#### 本节内容

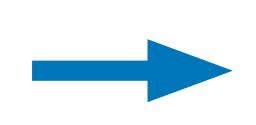
# KMP算法

求next数组



#### KMP算法

#### 根据模式串T,求 出 next 数组



#### 利用next数组进行匹配 (主串指针不回溯)

T = 'abaabc'

#### next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
	0	10	1	2	2	3



其中,求 next 数组时间复杂度 O(m) 模式匹配过程最坏时间复杂度 O(n)

```
int Index_KMP(SString S,SString T,int next[]){
   int i=1, j=1;
   while(i<=S.length&&j<=T.length){</pre>
       if(j==0||S.ch[i]==T.ch[j]){
           ++i;
                                 /继续比较后继字符
           ++j;
       else
           j=next[j];
                                //模式串向右移动
   if(j>T.length)
       return i-T.length;
                                //匹配成功
   else
       return 0;
```

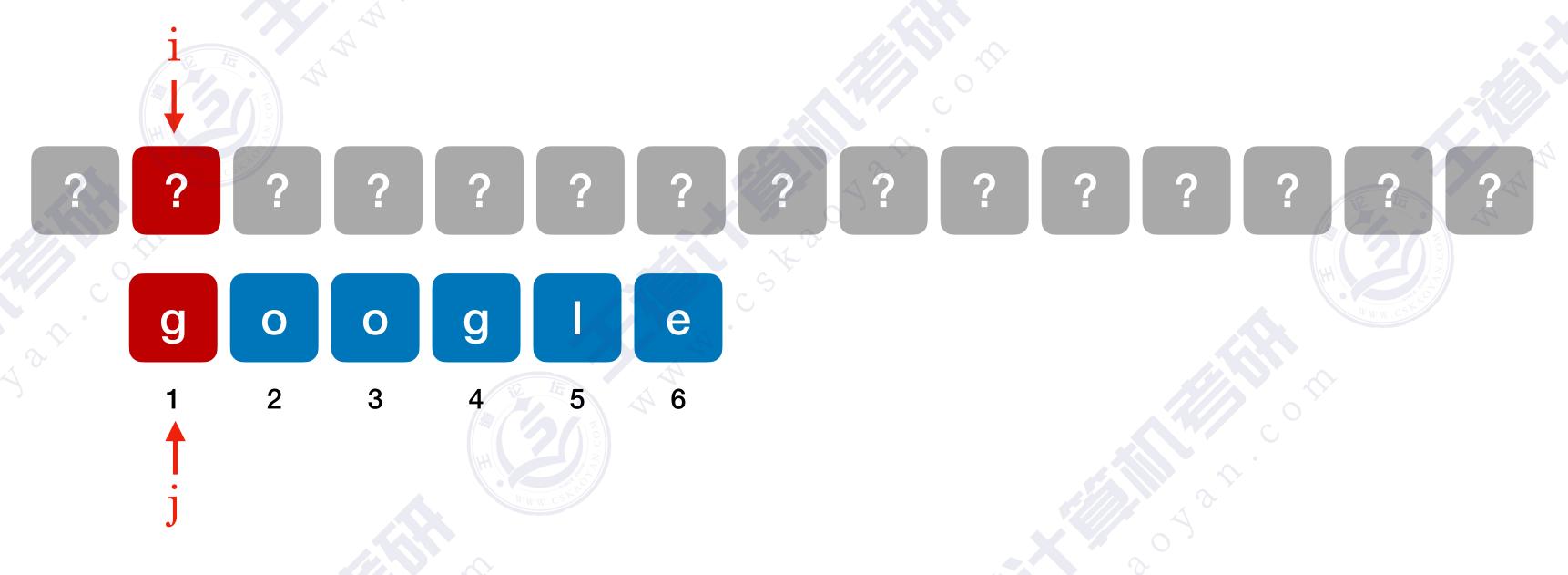
next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配



next数组:

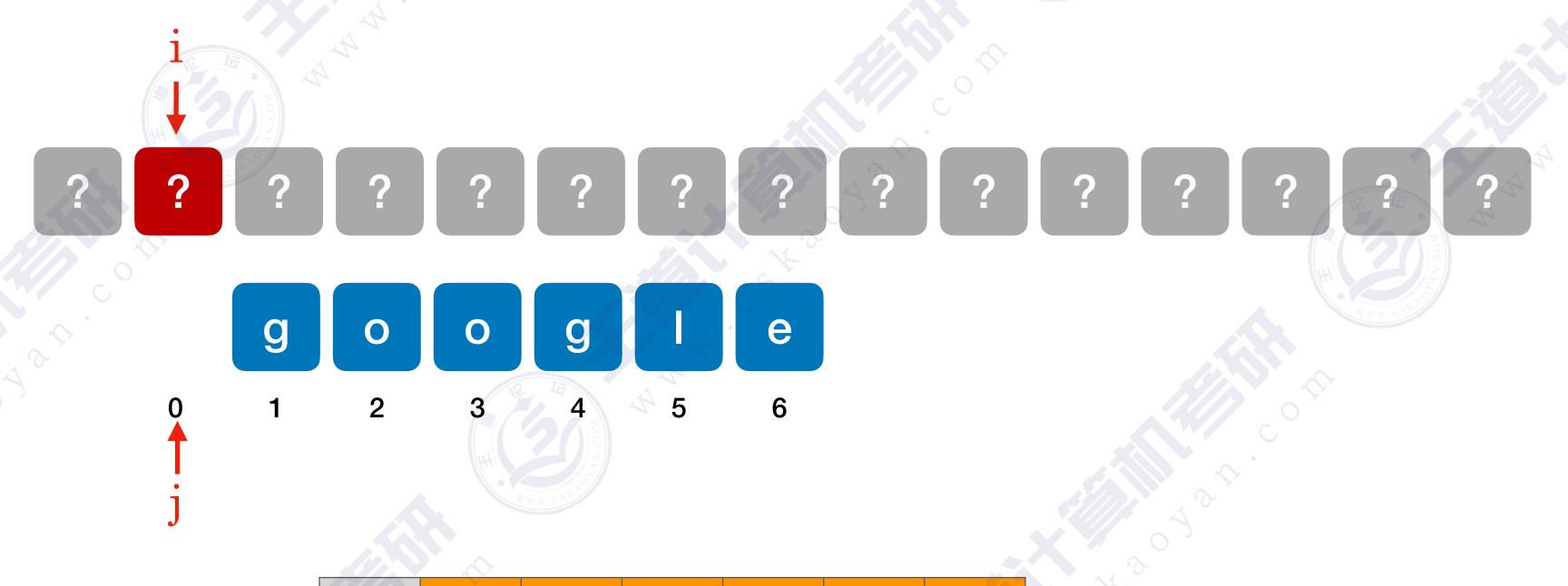
next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
40						

