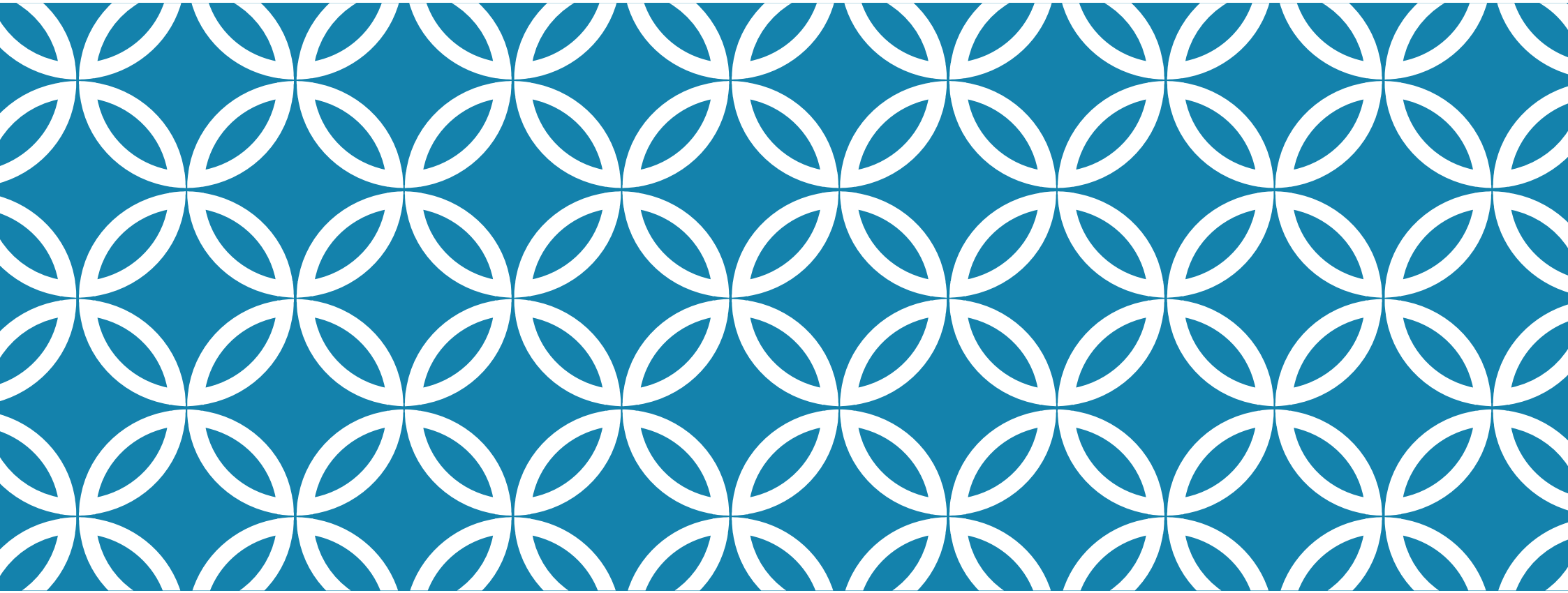


TI WAI HUA

MANACHAR算法

CZC

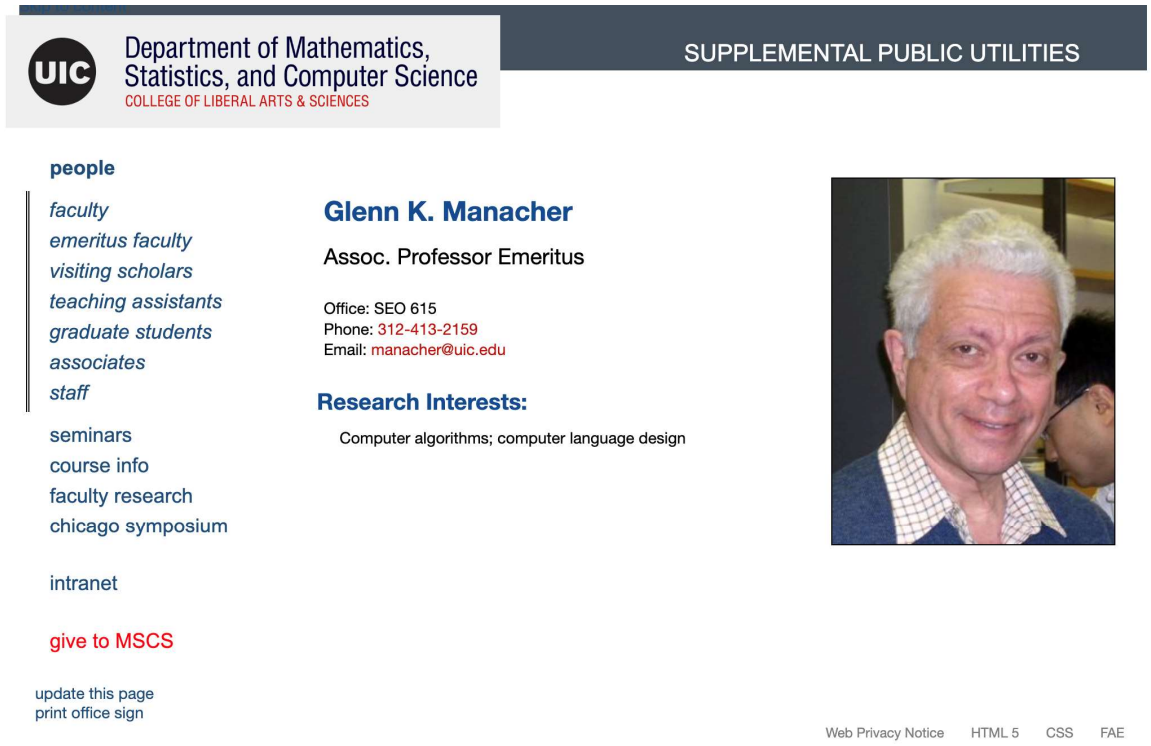
2023-06-07

v2.1

算法背景

马拉车算法 Manacher‘s Algorithm 是用来查找一个字符串的最长回文子串的线性方法，由一个叫Manacher的人在1975年发明的，这个方法的最大贡献是在于将时间复杂度提升到了线性。

manacher 可能是他⬇



算法基本信息

解决问题：求字符串中的回文串

预处理：因为处理回文串要分奇偶长度分类讨论
