对于每一次询问，考虑 dp——f_i 表示前缀 i 能否被完全拆分成字典中单词，值为 $1,0$，初值 $f_0=1$。将字符串沿着 AC 自动机匹配，走到终止节点时更新答案。若该字典中串长为 l 则 $f_i|=f_{i-l}$。

二、给定一个字典 $\mathtt{T}_1,\mathtt{T}_2,\cdots,\mathtt{T}_n$，有一个初始串 $\mathtt{S}$。每一轮操作以 p_i 的概率向 $\mathtt{S}$ 添加字符 i，当 $\mathtt{S}$ 包含字典中任意一个单词则停止。现给出长度为 m 的字符串 $\mathtt{R}$，求 $\forall i\in[1,m],\mathtt{S}=\mathtt{R}[1,2,\cdots,i]$ 时的答案。$n\le 100,m\le 1\times 10^4$，字典中单词总长小于 1×10^4。

Source：https://codeforces.com/gym/103119/problem/B

首先以 $\mathtt{T}_1,\mathtt{T}_2,\cdots,\mathtt{T}_n$ 建立 AC 自动机。考虑向后面添加一个字符代表了什么——沿着某一个字母出边走到了另外一个节点。因而原题转化为在一张图上随机游走，走到某一个特定点的期望。 由于有多组询问，考虑优化——由条件概率公式，第 i 个答案可以很自然的转化到第 $i+1$ 个答案，通过除以 $\mathtt{R}_i$ 的出现概率，所以只需要一次操作即可，最后 $O(|\mathtt{R}|)$ 递推一下即可。

由随机游走，将每个点的概率设为未知量，很容易得到一个线性方程组，变元数量就是点的数量。显然，如果节点 x 是一个叶子节点，则 $E(x)=0$。否则，$E(x)=1+\sum\limits_{1\le j\le|\Sigma|}E(g(x,j))p_j$，其中 $g(x,j)$ 表示从 x 号节点沿着字符 j 出边出发到着的点编号。

如果点数不是很多可以直接使用高斯消元求解。可是这里总点数达到了一万，太多了，考虑优化。AC 自动机上的出边要么是 Trie 树上的儿子，要么深度小于当前节点。考虑对 Trie 树进行广搜，将每一个节点的 $E(x)$ 表示成为几个未知量的线性组合，从浅到深的进行。对于每一个节点对应的方程，将其中的未知量尽可能的进行带入消元。如果深度小于当前节点则直接带入；否则将深度大于当前节点的 t 个儿子中的 $t-1$ 设为新的未知量，则最后一个儿子 c 可以由 $E(c)=\dfrac{E(x)-1-\sum_{1\le j\le|\Sigma|,j\ne c}E(j)}{p_c}$ 线性表示。最后所有的叶子节点 $E(l)=0$ 则是最后的方程，这些叶子节点可以表示成上面几个节点的未知量的线性组合表达出来。由于只有 n 个字符串因而 Trie 树上只有 n 个叶子，因而方程数目只有 n 个，可以高斯消元求解。因而整体复杂度 $O(n^3+n^2mk+|\mathtt{R}|)$。

三、给定一棵 n 个点的有根树，树上每条边有个小写字母。q 次询问，每次给定一个子串 $\mathtt{T}$ 和树上节点 u,v，问从 u 开始走一个最短路径到达 v 所构成的串 $\mathtt{S}$ 中 $\mathtt{T}$ 出现了多少次。$n,q\le 1\times 10^5$，$\sum|\mathtt{T}|\le 3\times 10^5$。

Source：BZOJ 4231

解法：$\mathtt{T}$ 在 $\mathtt{S}$ 串上匹配存在两种情况：

1. 经过 lca 而转弯。这部分可以通过暴力找到 lca 两侧长度为 $|\mathtt{T}|$ 的串拼接起来直接暴力跑 KMP 解决。
2. 祖孙方向的匹配。

