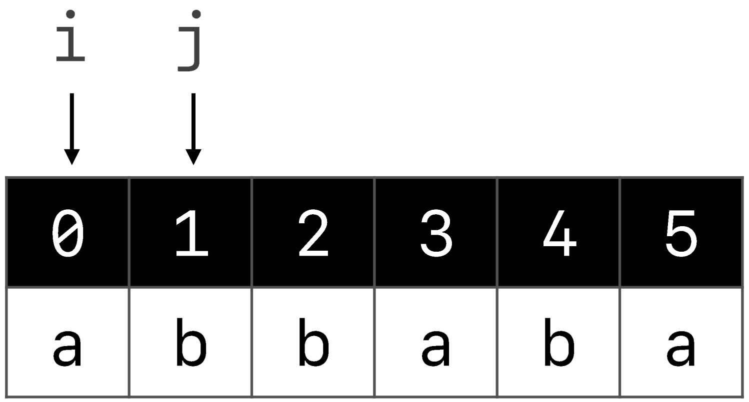
5. 最长回文子串

给定一个字符串 `s`，找到 `s` 中最长的回文子串。你可以假设 `s` 的最大长度为 1000。

```
输入: "babad"
输出: "bab"
注意: "aba" 也是一个有效答案。
```

```
输入: "cbbd"
输出: "bb"
```

暴力法



- 列举出所有的子串，时间复杂度： O(n^2)
- 检查每一个子串是否为回文串，每一个子串所需时间复杂度：O(n)
- 总共时间复杂度：O(n^3)，空间复杂度：O(1)

	j	b	a	b	a	d
i	dp	0	1	2	3	4
b	0	T	F	T	F	F
a	1		T	F	T	F
b	2			T	F	F
a	3				T	F
d	4					T

■ 动态规划解法

□ 其实是基于暴力法的优化，优化的部分：判断每个串是否为回文串

□ 时间复杂度：O(n^2)

□ 空间复杂度：O(n^2)

□ 空间复杂度可以优化至O(n)

■ 假设字符串（"babad"）为s，它的长度为n

■ dp是大小为n * n的二维数组，dp[i][j]表示s[i, j]是否为回文串，存储true、false

■ 如何求出dp[i][j]的值？分2种情况

① 如果s[i, j]的长度（j – i + 1）≤ 2时

□ 如果s[i]等于s[j]，那么s[i, j]是回文串，所以dp[i][j] = s[i] == s[j]

② 如果s[i, j]的长度（j – i + 1）> 2时

□ 如果s[i + 1, j – 1]是回文串，并且s[i]等于s[j]，那么s[i, j]是回文串

□ 所以dp[i][j] = dp[i + 1, j – 1] && (s[i] == s[j])

扩展中心法

- 假设字符串（"abbaba"）的长度为n，那么一共有n + (n – 1) == 2n – 1个扩展中心
- 时间复杂度：O(n^2)
- 空间复杂度：O(1)

基于扩展中心法的优化

i　r

0	1	2	3	4	5	6	7	8
b	a	b	b	b	a	b	a	a

- 算法的核心思想：由连续的相同字符组成的子串作为扩展中心
- 所以，字符串"babbbabaa"的扩展中心有
  - "b"、"a"、"bbb"、"a"、"b"、"aa"

- 核心逻辑
  - 找到右边第一个不等于s[i]的字符，记为位置r，i左边位置记为l
  - r作为下一次的i
  - 由l开始向左、r开始向右扩展，找到最长的回文子串

Manacher（马拉车）

			0		1		2		3		4		5		
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
cs	^	#	a	#	b	#	b	#	a	#	b	#	a	#	$
m	0	0	1	0	1	4	1	0	3	0	3	0	1	0	0

- 中间的#字符可以是任意字符，头部的^字符、尾部的$字符，必须是原字符串中不包含的字符
- m[i]的含义
  - 是以cs[i]为扩展中心的最大回文子串的长度（不包含#字符）
    - ✓ 最大回文子串在原字符串中的开始索引：(i – m[i]) >> 1
  - 是以cs[i]为扩展中心的最大回文子串的右半部分或左半部分的长度（包含#字符）
- 所以，Manacher算法的关键在于求出m数组

Manacher（马拉车）

			0		1		2		3		4		5		
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
cs	^	#	c	#	b	#	a	#	b	#	c	#	a	#	$
m	0	0	1	0	1	0	5	0							
		↑ l			↑ li		↑ c		↑ i			↑ r			

■ 已知

- 索引l、li、c、i、r的值分别为1、4、6、8、11
- cs[l, r]是以c为中心的最大回文串
- i、li以c为中心对称，m[i]是待求项
- m[li] == 1
- i + m[li] < r

■ 由于回文的对称性，得出结论

- m[i] = m[li]
- m[i] == 1

Manacher（马拉车）

			0		1		2		3		4		5		
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
cs	^	#	a	#	b	#	a	#	b	#	a	#	b	#	$
m	0	0	1	0	3	0	5	0							
		↑ l			↑ li		↑ c		↑ i			↑ r			

- 已知
  - m[li] == 3
  - i + m[li] == r

- 结论
  - m[i]至少是m[li]，也就是说，至少是3
  - 接下来利用扩展中心法以i为中心计算出m[i]

- 当i + m[i] > r时，更新c、r
  - c = i
  - r = i + m[i]

Manacher（马拉车）

			0		1		2		3		4		5		6		7		8		
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
cs	^	#	c	#	b	#	a	#	b	#	c	#	b	#	a	#	b	#	b	#	$
m	0	0	1	0	1	0	5	0	1	0	7	0	1	0							
				l			li				c				i			r			

- 已知
  - m[li] == 5
  - i + m[li] > r
- 结论
  - m[i]至少是r – i，也就是说，至少是3
  - 接下来利用扩展中心法以i为中心计算出m[i]
- 当i + m[i] > r时，更新c、r
  - c = i
  - r = i + m[i]

Manacher（马拉车）

			0		1		2		3		4		5		
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
cs	^	#	c	#	a	#	b	#	a	#	a	#	a	#	$
m	0	0	1	0	1	0	3	0	1						

l　　c　　r

i

- 当i == r时
  - 直接利用扩展中心法以i为中心计算出m[i]
- 当i + m[i] > r时，更新c、r
  - c = i
  - r = i + m[i]