干脆一步到位，在串之间和首尾插入#号（回文中不会出现的字符）
这样长度为N的字符串，就变成了N+N+1 的串，
即2N+1。 全部处理为奇数长度。

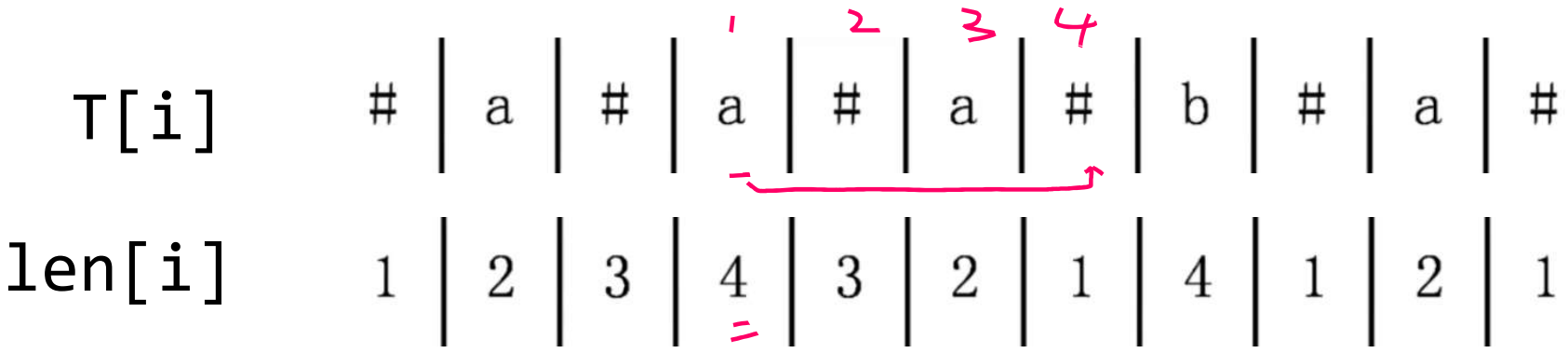
aaabbcc => #a#a#a#b#b#c#c#

Q这样处理影响回文性质么？

辅助数组LEN 重要性质

len[i]-1 就是以T[i]为中心的回文子串在原字符串中的长度。

例如：下图的a，len[pos[a]]-1 = 4 - 1 =3



辅助数组LEN 重要性质

len[i]-1 就是以T[i]为中心的回文子串在原字符串中的长度。

证明?