next数组的作用: 当模式串的第 j 个字符失配时,从模式串的第 next[j] 的继续往后匹配



next数组:	next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
	400					W III	

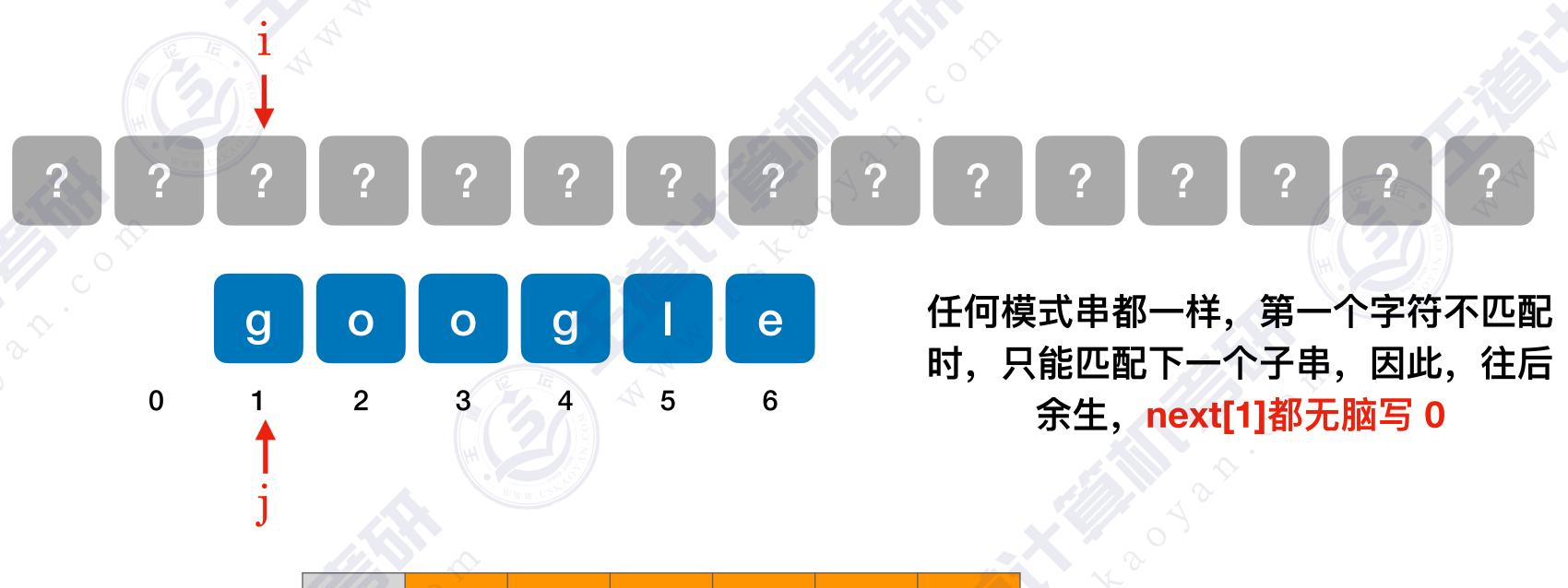
next数组的作用: 当模式串的第 j 个字符失配时,从模式串的第 next[j] 的继续往后匹配



next数组:

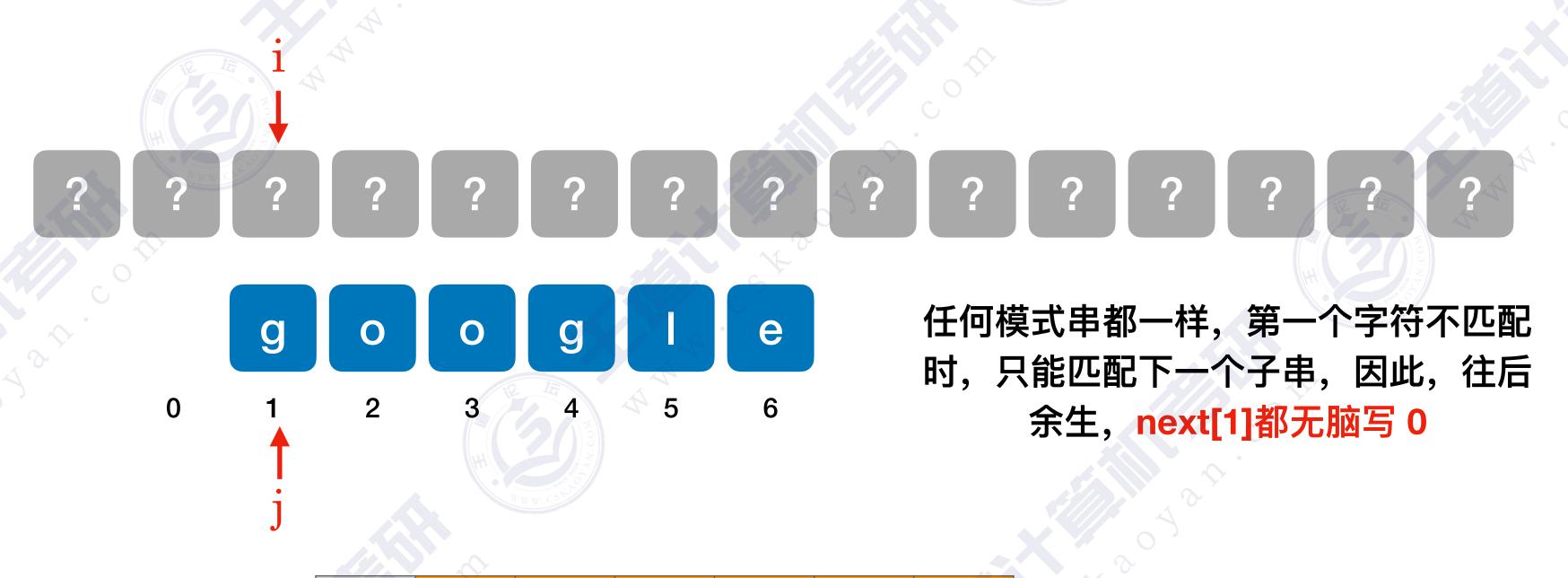
next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
3					W III	437
				//100		

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配



next貨组: next[0] next[1] next[2] next[3] next[4] next[5] next[6]

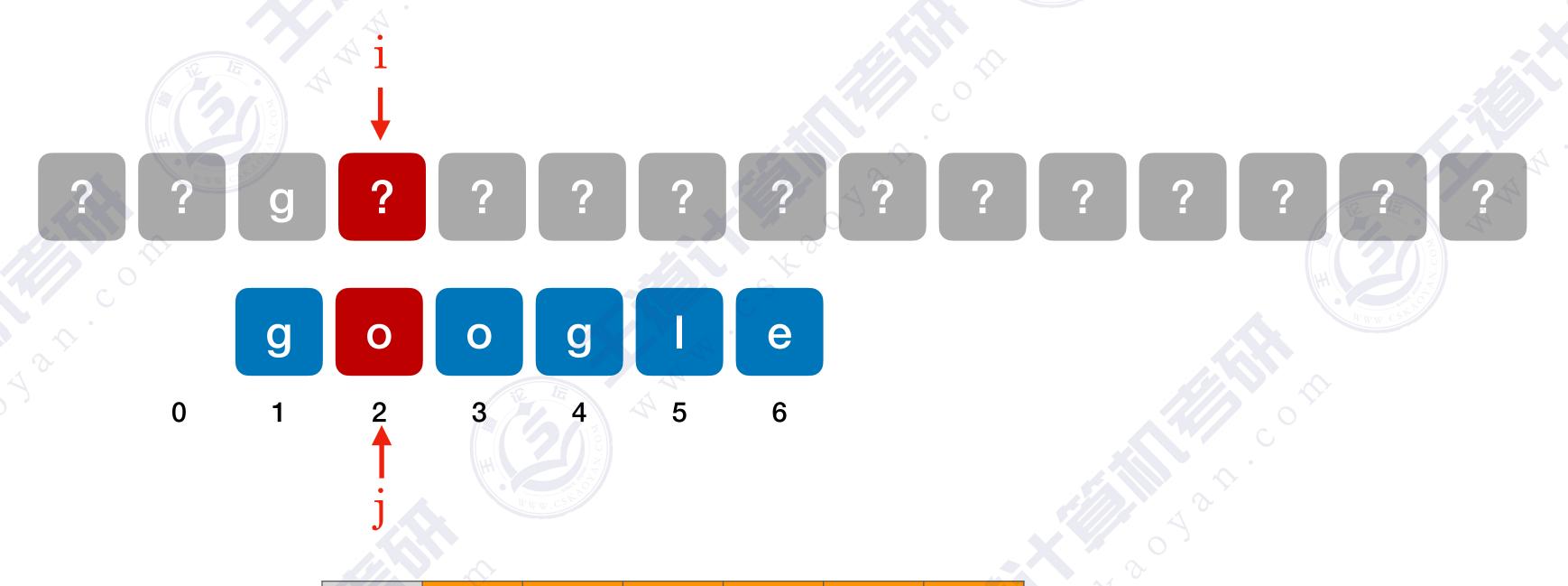
next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配