考虑祖孙方向的匹配，认为从根到叶是正方向，则 u 到 lca 部分为反串匹配，v 到 lca 为正串匹配。定义起始节点为匹配成功时开始位置的节点，则原问题可以拆分成为——根到 u 路径上 $\mathtt{T}$ 串的匹配数，减去根到 u 的 $\mathtt{T}-1$ 级祖先的 $\mathtt{T}$ 串匹配数。那么问题进一步转化到祖孙方向链上串匹配问题。

由于串太多，因而离线询问，将询问的串正反都加入 AC 自动机，然后对树进行遍历。此时 dfs 序会产生一个字符串，并且一次只加入一个字符，或者删除一个字符，符合 AC 自动机匹配原理。因而每到一个节点，记录一下当前到 AC 自动机上哪一个节点，供回溯使用，然后像 AC 自动机一样正常匹配。如果匹配成功则是对应子树的答案都增加，可以使用欧拉序+树状数组维护。

四、给出一串单词，每个有一个权值。顺序不变的情况下，删掉一些，使得相邻两单词，前一个是后一个的子串。同时要求使得剩余单词权值和最大，求最大是多少。

Source：HDU 4115

解法：首先 DP 方程非常好描述——f_i 表示前 i 个串中，所有剩余单词中包含第 i 个的情况中最大权值和是多少，则 $f_u = \max\{f_v + w_v\}$，满足 v 是 u 的某一个子串。那么问题变为快速寻找某串的子串。

子串是某一前缀的后缀，考虑到 AC 自动机上 fail 指针的指向为某一个串的前缀指向比它的某一个后缀，同时该后缀是某一串的前缀。由于我们现在并不关心某一前缀的后缀是什么，而是当前这个串可以作为哪些串的前缀的后缀，因而反向建立 fail 树，树上挂权值 f_v 和 w_v。考虑每个点答案如何更新——寻找其祖先节点 $f_v + w_v$ 值最大的一个，那么可以将其转化为向其欧拉序上子树进行值的更新与最大化。

五、给定两棵有根树 A 和 B，每条树边上有个小写字符。定义 $A_{s\to t}$ 表示从 s 开始沿着最短路线走到 t，将途径的边上的字符按顺序写下来得到的串。一开始 A 和 B 都只有根节点 1，有 n 次操作，$n \le 1 \times 10^5$，每次操作会选择其中一棵树，在某个点下面添加一个叶子。请在每次操作之后统计满足条件的三元组 (i, j, k) 个数：

1. $1 \le i \le |A|, 1 \le j, k \le |B|$，且 j 是 k 的祖先。
2. $A_{1\to i} = B_{k\to j}$。

Source：BZOJ 3467

解法：先将操作离线下来，建出完整的树 A, B。由于 B 树中一个节点存在大量的串——具有后缀关系，考虑对其 2^k 级祖先和中间子串的哈希值记录下来。考虑对 B 树中节点按后缀排序的方式进行排序——比较各级哈希值，相同则继续，否则终止。此时 A 树中每个节点对应于 B 树中节点按照后缀排序的方式下排列的连续一段，且父节点的区间一定包含子区间。

此时考虑加入 A 中的一个节点，那么就是在 B 树节点上建立线段树求其对应区间和；如果是 B 树上加一个节点，则是 A 树上一条祖孙链加。将 A 树欧拉序列化，因而每一个 B 树节点都对应于 A 树欧拉序列的一个区间。对 A 树欧拉序列也搭建一个线段树即可。

~~Source：HDU 5081~~

后缀数组 SA、后缀自动机 SAM

一、求一个字符串本质不同子串个数

Source：https://www.luogu.com.cn/problem/SP694、https://www.luogu.com.cn/problem/P2408