证明:

- 在转换后的字符串T中，显然回文字符串的长度为奇数。
- 假设已经求出len数组，则T[i]的最长回文字符串的长度为2*len[i]-1
- 记T[i]所在的最长回文字符串长度为N_i，加入的分隔符#的个数是N_i+1，总长度N_i+N_i+1
- 有公式: $N_i+N_i+1 = 2*len[i] -1$
- 所以 $N_i= len[i]-1$

求LEN数组

现在问题转化为了求len数组。

1. mx为之前找到过的最靠右的回文字符串子串最右的位置
2. mid为这个串的中间位置（加了#后的奇数串）
3. i为当前所求的位置
4. j为i关于mid的对称点

假设你有一个字符串

ABCSADWFAWSDAWASDFWFCAFW

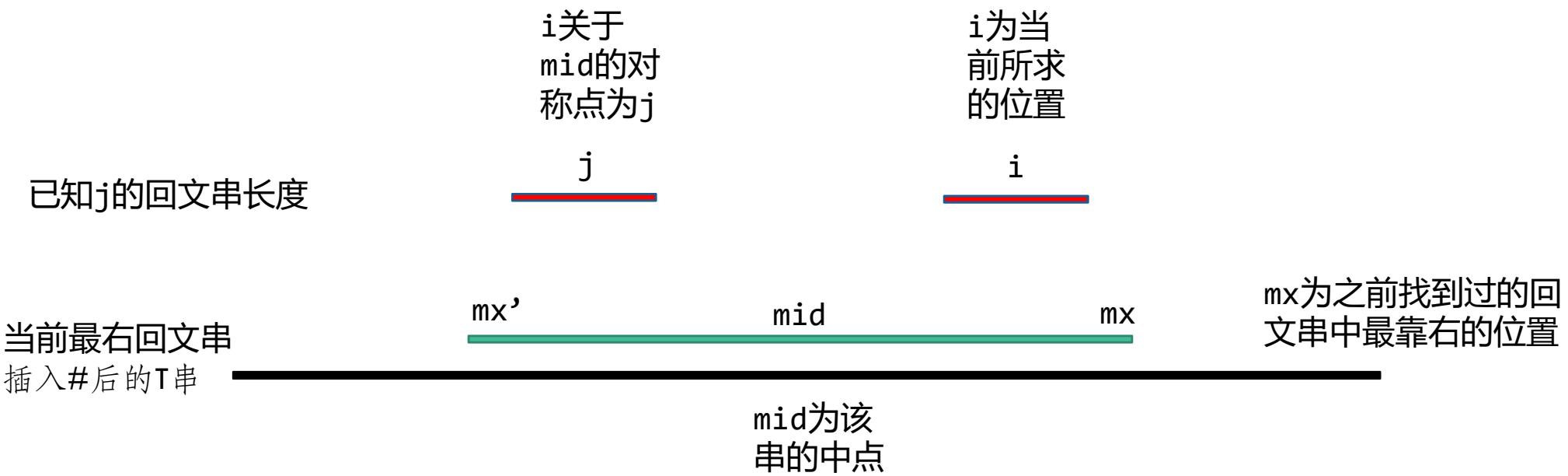
内容是什么不重要，抽象成线

Case1 当i<=mx时

Case1.1 且当T[j]回文串在[mx’, mx]内时(如图)

显然len[i]=len[j]
因为回文串的
对称串也是回文串

更严格的证明:
采取反证法
假定答案len[i]比当前更长?

