提高算法班

离散化、Z Algorithm、Trie

Mas

离散化

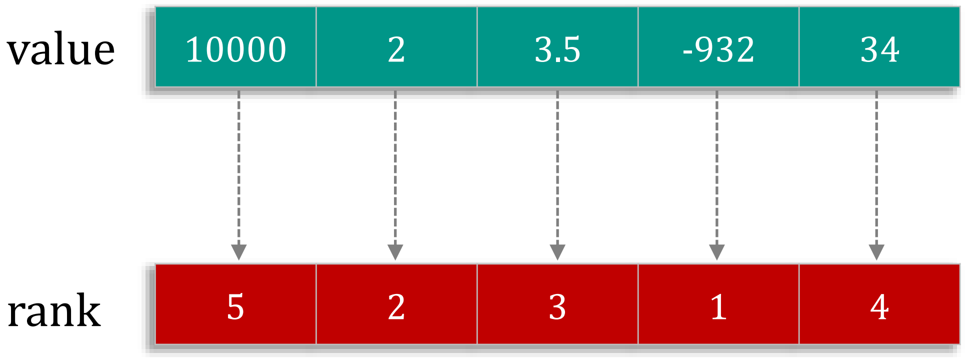
离散化本质上是一种哈希

其保证数据在哈希后仍保持原来的全/偏序关系

当有些数据因为本身很大或者类型不支持，自身无法作为数组下标使用

而影响最终结果的只有元素间的 **相对大小关系** 时

将原来的数据按照从小到大编号来处理问题，即离散化



用来离散化的可是大整数、浮点数、字符串等

#2718、二维离散化

题目描述

给你 n 个矩形

每个矩形的左上角是两个实数 x_1, y_1 ,右下角是两个实数 x_2, y_2

请你输出则 n 个矩形的面积并

> *注意如果一片区域被多个矩形包含,则在计算总面积时只计算一次*

输入格式

第一行包含整数 n ,表示总的矩形数量

接下来 n 行,每行包含四个数字, x_1, y_1, x_2, y_2

其中 (x_1, y_1) 和 (x_2, y_2) 分别是矩形的左上角位置和右下角

> *坐标轴 x 轴从上向下延伸, y 轴从左向右延伸*

输出格式

面积,输出保留两位小数

思路1

扫描线 + 线段树维护

详见 线段树课件

时间复杂度 $O(n\log n)$

数据范围

对于全部的数据范围 $1 \le n \le 100, 0 \le x_1 < x_2 \le 10^7, 0 \le y_1 < y_2 \le 10^7$

#2718、二维离散化

思路2

将矩形四个端点进行离散化

离散化后的点重新建立矩形

矩形格点数不超过 200×200

在二维数组中进行标记，统计标记后的点

离散后的整个二维平面上的单元格不再是 1×1 矩形

单元格 (i, j) 的面积为

$$(X_i - X_{i-1}) \times (Y_j - X_{j-1})$$

其中 X, Y 为横纵坐标去重排序后的数组

时间复杂度 $O(n^3 + n\log n)$

```cpp
for (int i = 1; i <= n; i++)
{
  scanf("%lf%lf%lf%lf", &a[i].x1, &a[i].y1, &a[i].x2, &a[i].y2);
  xPos.push_back(a[i].x1), xPos.push_back(a[i].x2);
  yPos.push_back(a[i].y1), yPos.push_back(a[i].y2);
}
disc();
for (int i = 1; i <= n; i++)
{
  int tx1 = findX(a[i].x1), tx2 = findX(a[i].x2);
  int ty1 = findY(a[i].y1), ty2 = findY(a[i].y2);
  for (int x = tx1 + 1; x <= tx2; x++)
    for (int y = ty1 + 1; y <= ty2; y++)
    {
      if (!vis[x][y])
        ans += (xPos[x] - xPos[x - 1]) * (yPos[y] - yPos[y - 1]);
      vis[x][y] = true;
    }
}
printf("%.2f", ans);
```

#2720、还是二维离散化

题目描述

有一个 $R \times C$ 个格子组成的迷宫

迷宫中有 T 面墙,每面墙的坐标为 x_i, y_i

每个格子与上下左右四个方向的格子相连,墙不可通过

请你找出有多少个连通块,以及各个连通块的大小

输入格式

第一行输入两个个正整数 R, C

第二行输入一个正整数 T

接下来每行输入两个正整数 x_i, y_i 表示墙的坐标

输出格式

第一行输出连通块的个数

第二行输出若干个数,各个连通块的大小(升序输出)

将每个墙壁的横纵坐标分别进行离散化

墙壁 (1,5) 和 (5,5) 在离散化后的迷宫中坐标直接相邻

将中间空白区域压缩了

将每个墙壁的坐标 (x, y) 及 $(x \pm 1, y \pm 1)$

加入数组进行离散化可避免上述情况

预处理出所有空白区域的原始大小

进行 DFS/BFS 即可

数据规模

对于全部的数据 $1 \leq R, C \leq 10^9, 1 \leq T \leq 200, 1 \leq x_i \leq R, 1 \leq y_i \leq C$

Z 函数

对于字符串 S

定义函数 $Z[i]$ 表示 S 和 $\mathrm{Suffix}(S, i)$ 的最长公共前缀 (Longest Common Prefix) 长度

特别地 $Z[0] = 0$

如下展示了对于不同字符串的 Z 函数

$$Z(\text{"aaaaa"}) = [0, 4, 3, 2, 1]$$

$$Z(\text{"aaabaab"}) = [0, 2, 1, 0, 2, 1, 0]$$

$$Z(\text{"abacaba"}) = [0, 0, 1, 0, 3, 0, 1]$$

不难想出 $O(|S|^2)$ 的求解方法 (枚举 i 逐位比对)