法一：后缀数组。首先如果不考虑重复问题，则总的子串个数应该是 $\dfrac{n(n-1)}{2}$ 的。经过后缀排序后，存在公共前缀的后缀就会被排序到一起，height 数组表示了相邻的两个后缀其 lcp 长度，也表示了这两个后缀它们的前缀产生了这么多个本质相同的子串（由于起始点不同因而必然位置不同，但是它们的内容相同）。因而答案为 $\dfrac{n(n-1)}{2} - \sum\limits_{i=2}^{n} \mathrm{height}_i$。

法二：后缀自动机。后缀自动机上一个节点表示了一类子串——它们具有共同的 endpos 集合。那么出现位置不同但是本质相同的字符串就只由这一个节点表示了。一个节点包含的子串个数由 $cnt_u = \mathrm{maxlen}_u - \mathrm{maxlen}_{\mathrm{link}_u}$ 给出，因而本质不同子串就是每个节点的 cnt 之和。

二、求一个字符串第 k 大子串。

Source：https://www.luogu.com.cn/problem/P3975

解法：求解第 k 大通常采用二分答案。

法一：后缀数组。后缀排序后所有的子串按大小顺序排列，因而第 k 大只需要二分处于哪一个后缀即可。预处理后缀排序后前 i 个后缀具有的本质不同子串个数和即可。

法二：后缀自动机。由于后缀自动机上字母出边构成了一个 DAG，因而可以考虑 DP——$\mathrm{cnt}_u = \sum_{DWAG} \mathrm{cnt}_v$，其中 u 到 v 有一个字母出边。那么沿着这个 $DWAG$ 出发即可，类似于二叉树求解第 k 大数的方法——如果当前字母出边对应的剩余 DAG 内字符串个数多于 k，则当前字符就是当前这条字母出边；反之 k 减去这一子树的大小。

三、给定 n 个串，每个串的“独特系数”为只在这个串中出现的、不同子串的数目，求每个串的独特系数。$n \leq 1 \times 10^5$。

Source：https://www.luogu.com.cn/problem/P4081

解法：广义后缀自动机，可以使用简单版本的广义 SAM 构建方法。在添加一个字符串的时候，在树上的节点对应添加当前节点被几个字符串“染色”过，最后一次染色的是哪一个字符串。最后遍历一遍 SAM 上的每一个节点，统计仅被一个字符串染过色的节点，统计该节点对应的子串个数即可。

四、求两个字符串 $\mathtt{S}$，$\mathtt{T}$ 的最长公共子串

Source：http://poj.org/problem?id=2774

法一：后缀数组。构建一个新串 $\mathtt{S\$T}$，对其进行后缀数组排序。那么当前公共子串长度一定是某个长度大于 $|\mathtt{T}|+1$ 的后缀的 height 值——仅包含 $\mathtt{T}$ 的短者在后缀数组排序一定居前。因而 $O(n)$ 的遍历一遍 height 数组即可。

法二：后缀自动机。对串 $\mathtt{S}$ 建立后缀自动机。考虑类似 AC 自动机的匹配规则：如果能够匹配则沿着边匹配下去，否则就找到当前匹配段的最长后缀，使得它的后面可以接受失配字符。注意到 link 树的性质：同一个节点存储了一系列子串，这些子串的出边都是一样的；最短的、出边不同的可能后缀只有可能在 link 树上的 father。因而暴力跳 link，找到能匹配的出边为止。如果走到 dummy 节点还是失配则调过（也可以认为 dummy 节点具有全部的字符出边，保证一定可以接受）。

五、给定若干个串，求至少在 k 个串中出现的子串的最大长度。

Source：http://poj.org/problem?id=3294

法一：后缀数组。把所有的串中间插入 $\$$ 连接起来进行后缀数组排序，在 height 数组上找到一段长度为 k 的连续子序列的最小值的最大值。

法二：广义后缀自动机。考虑在构建自动机的时候，经过一个节点则把该字符串的 id 给标记到树上。最后遍历一下树上节点，对于节点上颜色个数超过 k 的节点，将其 maxlen 拿去更新答案即可。