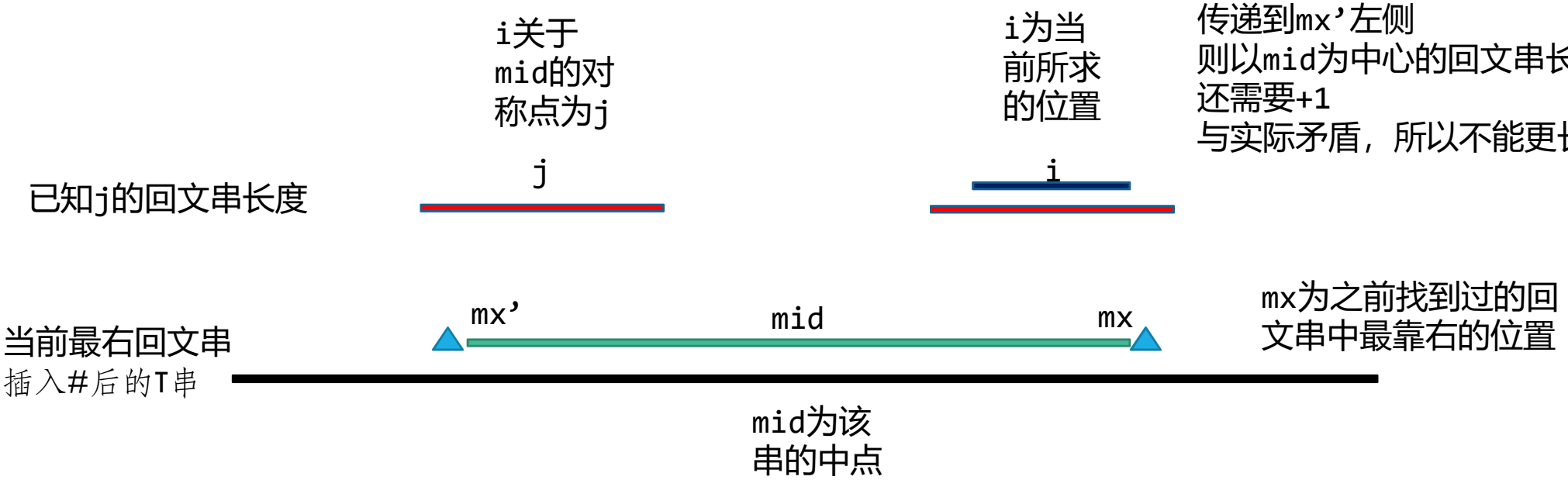


Case1 当i<=mx时

Case1.1 且当T[j]回文串在[mx’, mx]内时 显然len[i]=len[j]

Case1.2 且当T[j]回文串超过左边界 mx’时(如图) len[i]=mx-i+1

反证法可证明其正确性:
假定i中心的回文串还可更长
则需包括蓝色三角
由于j中心点的回文对称性
传递到mx’左侧
则以mid为中心的回文串长度
还需要+1
与实际矛盾，所以不能更长。