一般将 $O(|S|)$ 计算 Z 函数的算法称为 **Z Algorithm** (国内称其为 扩展 KMP)

Z Algorithm

对于 i 称区间 $[\,i\,,\ i+z[i]-1\,]$ 是 i 的 **匹配段**（也称为 $\mathrm{Z-box}$）

考虑从 $0\ \sim |\mathrm{S}|-1$ 顺次计算 $\mathrm{Z}[i]$，在计算 $\mathrm{Z}[i]$ 的过程中利用已计算好的 $\mathrm{Z}[0],\ldots,\mathrm{Z}[i-1]$

计算过程中维护右端点最靠右的 $\mathrm{Z-box}$,记作 $[\mathrm{L},\mathrm{R}]$

根据定义 $\mathrm{S}[\mathrm{L}\ldots\mathrm{R}]$ 是 S 的前缀

在计算 $\mathrm{Z}[i]$ 时保证 $\mathrm{L}\le i$，初始时令 $\mathrm{L}=\mathrm{R}=0$

在计算 $\mathrm{Z}[i]$ 的过程中

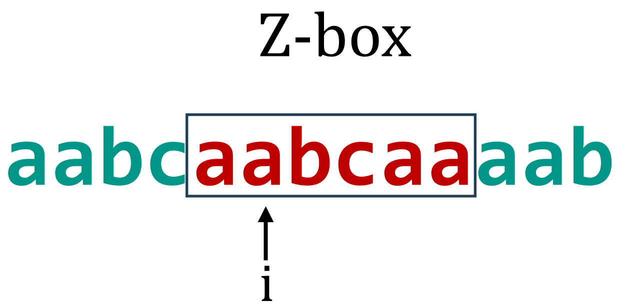
- 若 $i\le \mathrm{R}$

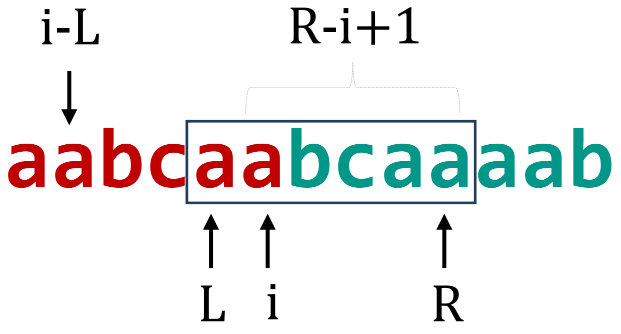
  根据 $[\mathrm{L},\mathrm{R}]$ 的定义有 $\mathrm{S}[i\ldots\mathrm{R}]=\ \mathrm{S}[i-\mathrm{L}\ldots\mathrm{R}-\mathrm{L}]$

  因此 $\mathrm{Z}[i]\ge\min(\mathrm{Z}[i-\mathrm{L}],\mathrm{R}-i+1)$

  若 $\mathrm{Z}[i-\mathrm{L}]<\ \mathrm{R}-i+1$

  则 $\mathrm{Z}[i]=\ \mathrm{Z}[i-\mathrm{L}]$





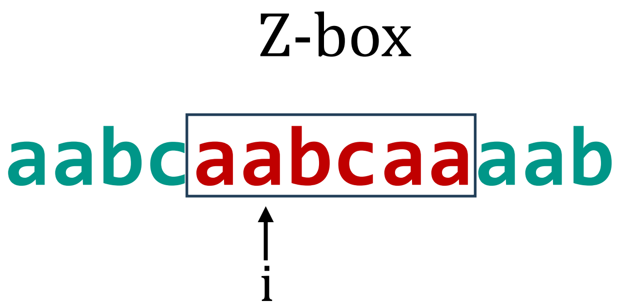
Z Algorithm

对于 i 称区间 $[\, i\,,\ i+z[i]-1\,]$ 是 i 的 **匹配段** (也称为 $\mathrm{Z-box}$)

考虑从 1 ~ |S| 顺次计算 $\mathrm{Z}[i]$

在计算 $\mathrm{Z}[i]$ 的过程中利用已计算好的 $\mathrm{Z}[0], \dots, \mathrm{Z}[i-1]$

计算过程中维护右端点最靠右的 $\mathrm{Z-box}$ ，记作 $[\mathrm{L}, \mathrm{R}]$



根据定义 $\mathrm{S}[\mathrm{L} \dots \mathrm{R}]$ 是 S 的前缀

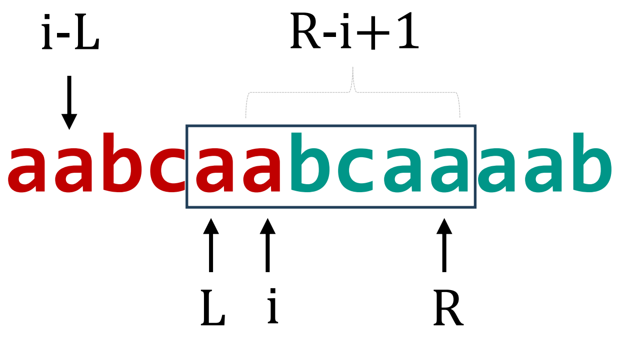
在计算 $\mathrm{Z}[i]$ 时保证 $\mathrm{L} \le i$, 初始时令 $\mathrm{L}=\mathrm{R}=0$

在计算 $\mathrm{Z}[i]$ 的过程中

- 若 $i \le \mathrm{R}$

  根据 $[\mathrm{L}, \mathrm{R}]$ 的定义有 $\mathrm{S}[i \dots \mathrm{R}] = \mathrm{S}[i-\mathrm{L} \dots \mathrm{R}-\mathrm{L}]$