六、给定一个长为 n 的串 $\mathtt{S}$，求重复次数最多的一个连续子串，即该串可以分解为若干个相同子串的拼接，如有多个选字典序最小的输出。

Source：http://poj.org/problem?id=3693

解法：（此题和第二十一题有相似之处）考虑枚举子串的循环节长度 l，对于一个给定的 l，将字符串按 l 的长度划块产生锚定点（这部分是调和级数的复杂度），那么如果存在某一个长度为 l 的子串重复多次，则一定会至少覆盖相邻的两个锚定点。因而枚举每个相邻的锚定点，考察其向前匹配的最长长度（使用哈希）和向后匹配的长度（使用 height 数组），计算其和 k，则其循环次数为 $\frac{k}{l}+1$，然后统计答案即可。最后由于要输出具体的字符串，因而需要使用 SA 数组来筛选答案。总复杂度 $O(n\log^2 n)$。

（注：此题也可不用哈希——因为向前的部分可以只通过向前探测一个循环节长度实现，因为更长的是由前一组相邻锚定点所覆盖）

七、给定字符串 S，求 $\sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{lcp}(s_i,s_j)$。$|\mathtt{S}|\le 1\times 10^6$。

Source：https://atcoder.jp/contests/abc213/tasks/abc213_f

类似题：https://www.luogu.com.cn/problem/P4248

解法：后缀数组排序后，问题等价转化为 $\sum_{i=1}^{n}\sum_{j=1}^{n}\min_{i\le k\le j}\mathrm{height}_k$。使用单调栈正反各操作一次即可。在单向遍历的过程中，使用一个单调栈维护单调递增的序列，以及对应的管辖长度，并单独使用一个变量统计总答案。该单调栈方法与 ARC 115E 相同。

八、给定两个串 S，T，求它们公共子串个数。出现位置不同即认为不同。

Source：https://www.luogu.com.cn/problem/P3181

解法：后缀自动机。考虑对 S 建立后缀自动机，然后将 T 串进行匹配。在匹配的过程中，同步的维护当前 S 和 T 的匹配长度 l。考虑匹配到 T 串的第 i 个字符时，应该加入前缀 i 的全部匹配后缀数目。该部分由当前节点 j 和其祖先节点的贡献和构成。当前节点的答案就是 $|\mathrm{cnt}(u)|(l-\mathrm{len}_{f_u})$，而祖先节点贡献可以预先算出——$|\mathrm{cnt}(u)|(\mathrm{len}_u-\mathrm{len}_{f_u})$，遍历时在 SAM 上打上标记，最后一次遍历完成。

九、给定两个串 S，T，求它们长度不小于 l 的公共子串的个数（子串可以相同，只要位置不同即可）。

Source：http://poj.org/problem?id=3415

法一：后缀数组。后缀数组排序后，公共子串个数和长度均可以用两个位置 i,j 之间 $\min_{i\le k\le j}\mathrm{height}_k$ 表达。因而问题转化成为了给定一个数列，问有多少对 (i,j) 满足 $\min_{i\le k\le j}a_k\ge l$，其贡献为 $\min_{i\le k\le j}a_k-l$。此处方法同上。

法二：后缀自动机。用字符串 S 构造 SAM，在 SAM 上匹配第二个字符串 T。设当前匹配长度为 len，且位于状态 p。如果有对应的字母转移出边则转移过去，长度增加；否则跳到 link_u，匹配长度变成对应的节点的长度。此处类似于 AC 自动机——如果能匹配则继续匹配，否则找到已经匹配段的最长后缀，该后缀是“字典”中某一个单词的前缀。此处的字典就是 S 的全部子串，考虑在 SAM 上一类子串具有相同的出现次数，也意味着它们有相同的字母出边。因而最长的可能的后缀至少是 link_u 中的最长子串。