Case1 当i<=mx时

len[i]=len[j]
len[i]=mx-i+1

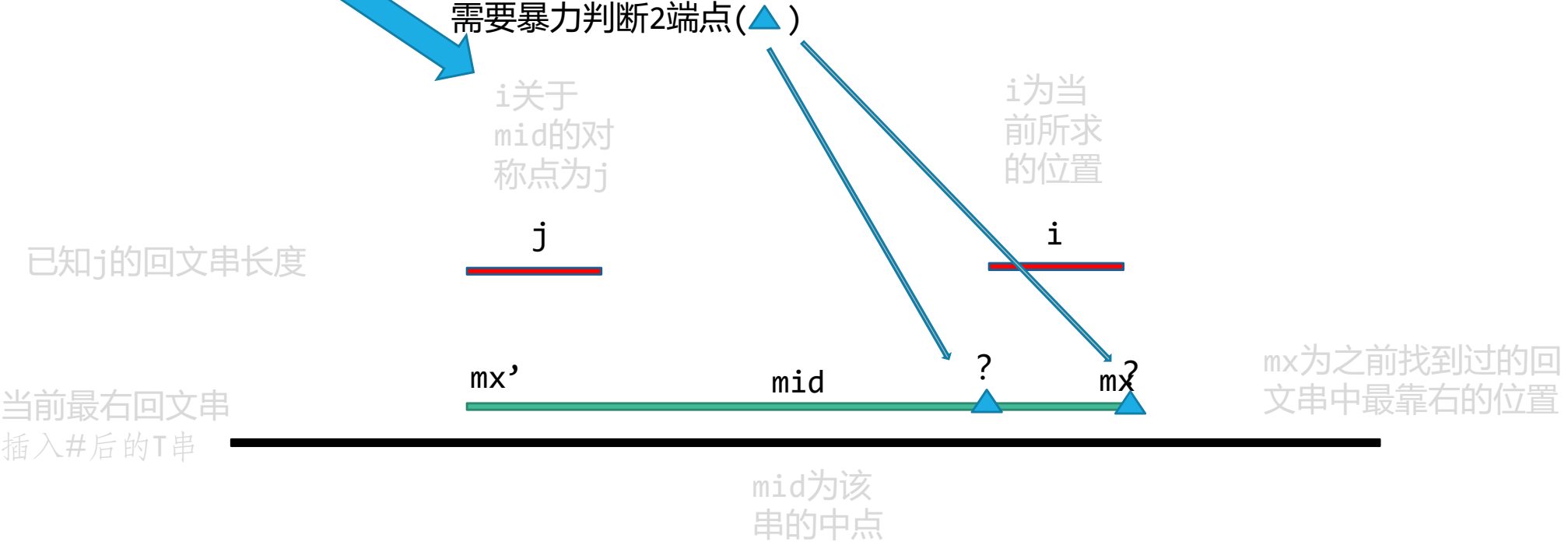
len[i]=min(len[j],mx-i+1)

实际上当len[i] == mx-i+1 == len[j] 的时候
没办法借助当前信息推测i中心回文串左右两端数据
还需要进一步处理

当 len[j] == mx-i+1 时
需要暴力判断2端点(△)

Case1 当i<=mx时

len[i]=min(len[j],mx-i)
再进行朴素算法检验是否可继续拓展



Case2 当i>mx时

朴素算。

伪代码：

```
FOR i in 1~n:
    IF i<=mx:
        len[i] =min(len[mid*2-i],mx-i)
    朴素检验len[i]是否还能拓展
    IF 找到更右的回文串:
        更新mid,mx
```

易错点：

数组越界：算到开头到0了还算。

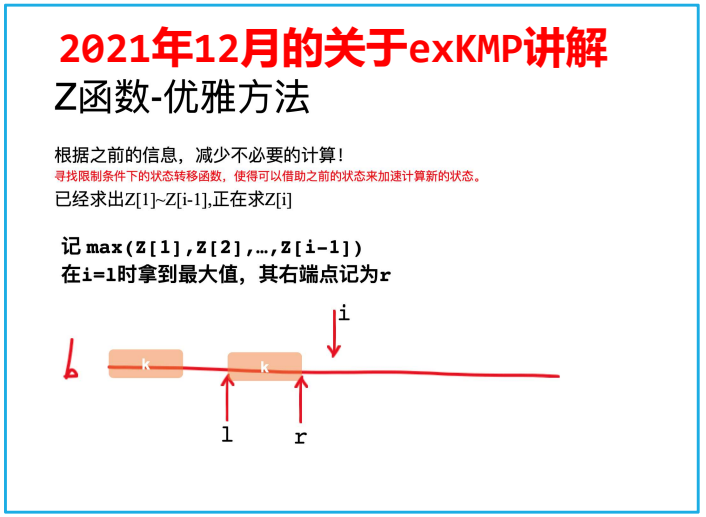
解决方法：在0位置添加一个特特殊的字符，匹配失败停止

空间开大一点，避免开O2后不必要的错误。比如2N+100

时间复杂度分析

```
FOR i in 1~n:
    IF i<=mx:
        len[i] =min(len[mid*2-i],mx-i)
    朴素检验len[i]是否还能拓展
    IF 找到更右的回文串:
        更新mid,mx
```

O(n)