  因此 $\mathrm{Z}[i] \ge \min(\mathrm{Z}[i-\mathrm{L}], \mathrm{R}-i+1)$



Z Algorithm

若 $Z[i-L] < R-i+1$

则 $Z[i] = Z[i-L]$

否则 $Z[i-L] \geq R-i+1$

令 $Z[i] = R-i+1$

再枚举下一个字符扩展 $Z[i]$ 直到无法扩展

- 若 $i > R$

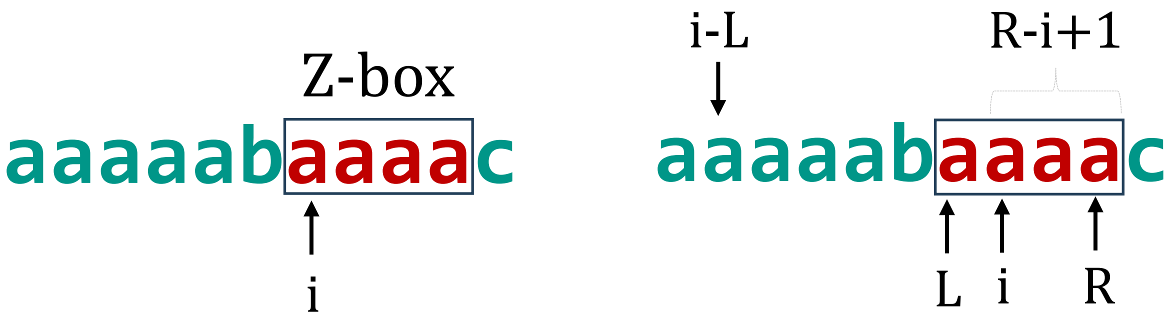
从 $S[i]$ 开始比较,暴力求出 $Z[i]$

在求出 $Z[i]$ 后，若 $i+Z[i]-1 > R$

需要更新 $[L,R]$，即令 $L=i$， $R=i+Z[i]-1$

不难看出 R 至多移动 $|S|$ 次

时间复杂度 $O(|S|)$

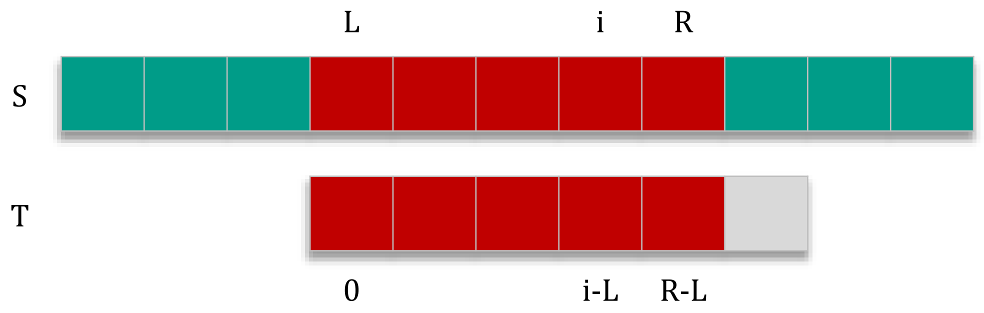



```
void getZ(char str[])
{
    int len = strlen(str), l = 0, r = 0;
    Z[0] = 0;
    for (int i = 1; i < len; i++)
    {
        if (i <= r && Z[i - l] < r - i + 1)
            Z[i] = Z[i - l];
        else
        {
            Z[i] = max(0, r - i + 1);
            while (i + Z[i] < len && str[Z[i]] == str[i + Z[i]])
                Z[i]++;
        }
        if (r < i + Z[i] - 1)
            l = i, r = i + Z[i] - 1;
    }
}
```

Extend KMP

给定主串 S 和模式串 T

定义数组 extend[i] 为 Suffix(S, i) 与 T 的最长公共前缀长度



同样维护一个区间 [L， R] 使得 $S[L \dots R] = \text{Prefix}(T, R-L)$

不难发现 $S[i \dots R] = T[i-L \dots R-L]$

仅需考察 T 的 $Z[i-L]$

若 $Z[i-L]$ 小于 $R-L+1$, 那么 $\text{extend}[i] = Z[i-L]$

否则对于超出的部分暴力匹配

```
void exKMP(char s[], char t[])
{
  getZ(t);
  int len1 = strlen(s), len2 = strlen(t), l = 0, r = 0;
  while (ext[0] < len1 && ext[0] < len2 && s[ext[0]] == t[ext[0]])
    ext[0]++;
  for (int i = 1; i < len1; i++)
  {
    if (i <= r && Z[i - l] < r - i + 1)
      ext[i] = Z[i - l];
    else
    {
      ext[i] = max(0, r - i + 1);
      while (i + ext[i] < len1 && s[i + ext[i]] == t[ext[i]])
        ext[i]++;
    }
    if (r < i + ext[i] - 1)
      l = i, r = i + ext[i] - 1;
  }
}
```

Extend KMP

不难想到当 S = T 时, extend为 Z 函数

当 S $\neq$ T 时

设 \$ 为字符集外字符，求 T + \$ + S 的 Z 函数

则 extend$[i]$ = Z[$|T| + i + 1$]

时/空间复杂度 $O(|S| + |T|)$

```
void exKMP(string s, string t)
{
    getZ((t + "#" + s).c_str());
    int len1 = s.size(), len2 = t.size();
    for (int i = 0; i < len1; i++)
        ext[i] = Z[len2 + i + 1];
}
```