考虑每匹配一个字符对应答案增加多少。如果当前匹配段长度小于 k 则继续，此处不会增加贡献；否则，当前节点增加的贡献就是当前节点的 endpos 集合大小+其祖先节点的匹配贡献。这是由于多匹配了一个字符，因而匹配的右端点发生变化——前面所有的匹配子串长度全部要增加 1 变成新的匹配段。注意，如果在 S 匹配过程中再次访问到同一个节点，答案依旧要是要计算——在 T 中匹配位置相同时 S 中匹配位置不同也要算作不同的匹配。

显然，对于其祖先节点的贡献不能直接暴力计算——考虑挂一个祖先标记，表示从这里开始到根节点，只要匹配长度超过 k 就要统计答案。最后倒序遍历整棵树把标记逐一上放并计算即可。

十、给定由若干个音符（1 到 88 之间的数字）组成的音乐片段，定义该片段的一个主题是满足以下条件的一串连续音符：

1. 至少由 5 个音符组成
2. 至少在整个片段中出现 2 次（包括转调，也就是每个音符都偏移相同的音高）
3. 至少有 2 次出现不重叠

求最长主题的长度。

Source：http://poj.org/problem?id=1743

解法：由于可以转调，因而不是直接的完全相同。但是偏移量相同，则表示其差分数组相同。因而首先对原式做一个差分，变成完全相同。现在要求至少由 4 个字符构成，并且出现不重叠。但是考虑到差分数组会导致新数列不重叠但是原数列重叠（例如 $1,2,3$ 差分数组变成 $1,1,-3$，$1,1$ 不重叠但是 $1,2$ 和 $2,3$ 重叠），所以要求出现的两个子串相差一个数（即串长减一）。因而一个简易方法是二分这个子串长度，然后扫一遍 height 数组，看有无一段长度为 k 满足值均大于等于 k 的子数组即可。

十一、给定两个字符串 $\mathtt{S}$，$\mathtt{T}$，计算：

1. $\mathtt{S}$ 的一个最短的子串，它不是 $\mathtt{T}$ 的子串。
2. $\mathtt{S}$ 的一个最短的子串，它不是 $\mathtt{T}$ 的子序列。
3. $\mathtt{S}$ 的一个最短的子序列，它不是 $\mathtt{T}$ 的子串。
4. $\mathtt{S}$ 的一个最短的子序列，它不是 $\mathtt{T}$ 的子序列。

Source：https://www.luogu.com.cn/problem/P4112

解法：考虑 SAM 可以接受全部的连续子串，序列自动机可以接受全部的子序列，因而把 $\mathtt{S}$，$\mathtt{T}$ 的 SAM 和序列自动机都搭建出来，**同步**进行广搜，即枚举同一个转移字符，然后两个指针在两棵搜索树上一起转移。如果在 $\mathtt{S}$ 的自动机上可以识别，而 $\mathtt{T}$ 上不可识别，则找到了答案。

十二、给定一个长度为 n 的字符串 $\mathtt{S}$，要求通过合理的设定一个概率分布函数 p，最大化 $\min\limits_{1\le j\le n}\{\sum\limits_{k=1}^{n}p_k\mathrm{lcp}(s_k,s_j)\}$，其中 s_i 表示 $\mathtt{S}$ 的后缀 i。

Source：https://codeforces.com/gym/102056/problem/J

解法：一个最大化一个最小化，可以看做先手的收益是 $\sum\limits_{k=1}^{n}p_k\mathrm{lcp}(s_k,s_j)$，后手的收益是 $\sum\limits_{k=1}^{n}p_k\mathrm{lcp}(s_j,s_k)$。所以这就是一个零和博弈，答案会在纳什均衡点处取到。即第一个人选择一种混合策略，使得自己在最坏情况下收益最大，也就是对面不管怎么决策，收益都一样。