得益于mx信息的记录
算法只有遇到未匹配点时
才会在O(1)的复杂度内完成匹配

代码实现

```
FOR i in 1~n:
    IF i<=mx:
        len[i] =min(len[mid*2-i],mx-i)
    朴素检验len[i]是否还能拓展
    IF 找到更右的回文串:
        更新mid,mx
```

```
int manachar(string s) {
    int mx = 0; 最右侧边界
    int pos = 0; ——— mx对应的对称中心
    int ans = -1;答案
    for (int ptr = 1; ptr < s.size() - 1; ++ptr) {
        if (ptr < mx)
            lens[ptr] = min (lens[2 * pos - ptr], mx - ptr);
        else
            lens[ptr] = 1;
        while (s[ptr - lens[ptr]] == s[ptr + lens[ptr]])
            lens[ptr]++;
        if (mx < ptr + lens[ptr]) {
            mx = ptr + lens[ptr];
            pos = ptr;
        }
        ans = max (ans, lens[ptr] - 1);
    }
    return ans;
}
```

start ↓ $ # a # b # a # c # ^ ↓ end

↑ ptr

关于pos对称的点

→ 当 lens[对称点] > mx时，说明应当从mx处开始扩展

当ptr > mx时，扩展mx的边界

扩展边界与对称中心

当前位置为中心的最大回文串长度

代码实现2

```
const int maxn = 2000000 + 5;
int lens[maxn];
string expand(string s) {
     string ss = "$";
     for (int i = 0; i < s.size(); ++i) {
          ss += '#';
          ss += s[i];
     }
     ss += "#^";
     return ss;
}
```

```
int manachar(string s) {
     int mx = 0;
     int pos = 0;
     int ans = -1;
     for (int ptr = 1; ptr < s.size() - 1; ++ptr) {
          if (ptr < mx)
               lens[ptr] = min (lens[2 * pos - ptr], mx - ptr);
          else
               lens[ptr] = 1;
          while (s[ptr - lens[ptr]] == s[ptr + lens[ptr]])
               lens[ptr]++;
          if (mx < ptr + lens[ptr]) {
               mx = ptr + lens[ptr];
               pos = ptr;
          }
          ans = max (ans, lens[ptr] - 1);
     }
     return ans;
}
```

#3465.「国家集训队」拉拉队排练

n个人从左到右排成一行，每个人手中都举了一个写有26个小写字母中的某一个的牌子。如果连续的一段人，有奇数个，并且他们手中的牌子所写的字母，从左到右和从右到左读起来一样，那么这一段人就被称作和谐小群体。

现在想找出所有和谐小群体，并且按照人的个数降序排序之后，前K个和谐小群体的人个数的乘积是多少。（若<k个 输出-1）

答案%19930726

N=1e6 K=1e9

一句话提炼问题：在长度为n的字符串中，求前k长的回文串长度之积。

#3465.「国家集训队」拉拉队排练

这是一个~~板子题~~
Manacher 求每个位置的回文半径（注意batab 的情况 需要统计t ata 和 batab）
只求奇数长度，那么就直接求即可，不需要额外添加无关字符#
再利用桶统计一下即可。

再次温馨提醒，注意只统计奇数长度。
因为回文字符串包含关系，其统计过程类似前缀和。

#3465.「国家集训队」拉拉队排练

N=1e6 K=1e9

这道题可能会在哪些细节上卡你?

1 快速幂

2 不开 long long 见祖宗

3 因为你没插# 所以之前的代码len[i]-1 需要进一步修改

具体问题具体分析，不能乱套模版

#3466.「国家集训队」最长双回文串

输入长度为n的串S，求S的最长双回文子串T,即可将T分为两部分X，Y，（|X|,|Y|≥1）且X和Y都是回文串。

2 <= abs(S) <= 1e5

Input: baacaabbacabb

out: 12

#3466.「国家集训队」最长双回文串

求2个回文串拼一起

求len时再额外多维护两个信息

L[i] 表示以i 开头的最长回文串的长度。

R[i] 表示以i 结尾的最长回文串的长度。

O(n) 再搜一遍即可。

和刚学LIS时的山峰题目类似。

练习时间

3478	「模板」Manacher 算法
2657	「POJ3974」Palindrome
3465	「国家集训队」拉拉队排练
3466	「国家集训队」最长双回文串
6724	回文匹配
6723	[THUPC2018]绿绿和串串
2460	「POI2010」Antisymmetry
6725	[JSOI2016] 扭动的回文串

建议复习一下hash