next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
727	0				14 J	

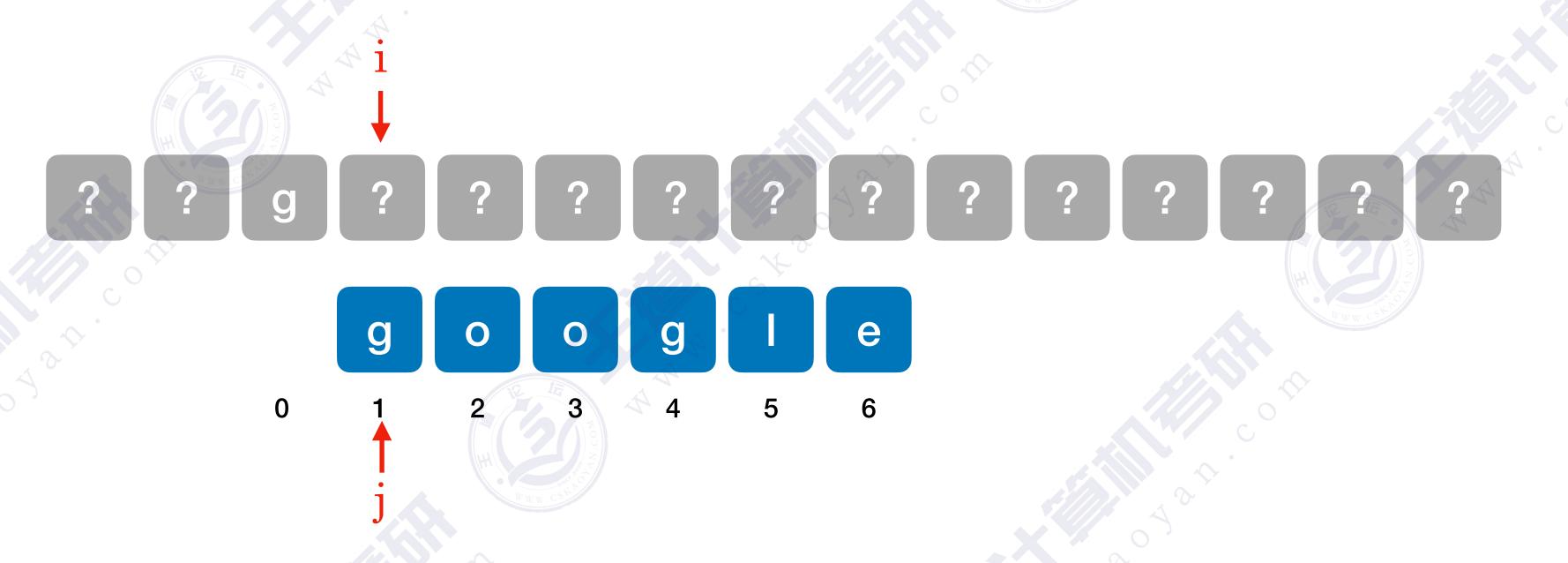
next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配



next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
701	0				II III	

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

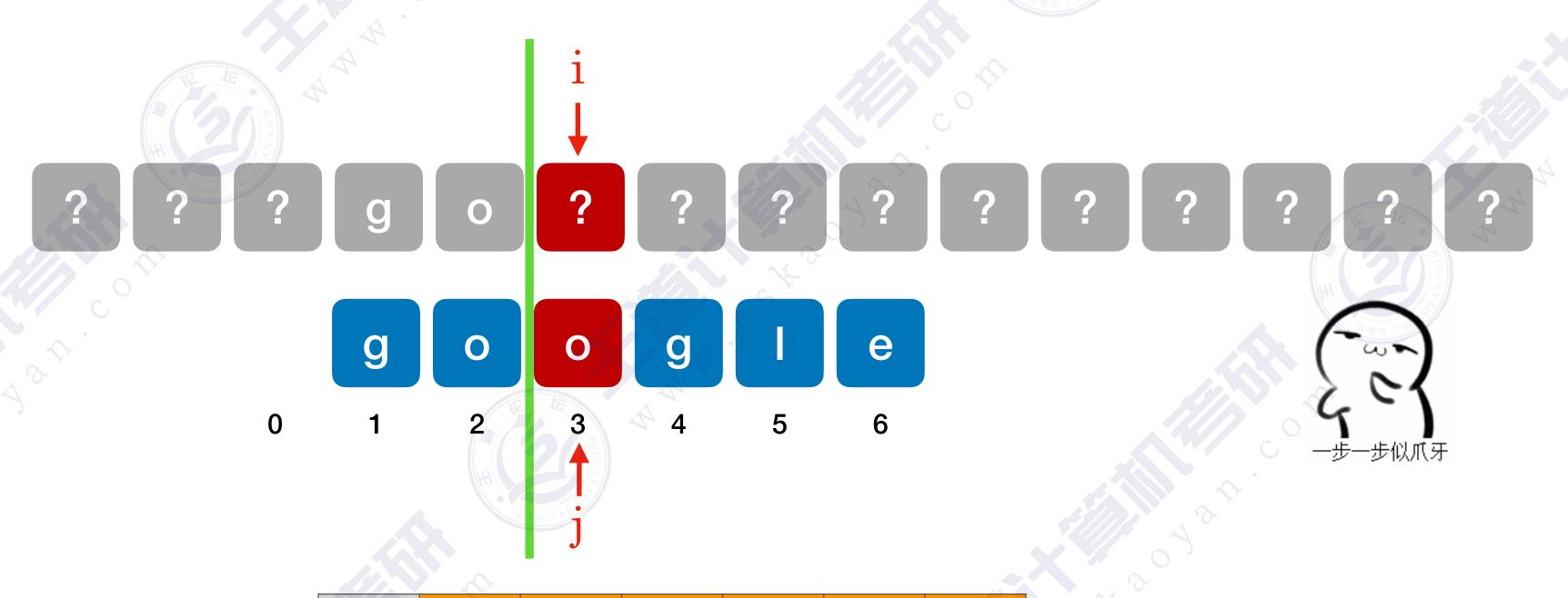


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
727	0	1			12 II	

任何模式串都一样,第2个字符不匹配时,应尝试匹配模式串的第1个字符,因此,往后余生,next[2]都无脑写 1

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

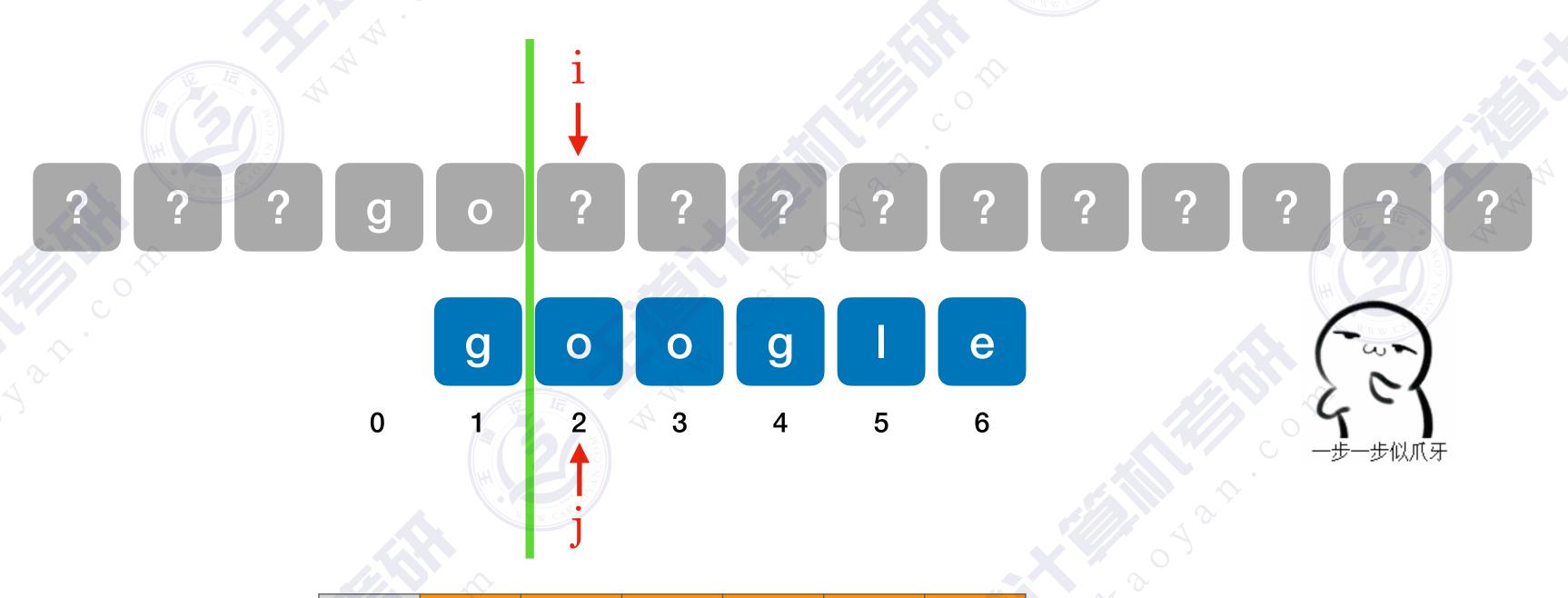


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
4	0	1			15 <u>111</u>	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