法一：后缀自动机。找出后缀树上的后缀节点，然后令它们的收益相同，从而解出各个位置的 p 和收益。例如，考虑三个子树 f_1,f_2,f_3，对其分配权值 p_1,p_2,p_3，则有 $p_1f_1=p_2f_2=p_3f_3,p_1+p_2+p_3=1$。暂时忽略 p_3，记 p_1 和 p_2 一起合并占据 p 的概率，则有 $p_1f_1=(p-p_1)f_2$，解出 p_1 关于 p 的线性表达式，再将 f_1 和 f_2 合并成 f，去解第三个等式，即可得到 p，也就算出了 p_1,p_2,p_3。复杂度 $O(n)$。

法二：后缀数组。我们知道排名在 $[l,r]$ 中的后缀的 lcp 是 $\min\limits_{l+1\le i\le r}\{\text{height}_i\}$，设 height 数组最小值的位置是 p，如果当前的两个后缀 j,k 在 p 的两边，那么此时的 lcp 就是 height_p；否则 j,k 在同侧，可以考虑递归到两边去做。能够想到的是，**局部最优决策下的概率比等于全局最优决策下的概率比。**也就是说令左边子区间在最优决策下，各位置的概率比为 $p_1:p_2:p_3\cdots p_k$，那么在于右区间合并，也就是在全局中，左区间各位置的概率比仍为$p_1:p_2:p_3:\cdots p_k$。这个结论不难证明，当 j,k 同在左区间中的时候，答案是和右区间无关的（否则 j,k 有一个在右区间的话，答案和 height_p有关，等会再讨论）。
所以我们可以分治去做，最后合并左右两区间，也就是 j,k 在 p 的异侧的时候。设左区间的答案函数为 l，整体分到的概率是 x，右区间的答案函数是 r，整体分到的概率就是 $1-x$。假如后手选择 j 在左区间，那么 k 在右区间时先手的收益是$l(x)+(1-x)\text{height}_p$；$j$ 在右区间，收益就是 $r(1-x)+\text{height}_p x$。
先手会令这两个式子相等，所以我们能解出 x，然后代到左式或右式就能得到当前区间的答案了。
分治复杂度 $O(n)$，建 SA 的复杂度 $O(n\log n)$。

~~Source：https://www.luogu.com.cn/problem/P4770~~

十四、给定一个长度为 n 的字符串 S 和长度为 n 的权值表 $\{a_n\}$，$|a_n|\le 1\times 10^9$。定义两个位点 p,q 是 r 相似的，即从 p,q 开始向后延申 r 个字符均相同。对于每个二元组 (p,q) 其权值为 $a_p\times a_q$。统计 $\forall i\in[0,n-1]$，i 相似的二元组 (p,q) 的对数，以及 i 相似的二元组的最大权值。

Source：https://www.luogu.com.cn/problem/P2178

解法：由于一个 r 相似的 (p,q) 对一定 $r-1$ 相似，那么显然，如果按照 r 单调递减的顺序进行统计，那么就是一个只合并不拆分的问题，考虑使用并查集。考虑对 SA 上的 height 值按大小排序，一个 height 值表征了两个位点的 r 相似关系，因而对两个位点进行合并（因为显然，如果再找到一个点能和 p 产生 k 相似，$k<r$，则也一定可以跟 q 产生 k 相似），答案增加两个子集的大小乘积。同时维护一个集合的最大、最小值（防止负负得正），同步的维护。

十五、给定一个母串 S 和 n 个询问串，对于每个询问串，如果能够将该询问串拆分成若干个 S 中连续子串（母串中连续子串可复用）的拼接，求拆分的子串最少数目。

Source：https://codeforces.com/gym/102428/problem/G

解法：如果没注意到贪心，很容易写出这样的一个 DP 方程——记 f_i 表示前 i 个字符能够分拆出的最少子串数，则 $f_i=\min\limits_{1\le j<i} f_j+1$，$j+1$ 到 i 为原串的一个子串。但是显然，如果 S 中一个子串能尽量延申，那这样一定最优——这样保证了后面尽可能的少。因而只需要建立一个 SAM，然后对于每个询问串在 SAM 上跑一遍即可。