	0	1	2	3	4	5	6
S	A	A	A	A	B	A	A
T	A	A	A	A	A		
extend	4	3	2	1	0	2	1

#2794、匹配统计

题目描述

阿轩在纸上写了两个字符串,分别记为 A 和 B

利用在数据结构与算法课上学到的知识,他很容易地求出了“字符串 A 从任意位置开始的后缀子串”与字符串 B 匹配的长度

不过阿轩是一个勤学好问的同学,他向你提出了 Q 个问题:

在每个问题中,他给定你一个整数 x ,请你告诉他有多少个位置,满足字符串 A 从该位置开始的后缀子串与 B 匹配的长度恰好为 x

例如: $A = \mathrm{aabcde}$, $B = \mathrm{ab}$,则 A 有 $\mathrm{aabcde,abcde,bcde,cde,de,e}$ 这 6 个后缀子串,它们与 $B = \mathrm{ab}$ 的匹配长度分别是 1、2、0、0、0、0

因此 A 有 4 个位置与 B 的匹配长度恰好为 0 ,有 1 个位置的匹配长度恰好为 1 ,有 1 个位置的匹配长度恰好为 2

输入格式

第一行输入三个整数 N, M, Q ,分别表示 A 串长度、B 串长度、问题个数

第二行输入字符串 A ,第三行输入字符串 B

接下来 Q 行每行输入 1 个整数 x ,表示一个问题

输出格式

输出共 Q 行,依次表示每个问题的答案

数据范围

对于全部的数据 $1 \leq N, M, Q, x \leq 200000$

#2794、匹配统计

思路1

求出 A, B 的 hash 值

考虑 A 的后缀 A[i , |A| − 1]

对于 A[i , |A| − 1] 的一个前缀能否成为 B 的前缀具有单调性

不难二分得出其最大 LCP 长度

差分维护即可

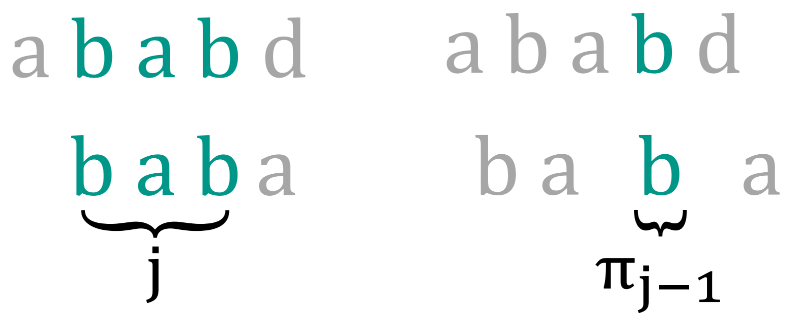
思路2

令 d_x 表示 LCP 长度**至少**为 x 的个数

求出 B 的前缀函数 π , 在 A 中匹配 B

考虑 A[0 , i] 与 B[0, $j-1$] 且已匹配了 j 个字符

这意味着 B[0, $j-1$] 与 A[$i-j+1$, |A|] 的 LCP 为 j



#2794、匹配统计

但 B[1 , $\pi[j-1]$] 也与 A[$i-\pi[j-1]+1$, |A|] 存在 LCP

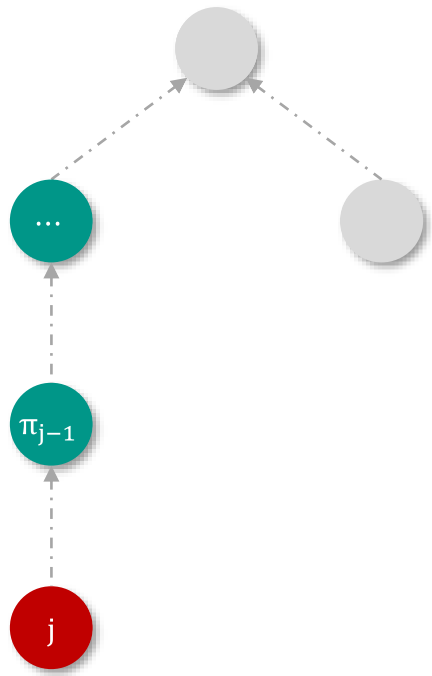
…

仅需令 $d_j \leftarrow d_j+1$, 最后 |B| → 1 遍历 d 数组 , 令 $d_{\pi[j]} \leftarrow d_{\pi[j]}+d_j$ 即可

对于长度为 x 的贡献即为 $d_x - d_{x+1}$

时间复杂度 $O(|A|+|B|)$

可理解为 boder 构成树形结构 (该树型结构被称为 border 树) , 树上差分维护该信息



思路3

求出 extend[i]

对于每个 extend[i] 计数

时间复杂度 $O(|A|+|B|)$

#2795、扩展回文串

题目描述

输入多个字符串

对于每个字符串 S ,求出一个字符串 S'

S' 需要满足:

- S 为 S' 的前缀
- S' 是一个回文字符串
- $|S'|$ 应尽可能小

对于每个 S ,输出 S' ,每行输出以换行符结尾

输入格式

输入包含多组数据

每组输入一各仅由小写字母组成的字符串 S

输出格式

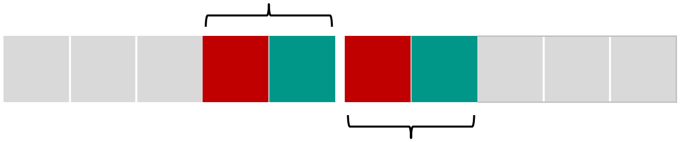
每组输出输出一行 S'