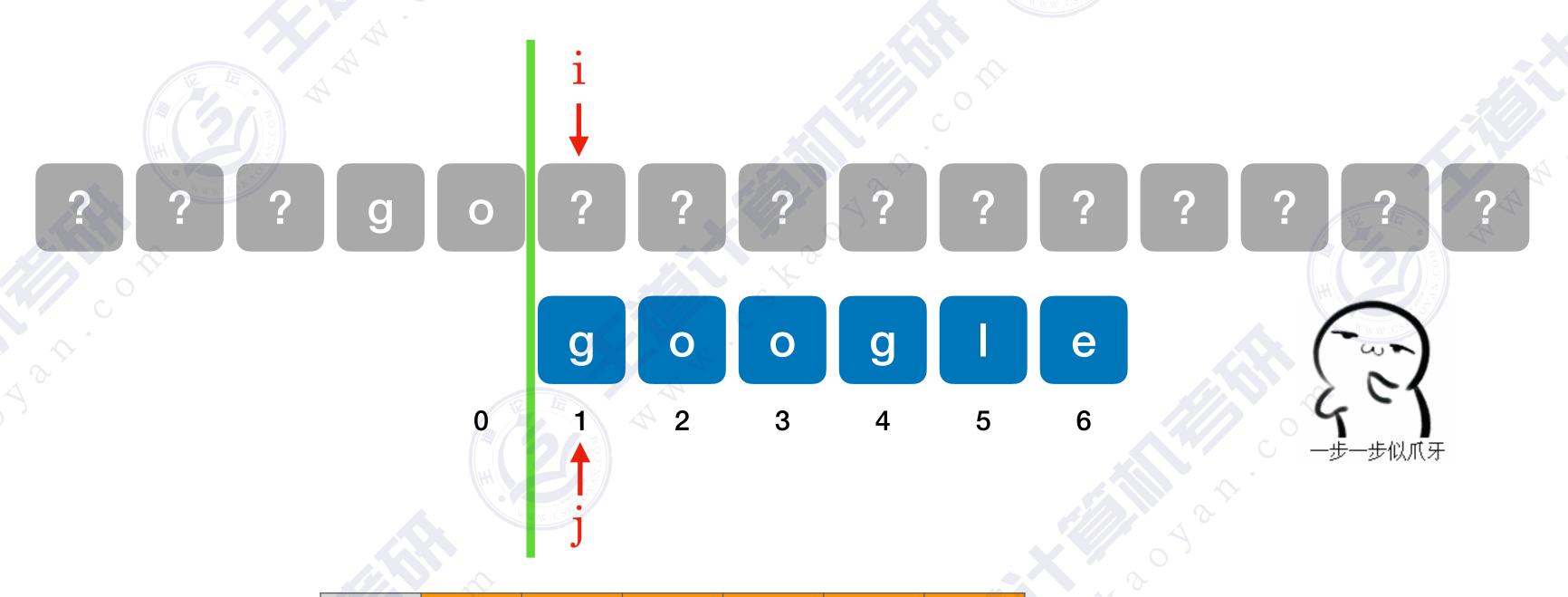


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
701	0	1			1 <u>11</u>	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

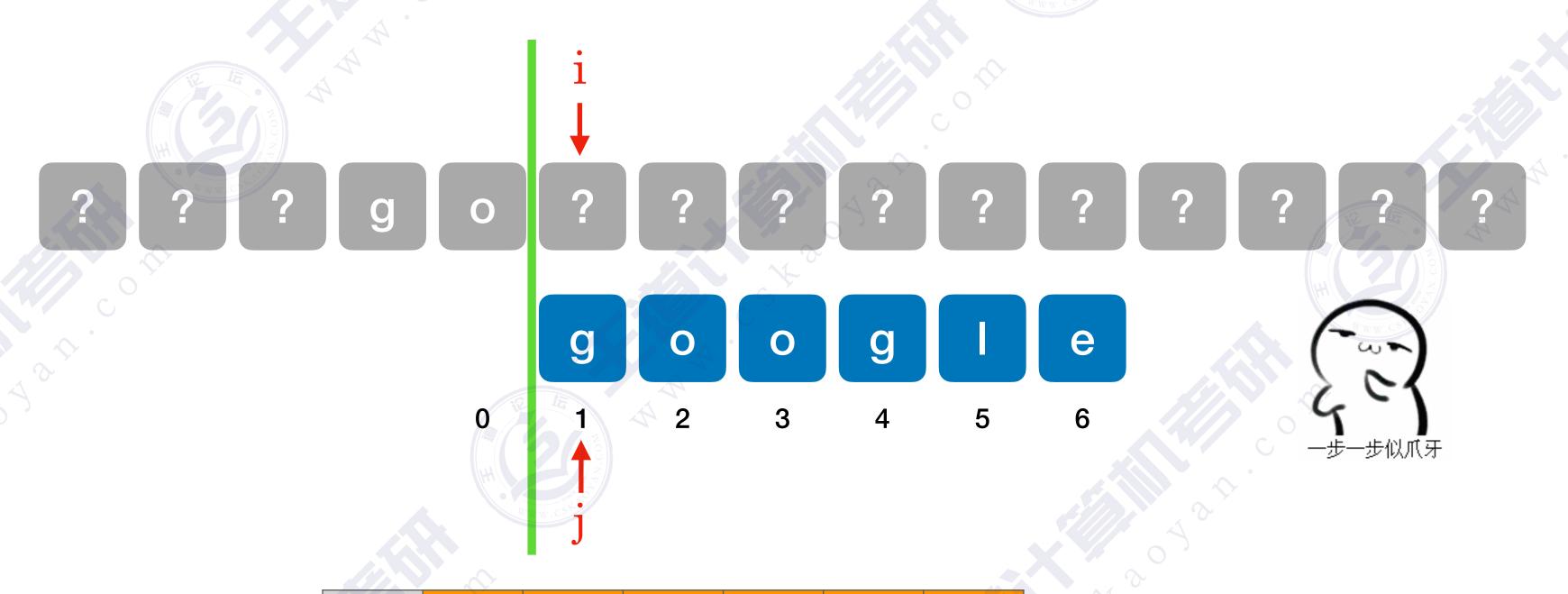


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
707	0	1			12 II	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