十六、给定一个空字符串，每次向其末尾插入一个字符，问每次插入完成后本质不同字符串个数。

Source：https://www.luogu.com.cn/problem/P4070

解法：注意到如果每次向后添加一个字符，那么生成的后缀数量是 $O(n)$ 的，且后缀数组排序的结果也都会变得混乱；但是如果反置这个串，显然答案是不会发生变化的，但是这样相当于每次向前添加一个字符，后缀数增加了 1，height 数组也是 $O(1)$ 的变化。因而维护一个 int 类型的 set，以下述规则排序：对于 i,j，首先求出后缀 i,j 的 lcp 长度 l（可用字符串哈希），然后比较 $s[i+l],s[j+l]$，定义 $\forall i>n,s[i]=0$。然后插入到对应的位置后，该后缀创造的本质不同子串个数为 $n-i+1-\max\{\text{lcp}(rk[i],rk[i-1]),\text{lcp}(rk[i],rk[i+1])\}$。可以用 set 维护。

十七、给定一个字符串 S，和一个操作序列 op，每次可以向 S 的末尾添加或者删除一个字符，问每次操作后 S 本质不同子串有多少个，操作序列需要叠加。

Source：https://codeforces.com/gym/103185/problem/M

解法：法一，后缀数组。此题是在上一题的基础之上增加了删除功能，但是基本思路一样。依旧是字符串翻转，插入的时候和上题相同；删除时只需要把这个串造成的影响消去，**同时补上** $\mathrm{lcp}(rk[i+1], rk[i-1])$ 即可。

法二，后缀自动机。对于插入操作和正常的后缀自动机相同，只需要在新添加的节点将之前的答案加上对应的 cnt 即可。对于删除操作，即是执行其逆过程——首先将对应的字符出边删去，然后将分裂的节点重新合并，同时删除其对应的 cnt。此题建议用此方法，常数较小。

十八、（2021 桂林 J）将一个字符串 S 的全部本质不同的连续子串按照长度为第一关键字、字典序为第二关键字进行排序，t 组询问，询问在这种排序规则下第 k 个字符串最左侧出现位置。

Source：https://codeforces.com/gym/103409/problem/J

解法：首先后缀数组排序，统计每个长度下有多少个本质不同的字符串。将询问离线按 k 大小排序，那么每一个长度的字符串只会首先二分一次可以得知对应的长度。使用一个线段树维护每一个后缀在当前长度下是否应该贡献一个长度为 k 的字符串——由 height 数组决定。由于长度单调递增，因而在线段树上的更新只有 $O(2n)$ 次——对于每一个位点，只有一次插入（当前长度恰好为当前节点的 height 值）和一次删除（当前后缀长度小于待求长度）。统计使用线段树二分即可。

~~Source：https://www.luogu.com.cn/problem/P4384~~

二十、给定一个长度为 n 的字符串 S，请将 S 分成不超过 k 段，使得每一段的所有子串中，字典序最大的子串最小。$n, k \leq 1 \times 10^5$。

Source：BZOJ 4310

解法：字典序最大的子串不好直接用串来刻画，那就用它在全串中所有子串的排名来刻画。因而首先 SA，然后二分出字典序最大的子串的排名。从后往前划块，因而每次都会添加一些新的子串。考虑新加入的子串中其中字典序最大的那一个——即全串。如果它符合排名要求，则继续；否则则需要重新划块。显然这样是不会影响之前的子串结果的。复杂度 $O(n \log n)$。