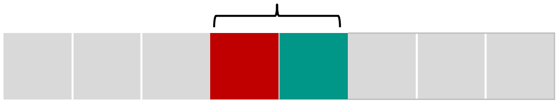
数据规模

对于全部的数据, $|S|$ 仅由大小写字母组成 $|S| \le 10^5, \sum |S| \le 10^6$

记 **rev**(S) 为 S 的翻转串

S + **rev**(S) 必然是回文串，考虑将其变短





#2795、扩展回文串

思路1

rev(S) 与 S 的 LCP 可作为公共部分

预处理 **rev**(S) 与 S 的 hash 值

枚举找出 LCP 长度即可

时间复杂度 $O(\sum|S|)$

思路2

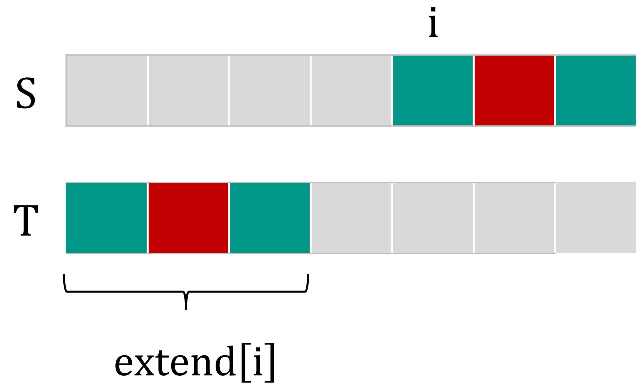
令模式串为 **rev**(S) ，对 S 求出 extend$[i]$

若有 $i + \text{extend}[i] - 1 = |S|$

说明 $\text{Suffix}(S, i)$ 可作为公共部分

找出最小满足条件的 i 可使得公共部分最大

时间复杂度 $O(\sum|S|)$



#2795、扩展回文串

思路3

设 \$ 为字符集外字符

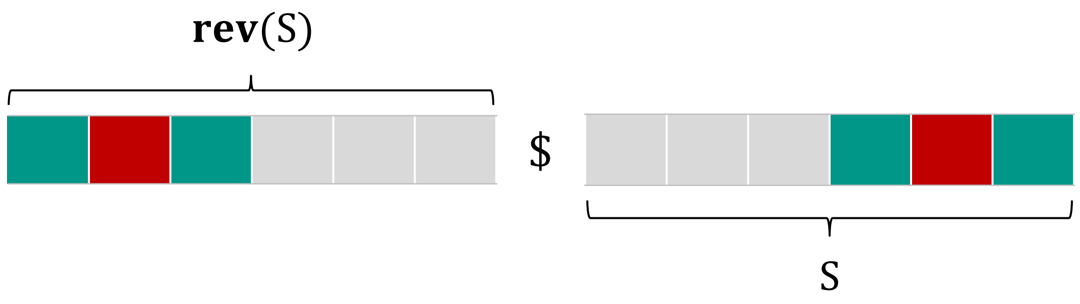
令 $X = \textbf{rev}(S) + \$ + S$

思路2 中满足 $i + \text{extend}[i] - 1 = |S|$ 的 i

在 X 中为 border ，最小 i 对应 X 的最大 border

求出前缀函数即可

时间复杂度 $O(\sum|S|)$



换言之 扩展 KMP (Z Algorithm) 也可求出字符串的 border

本题也可 Manacher 解决

#3250、 字符串函数

题意描述

对于一个长度为 N 的字符串 S 和一个整数 $i \in [0, N]$

定义函数 $f_i(S)$ 所得的字符串为以下三者顺次连接：

- S 的前 i 个字符
- 将 S 翻转得到的字符串
- S 的后 $N - i$ 个字符

如当 $S = \text{“abc”}, i = 2$ 时

$$f_i(S) = \text{“ab”+”cba”+”c”} = \text{“abcbac”}$$

现在有一个长度为 $2N$ 的字符串 T

你需要求出一对 (S, i) 满足 $f_i(S) = T$

若不存在输出 -1

记 $\mathbf{rev}(S)$ 表示字符串 S 的倒序

数据规模

对于 8% 的数据 $\sum N \le 100$

对于 16% 的数据 $\sum N \le 10000$

对于全部的数据 $\sum N \le 2 \times 10^6, |T| = 2N$

#3250、 字符串函数

思路1

根据下标 i 可将 T 分为

$$\overbrace{\mathrm{T}[0 \ldots\ i-1]}^{i} \quad \boxed{\overbrace{\mathrm{T}[i \ldots N-1]}^{N-i} \quad \overbrace{\mathrm{T}[N \ldots N+i-1]}^{i}} \quad \overbrace{\mathrm{T}[N+i \ldots 2N-1]}^{N-i}$$

满足条件的 i

- $\mathrm{T}[0 \ldots\ i-1] = \mathbf{rev}(\mathrm{T}[N \ldots N+i-1])$
- $\mathrm{T}[i \ldots N-1] = \mathbf{rev}(\mathrm{T}[N+i \ldots 2N-1])$

对 T 正向求出其 hash 值 , 也对 T 的逆序求出其 hash 值

仅需验证正反两个 hash 值即可判断

对于满足条件的 i , 令 $\mathrm{S} = \mathrm{T}[0 \ldots\ i-1] + \mathbf{rev}(\mathrm{T}[N+i \ldots 2N-1])$ 即为答案

时间复杂度 $O(\sum |\mathrm{T}|)$

单模 hash 无法通过本题

#3250、 字符串函数

思路2

令 $\mathrm{A} = \mathrm{T}[0 \ldots N-1]$, $\mathrm{B} = \mathbf{rev}(\mathrm{T}[N \ldots 2N-1])$