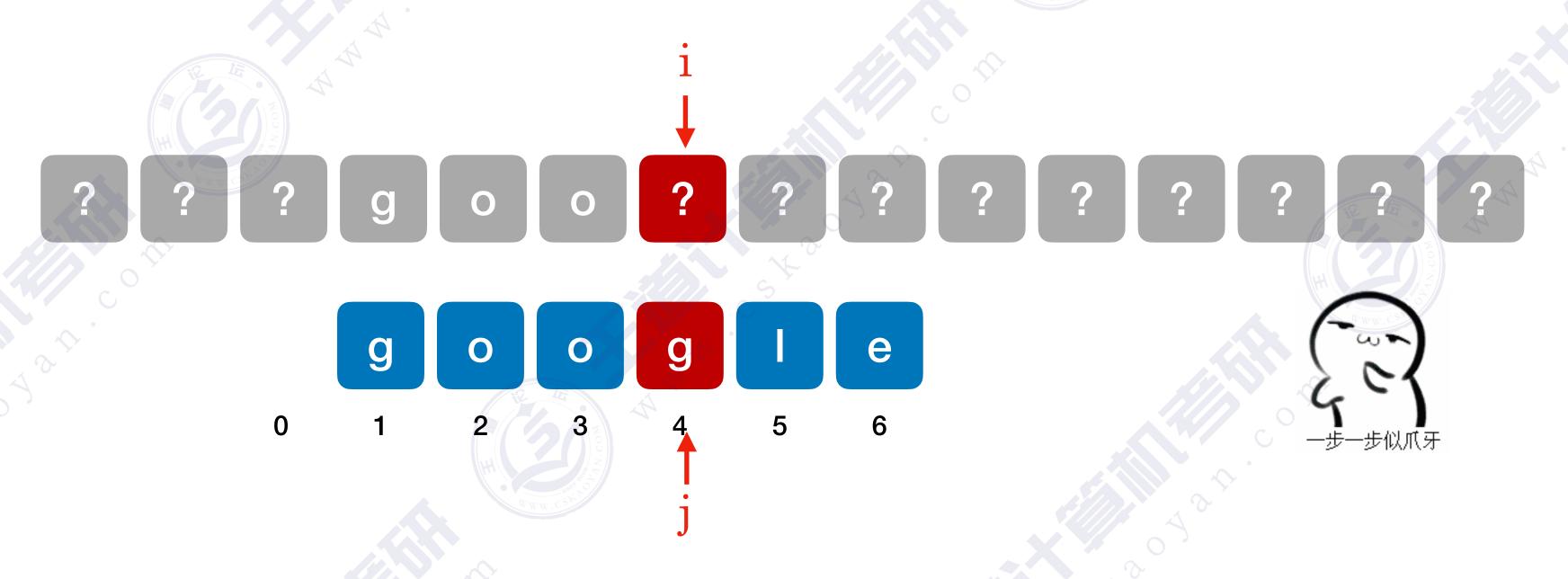


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
	0	1	1		10 III	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

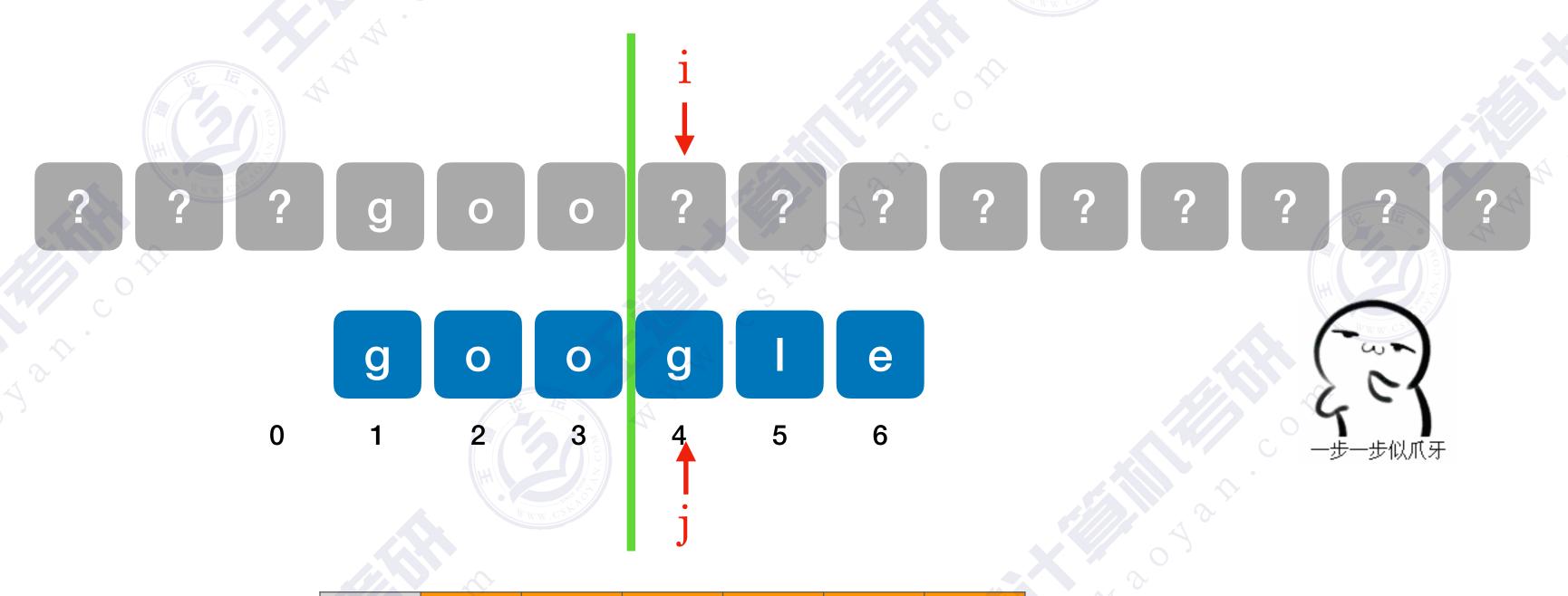


next数组:

next[0	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1		1 <u>11</u>	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

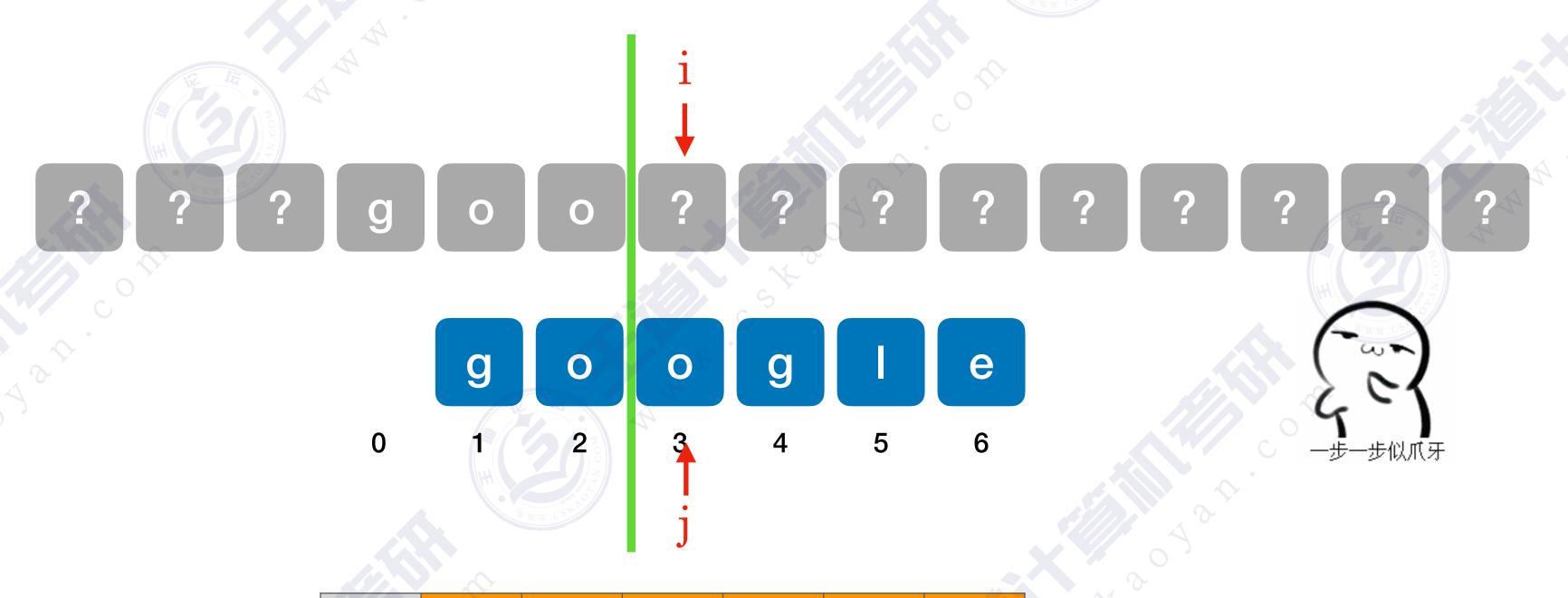


next数组:

next[0	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1		1 <u>11</u>	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