Manacher 与回文自动机 PAM

一、给定字符串 S，定义一个子串的存在值为这个子串在 S 中出现的次数乘以这个子串的长度，求所有回文子串中的最大存在值。

Source：https://www.luogu.com.cn/problem/P3649

解法：PAM 板子题，只需要在插入过程中顺带统计出现次数即可，PAM 的本质和 SAM 相同。最后遍历一遍 PAM 即可得到。

二、给定一个由 a 和 b 构成的字符串，求不连续回文子序列的个数

Source：BZOJ 3160

解法：首先为了将奇偶回文串一并处理，中间插入 $\#$ 来消除影响。考虑维护一个数组 f_i 表示以第 i 个字符为中心的最长不连续回文子序列的长度，则答案为 $\sum_{1\le i\le 2n+1} 2^{f[i]-1}$——因为要排除插入的特殊字符对答案的影响。

容易注意到——如果某两个位置为 i,j 的相同字符要产生贡献，则产生贡献的位置一定在 $\frac{i+j}{2}$ 处——只关于这个地方对称。则这个问题就被转变为卷积问题了，一个 FFT 即可。

三、输入长度为 n 的串 S，求 S 的最长双回文子串 T，即可将 T 分为两个非空部分 X,Y 且 X 和 Y 都是回文串。

Source：https://www.luogu.com.cn/problem/P4555

解法：考虑 Manacher 求出以每个点为中心扩展的最长长度 p_i，则可以用差分数组和线段树一起维护出 l_i 与 r_i，分别表示以 i 为回文串左端点和回文串右端点的最长回文串长度，然后答案即为 $\max_{1\le i<n}\{l_i+r_{i+1}\}$。

四、给定一个长度为 n 的字符串，问长度为奇数的前 k 长回文子串长度总乘积。
$n\le 1\times 10^5, k\le 1\times 10^{12}$

Source：https://www.luogu.com.cn/problem/P1659

解法：此题由于只求奇回文串，因而免去了插入特殊字符的过程。直接进行 Manacher，求出对应的 f 数组，然后在统计数目的桶中区间增加 1，最后枚举一遍桶即可。

Lyndon 分解

一、（2021沈阳 M）给定一个字符串 S，输出其每一个前缀中字典序最大的子串位置，若多次出现取最左侧出现位置。

解法：考虑 Lyndon 分解（反向）的结果——当前串被分解为若干个不严格单调递增的子串，且越往后越子串字典序越大。因而模拟 Lyndon 分解的过程——如果当前字符和循环部分相同，则延长循环部分；如果更小，则合并全部的类 Lyndon 串部分；否则从当前循环节开始，重新计数。

二、对于超长字符串，可以将其压缩成 $(S_1)k_1(S_2)k_2\cdots(S_n)k_n$ 的形式，其中 $(S)_k$ 表示 S 重复 k 次。不允许嵌套压缩。给定两个解压后等长的压缩串 S 和 T，请判断 S 和 T 是否循环同构。
压缩串长 ≤ 11000。解压后串长 $\le 2\times 10^8$。

Source：HDU 5509

解法：循环同构本质即最小表示法相同，因而对两个串进行 Lyndon 分解找最小表示法。此题的循环相当于在 Lyndon 分解过程当中预处理好了循环节——因而只需要拆出第一次循环的部分，和后面循环 $k-1$ 次的部分。第一次因为可能导致循环节更新因而不合并处理，后面循环 $k-1$ 次的部分在 Lyndon 分解中就是暴力循环。最后统计分解结果的时候也是利用这一循环进行存储。

序列问题相关

一、给定字符串 S，q 次询问区间 $[l,r]$，问至少需要几个字符才能不是区间 $[l,r]$ 的子序列。
$q\le 2\times 10^5, |\mathrm{S}|\le 2\times 10^5$。

Source：https://codeforces.com/gym/103389/problem/B

解法：考虑构建一个序列自动机，每一个节点指向下一个出现字符 ch 的位置。由于字符数最少，因而贪心的选取出边指向位置最远的地方。因而问题转化成为：询问区间 $[l,r]$，从 l 出发至少要跳转几步才能大于 r。使用倍增即可，整体复杂度 $O(nm+q\log n)$。