再令 $\mathrm{X} = \mathrm{B} + \mathrm{B}$

对于一个满足条件的 i 必然可在 X 中找到完整的 A

求出 A 的前缀函数 , 匹配即可

记 匹配成功的起始位置为 p , 那么 $n-p+1$ 即为 所求 i

因要求 i 最小 , 那么应当找出最靠后的匹配位置

时间复杂度 $O(\sum |\mathrm{T}|)$

$$\mathrm{T} \quad \text{abced cbade}$$

$$\mathrm{X} \quad \overbrace{\text{edabc}}^{\mathrm{B}}\ \overbrace{\text{edabc}}^{\mathrm{B}}$$

$$\mathrm{A} \quad \text{abc ed}$$

#3250、 字符串函数

思路3

令 $\mathrm{A} = \mathrm{T}[0 \ldots N-1]$, $\mathrm{B} = \mathbf{rev}(\mathrm{T}[N \ldots 2N-1])$

思路1 中的两个条件等价于

- $\mathrm{A}[0, i-1] \ = \ \mathrm{B}[N-i, N-1]$
- $\mathrm{A}[i, N-1] \ = \ \mathrm{B}[0, N-i-1]$

令 $\mathrm{X} = \mathrm{A} + \mathrm{B}$, $\mathrm{Y} \ = \ \mathrm{B} + \mathrm{A}$

记 ZX 为 X 的 Z 函数 , ZY 为 Y 的 Z 函数

那么满足 $\mathrm{A}[0, i-1] \ = \ \mathrm{B}[N-i, N-1]$ 时有 $\mathrm{ZX}[2N-i] = i$

满足 $\mathrm{A}[i, N-1] \ = \ \mathrm{B}[0, N-i-1]$ 时有 $\mathrm{ZY}[N+i] = N-i$

预处理 Z 函数 , 验证每个 i 即可

时间复杂度 $O(\sum |\mathrm{T}|)$

T abced cbade

$$\mathrm{X} \quad \overbrace{\mathrm{abced}}^{\mathrm{A}} \ \overbrace{\mathrm{edabc}}^{\mathrm{B}}$$

i − 1 2N − i

$$\mathrm{Y} \quad \overbrace{\mathrm{edabc}}^{\mathrm{B}} \ \overbrace{\mathrm{abced}}^{\mathrm{A}}$$

N − i N + i

#3260、珍珠项链

题目描述

Mas 得到了由 N 颗不同颜色的珍珠串成的长链

其中第 i 颗珍珠用字符 S_i 表示

现在 Mas 希望从长链的开头截取连续一段制作一条项链

Mas 认为具有如下性质的项链才是美观的

- 项链形如 $\mathrm{ABAB}\cdots\mathrm{ABA}$
- 项链需要有 $K+1$ 个 A , K 个 B

其中 A 和 B 都为长链中的连续一段,且 A, B 可为空

现在请你回答 $S[0\ldots i]$ 能否制作成项链

输入格式

第一行输入两个整数 N, K

第二行输入一个字符串 S 表示长链(S 仅由大小写字母组成)

输出格式

输出一个 01 序列,其中第 i 个字符表示 $S[0\ldots i]$ 能否制作成项链

若能则为 1 否则为 0

数据规模

对于 16% 的数据 $1\leq N,K\leq 400$

对于 36% 的数据 $1\leq N,K\leq 10000$

对于 68% 的数据 $1\leq N,K\leq 500000$

对于全部的数据 $1\leq N,K\leq 10^6$

记 T ＝ A＋B

合法的前缀可表示为

$$\overbrace{\mathrm{TTT}\cdots\mathrm{TTT}}^{\mathrm{K}}\,\mathrm{A}$$

#3260、珍珠项链

思路1

求出前缀函数 π，对于前缀 $\mathrm{S}[0 \ldots i-1]$ 记其最小周期长度为 $p = i - \pi[i-1]$

若满足 $|\mathrm{T}| = pq$（将 q 个最小周期作为整体）且有

$$i = K \times |\mathrm{T}| + |\mathrm{A}| \Rightarrow \left\lfloor \frac{i}{pq} \right\rfloor = K$$

$$i = (\mathrm{K}+1) \times |\mathrm{T}| - |\mathrm{B}| \Rightarrow \left\lceil \frac{i}{pq} \right\rceil = K + 1$$

可满足恰有 K 个 T，又由于恰有 $K+1$ 个 A

仅需在

$$\left\lceil \frac{i}{p(K+1)} \right\rceil \sim \left\lfloor \frac{i}{pK} \right\rfloor$$

找出整数解 q 即可满足

时间复杂度 $O(n)$

#3260、珍珠项链

思路2

处理出 S 的 hash 值，记 $|\mathrm{T}| \times K = p$

考虑枚举 |T|，不难想到其上界为 $\left\lfloor \frac{n}{k} \right\rfloor$

$$\overbrace{\mathrm{T}\ \mathrm{T}\ \mathrm{T}\ \cdots\ \mathrm{T}\ \mathrm{T}\ \mathrm{T}}^{\mathrm{K}}\ \mathrm{T}'$$

若 T 重复 K 次能覆盖 $\mathrm{Prefix}(\mathrm{S}, p-1)$

对于长度不小于 p 的前缀串 $\mathrm{Prefix}(\mathrm{S}, i)$

若 $\mathrm{Prefix}(\mathrm{S}, i)$ 存在一后缀 T' 且 T' 也为 T 的前缀

那么对于 $\mathrm{Prefix}(\mathrm{S}, p-1) \sim \mathrm{Prefix}(\mathrm{S}, p+|\mathrm{T}'|-1)$ 能够贡献答案 (能够满足题目要求)