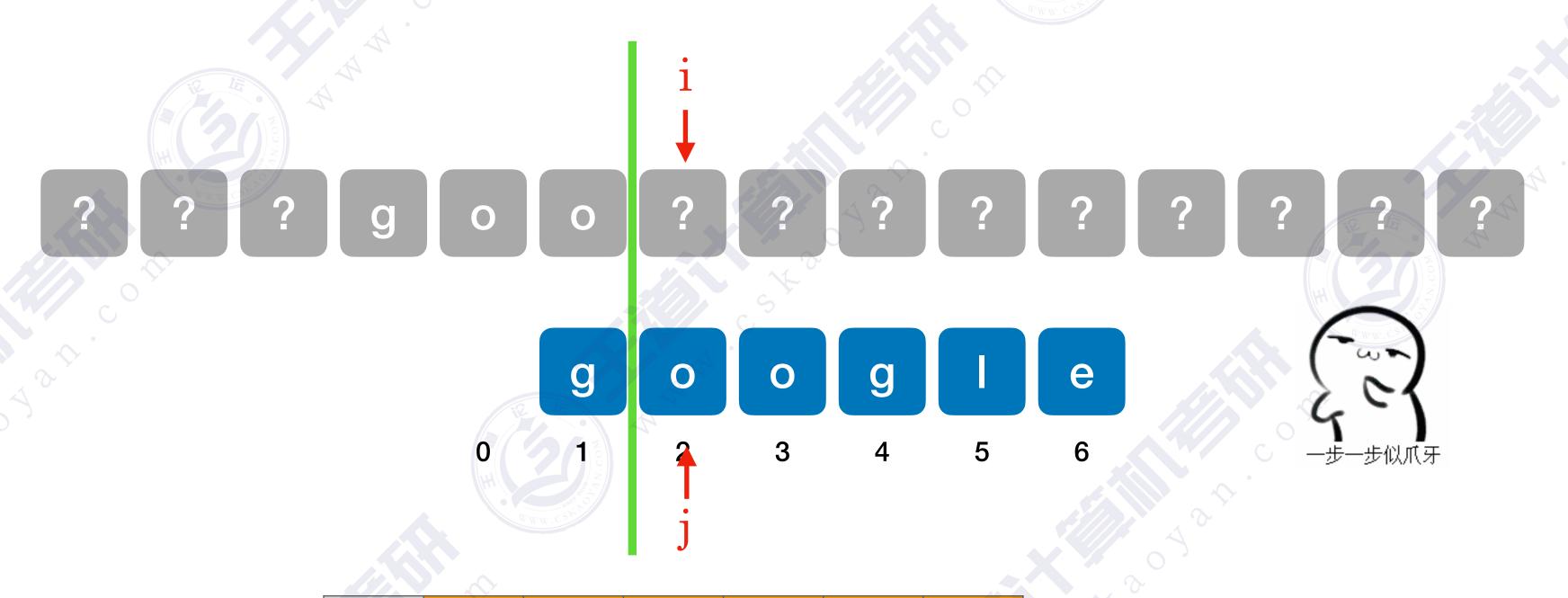


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1		1 <u>11</u>	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

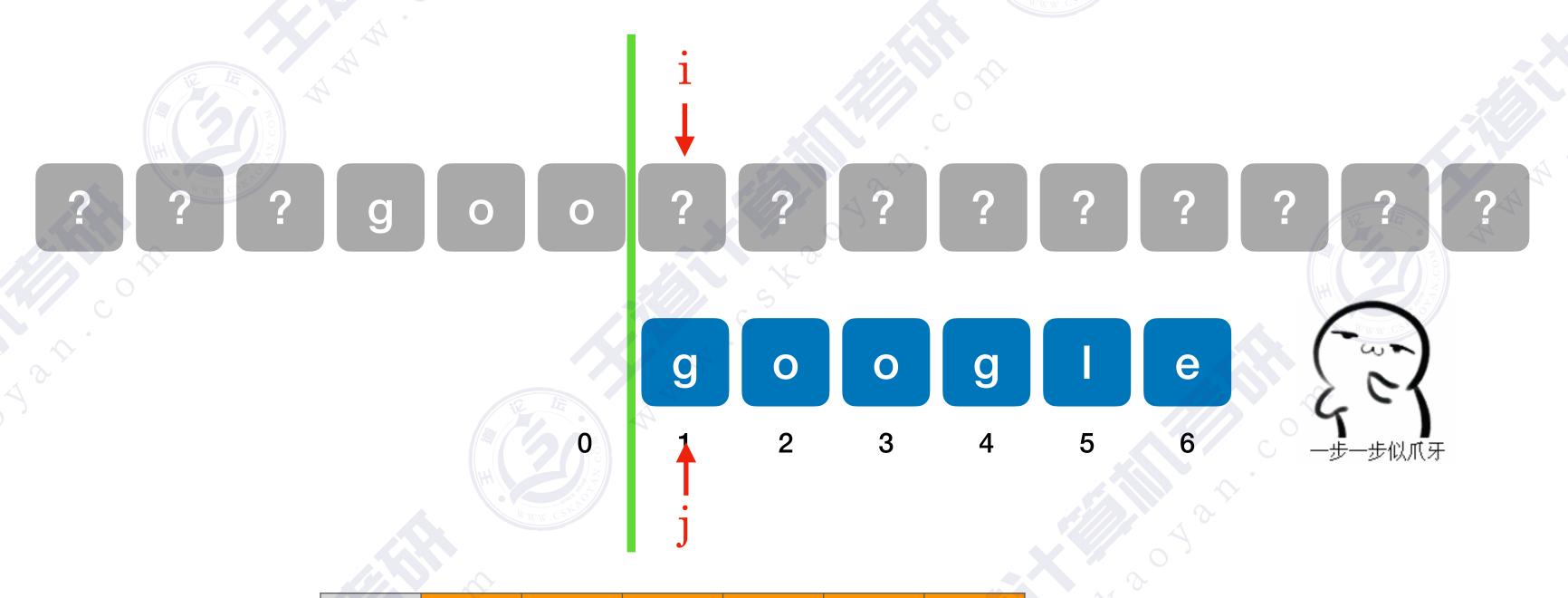


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1			

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

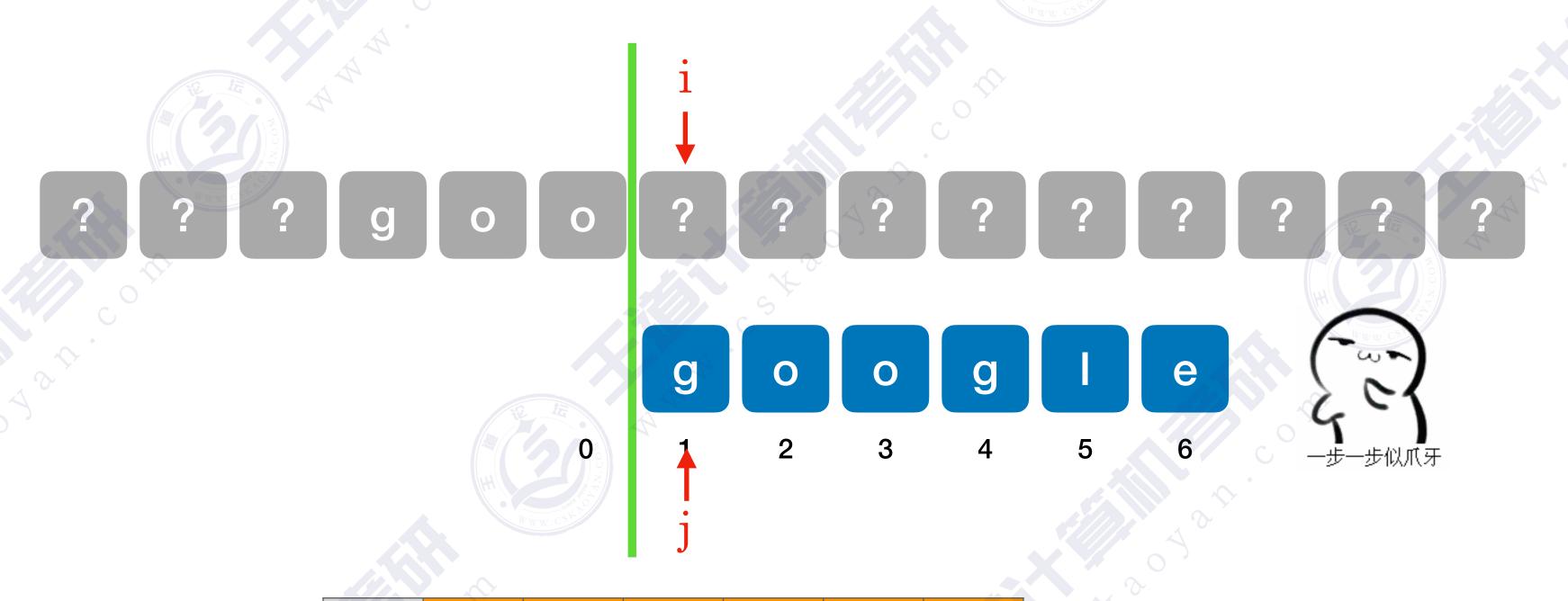


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1		10 II	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

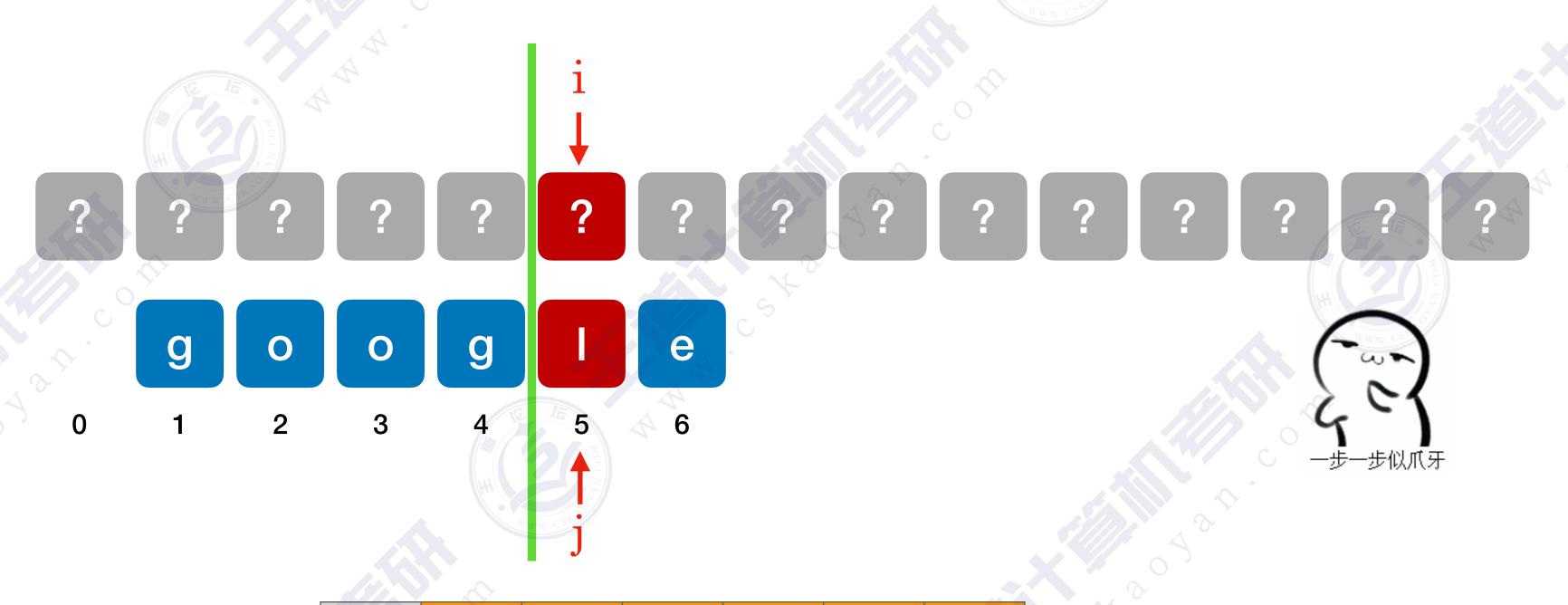


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
40	0	1	1	1	15 III	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

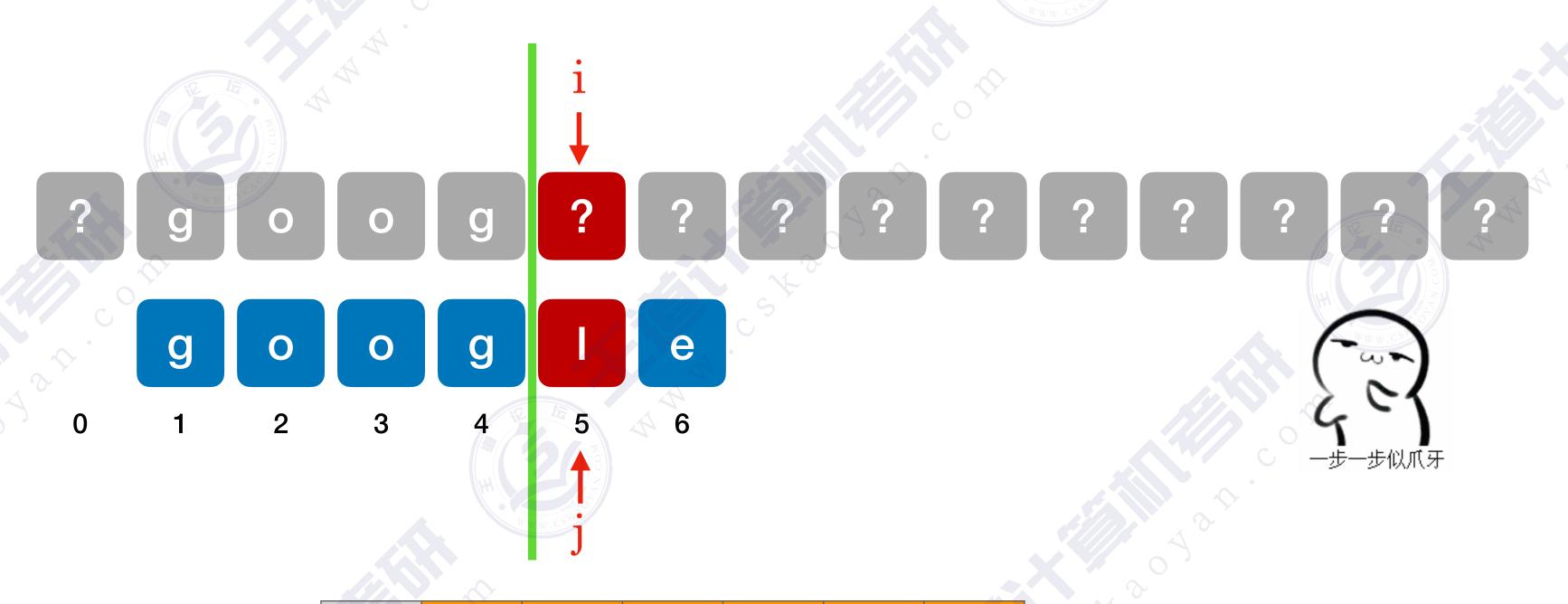


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
404	0	1	1	1	EL SE	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