对于 T’ 是否为 T 前缀显然满足单调性

二分求出 T’ 长度，差分维护贡献即可

时间复杂度 $O(n\log n)$

#3260、珍珠项链

思路3

求出 Z 函数

考虑枚举 |T| ，上界为 $\left\lfloor \frac{n}{k} \right\rfloor$

记 $|T| \times K = p$, 根据 Z 函数定义

若 Z[|T|] ≥ $|T| \times (K-1)$, 那么说明 T 重复 K 次能覆盖 $\text{Prefix}(S, p-1)$

对于长度不小于 p 的前缀串 $\text{Prefix}(S, i)$

若 $\text{Prefix}(S, i)$ 存在一后缀 T' 且 T' 也为 T 的前缀串

不难想到 |T'| 最大长度为 min(|T| , Z[p])

差分维护贡献即可

时间复杂度 $O(n)$

Trie

字典树（Trie）是一种特殊的树

Trie 使用边存储字母，从根到某一结点的路径代表了一插入的前缀串

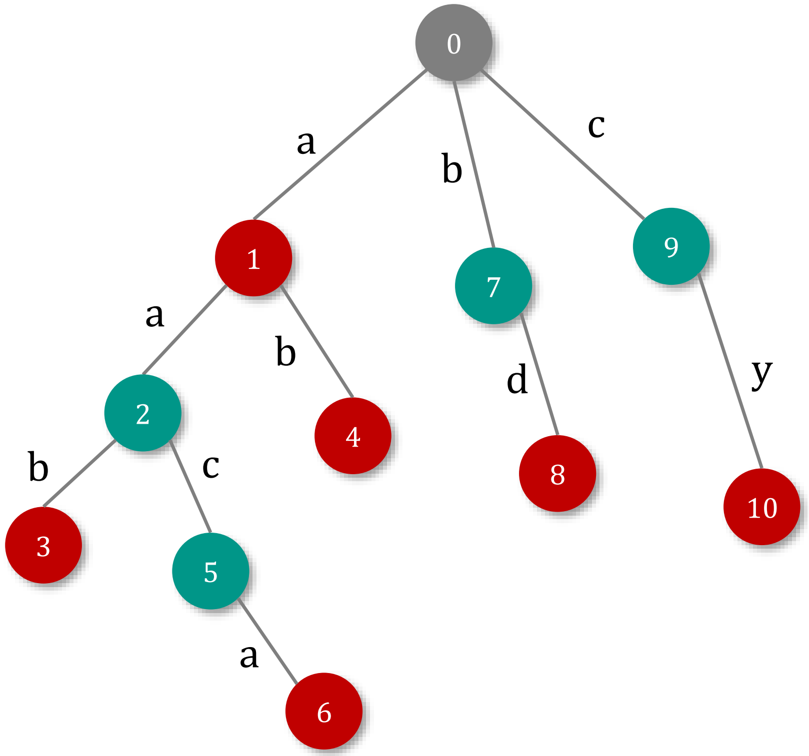
如右图 $0 \to 1 \to 2 \to 5 \to 6$ 代表字符串 “aaca”

使用 $\delta(u,c)$ 表示节点 u 的下一字符 c 表示的节点

$\delta(u,c)$ 对应字符串为 u 对应字符串追加字符 c 形成的字符串

为了区分完整串和前缀串，节点有时增加标记以区分

Trie 是 AC自动机 的一部分



a

aab

ab

aaca

ab

bd

cy

Trie

考虑将字符串 S 加入 Trie

从 Trie 根节点出发，遍历字符串 S

若不存 S[i] 对应节点则创建节点

移动到 S[i] 对应节点，继续遍历 S

时间复杂度 $O(|S|)$

在 Trie 中查找前缀串 S

从 Trie 根节点出发，遍历字符串 S

若不存 S[i] 对应节点返回 false

移动到 S[i] 对应节点，继续遍历 S

最后返回结果

时间复杂度 $O(|S|)$

```cpp
int t[MAXN][26], cnt[MAXN], pos;
void insert(char str[])
{
    int p = 0;
    for (int i = 0; str[i]; i++)
    {
        int c = str[i] - '0';
        if (!t[p][c])
            t[p][c] = ++pos;
        p = t[p][c];
    }
    cnt[p]++;
}
int find(char str[])
{
    int p = 0, res = 0;
    for (int i = 0; str[i]; i++)
    {
        int c = str[i] - '0';
        if (!t[p][c])
            return res;
        p = t[p][c];
        res += cnt[p];
    }
    return res;
}
```

#650、Phone List

题目描述

给定 n 个长度不超过 10 的数字串

问其中是否存在两个数字串 S,T

使得 S 是 T 的前缀

输入格式

第一行一个整数 T ,表示数据组数
对于每组数据
第一行一个数 n
接下来 n 行输入 n 个数字串

输出格式

对于每组数据
若存在两个数字串 S,T
使得 S 是 T 的前缀则输出 `NO` 否则输出 `YES`

请注意此处结果与输出的对应关系!

数据范围与提示

对于 100% 的数据，$1 \leq T \leq 40, 1 \leq n \leq 10^4$

将所有字符串插入 Trie

对于每个字符串做一次查找

统计查找路径上经过的单词

若数量大于 1 说明存在

#655、Secret Message 秘密信息

题目描述

贝茜正在领导奶牛们逃跑，为了联络,奶牛们互相发送秘密信息

信息是二进制的,共有 M 条

反间谍能力很强的 FJ 已经部分拦截了这些信息,知道了第 i 条二进制信息的前 b_i 位

他同时知道,奶牛使用 N 条密码，但是他仅仅了解第 j 条密码的前 c_j 位

对于每条密码 j ,他想知道有多少截得的信息能够和它匹配

也就是说,有多少信息和这条密码有着相同的前缀

当然,这个前缀长度必须等于密码和那条信息长度的较小者

输入格式

第一行输入 N 和 M

之后 N 行描述秘密信息,之后 M 行描述密码
每行先输入一个整数表示信息或密码的长度,之后输入这个信息或密码

所有数字之间都用空格隔开

输出格式

共 M 行,输出每条密码的匹配信息数

数据范围

对于 100% 的数据, $1 \leq M, N \leq 50000, 1 \leq b_i, c_j \leq 10000$

位的总数即 $\sum B_i + \sum C_i$ 不超过 500000

#655、Secret Message 秘密信息

将所有描述信息插入 Trie

各节点维护 dCnt 表示当前节点往下所有路径上单词数量

同时维护 cnt_u 表示节点 u 到根的路径上前缀串数量

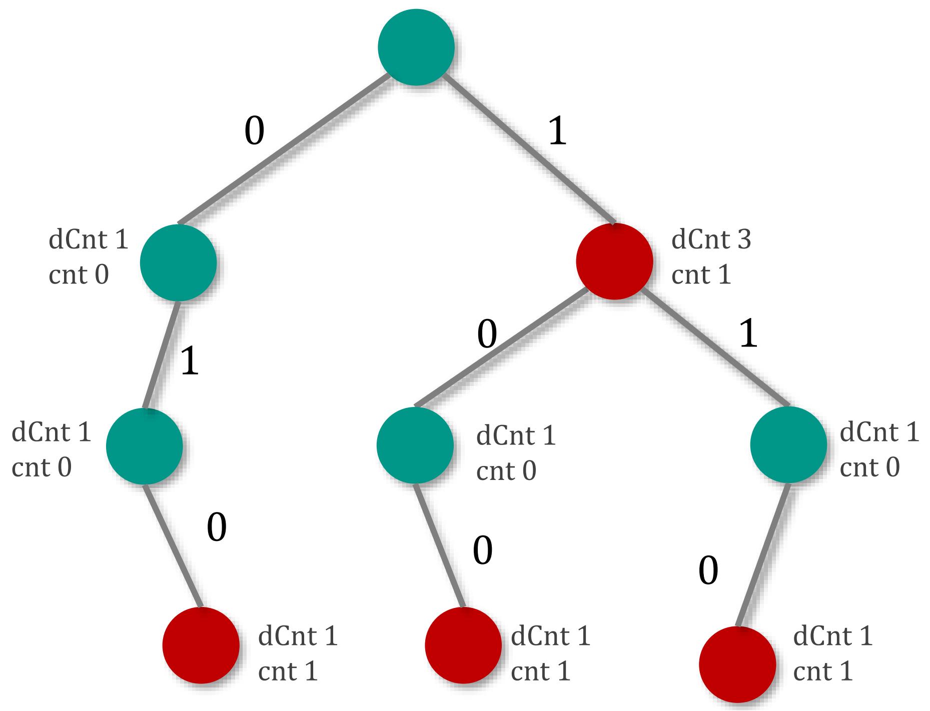
对于每个查找的串

累加串经过路径上的 cnt_p，假设最终到达点 p

最终答案为

$$\sum \mathrm{cnt}_u + \mathrm{dCnt}_p - \mathrm{cnt}_p$$

时间复杂度 $O(\sum B + \sum C)$



#651、The XOR Largest Pair

题目描述

在给定的 N 个整数 $A_1, A_2, \ldots, A_N$ 中选出两个进行异或运算

得到的结果最大是多少?

输入格式

第一行一个整数 N

第二行 N 个整数 A_i

输出格式

一个整数表示答案

样例输入

```
5
2 9 5 7 0
```

样例输出

```
14
```

每个 a_i 都可转化为长度 31 的 01 序列

对于一个数$(101)_2$

若存在 $(0??)_2$ 根据异或性质可获得 2^2 贡献

比任何 $(1??)_2$ 序列获得的贡献都大

数据范围

对于 100% 的数据, $1 \le N \le 10^5, 0 \le A_i < 2^{31}$

#651、The XOR Largest Pair

将所有 a_i 转为长度 31 的 01 序列（不足 31 位高位补 0）插入 Trie 中，贪心求解

对于 a_i 转为的 01 序列 S，从高到低遍历在 Trie 进行移动

对于第 x 位

　　若存在 $S[x] \oplus 1$ 对应分支，移动并累加 2^x 贡献

若不存在

　　移动到 $S[x]$ 对应分支

时间复杂度 $O(31 \times n)$

一般将字符集为 {0,1} 的 Trie 称为 01 Trie

```
int find(int x)
{
  int p = 0, res = 0;
  for (int i = 30; ~i; i--)
  {
    int c = x >> i & 1;
    if (t[p][!c])
    {
      res += 1 << i;
      p = t[p][!c];
    }
    else
      p = t[p][c];
  }
  return res;
}
```

#656、 树上路径异或和

题目描述

给定一棵 n 个点的带权树,求树上最长的异或和路径

输入格式

第一行一个整数 n

接下来 $n-1$ 行每行三个整数 u, v, w ,表示 u, v 之间有一条长度为 w 的边

> *结点下标从 1 开始到 N*

输出格式

输出一行一个整数,表示答案

样例输入

```
4
1 2 3
2 3 4
2 4 6
```

样例输出

```
7
```

数据范围与提示

对于 100% 的数据, $1 \le n \le 10^5, 1 \le u, v \le n, 0 \le w < 2^{31}$

将 $u \to v$ 的边权 w 下放至 v 维护

令 d_u 表示 u 到 根节点的路径异或和

$d_{1\sim n}$ 不难通过 DFS 一次求出

对于 $u \to v$ 的路径，其异或和即为 $d_u \oplus d_v$

$\text{LCA}_{u,v}$ 的权值为 $\text{LCA}_{u,v}$ 父节点到其权值，无需考虑

问题转为: 选出两数异或和最大

求解方式同 #651、The XOR Largest Pair

LAB
实验舱
青少年编程
走近科学 走进名校

谢谢观看