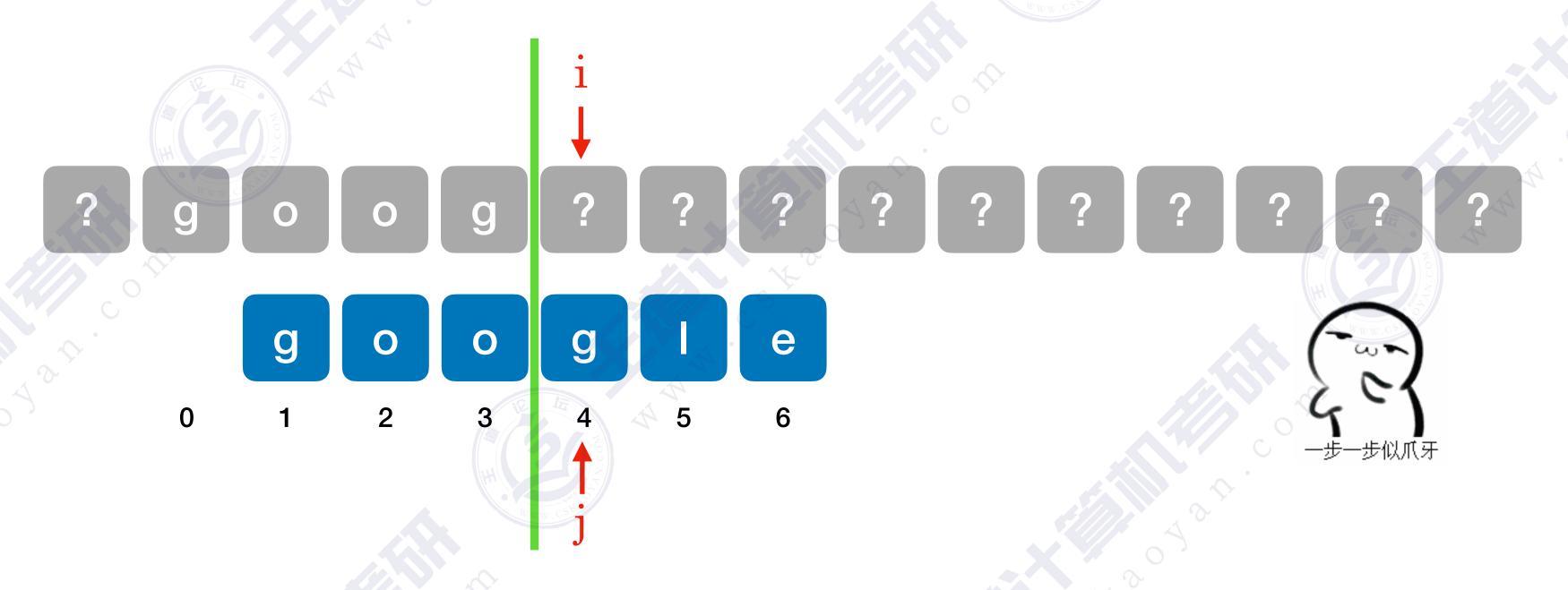


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	1 <u>11</u>	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

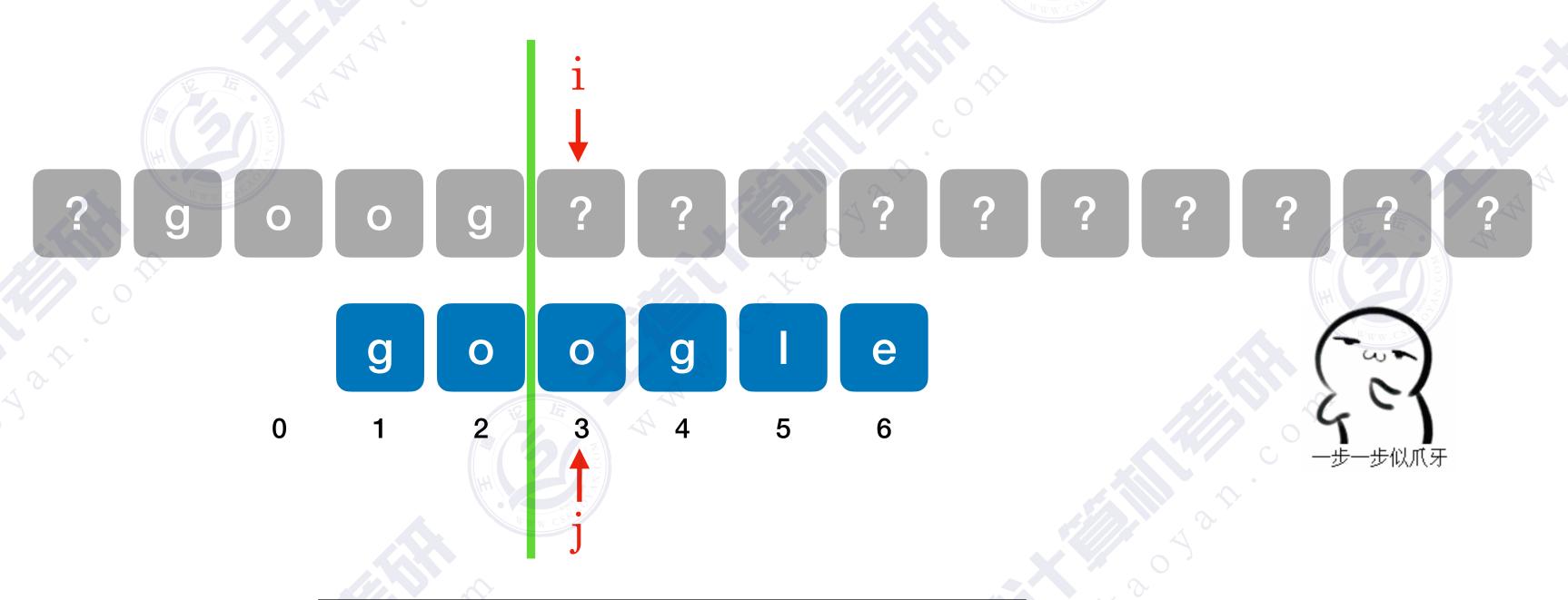


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
4	0	1	1	1	1 <u>11</u>	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

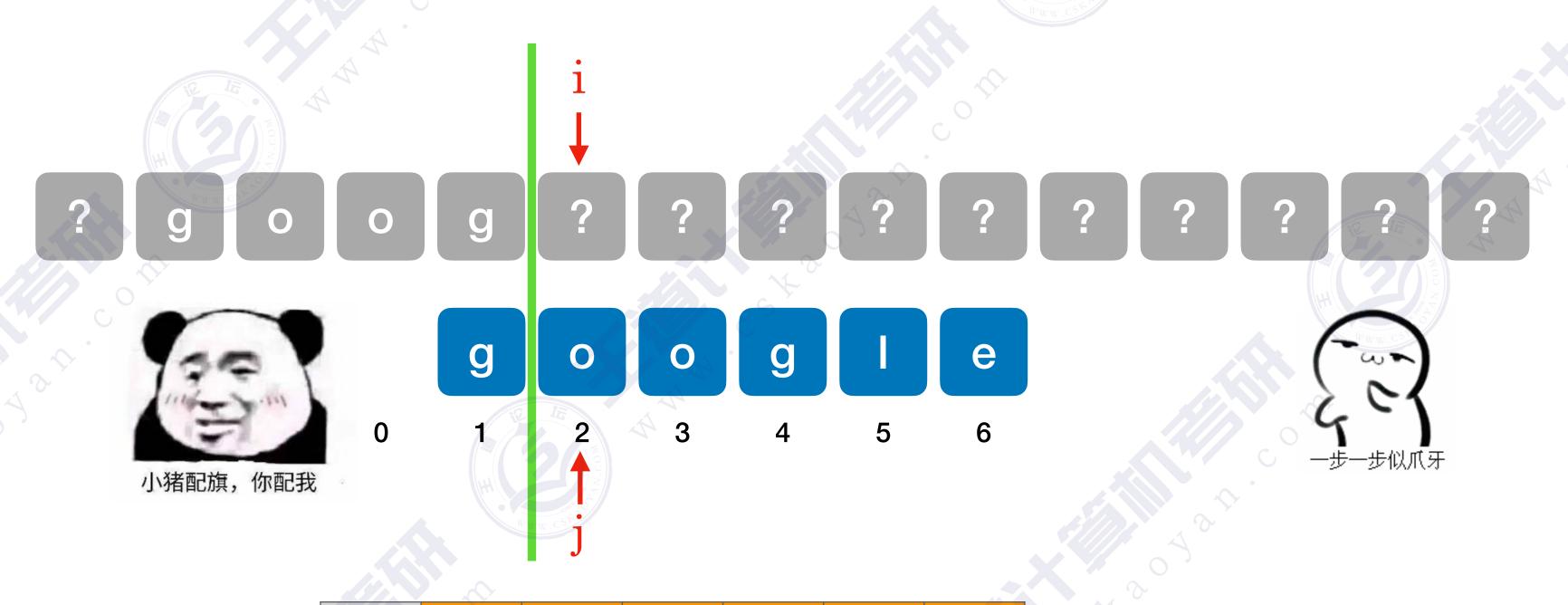


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
404	0	1	1	1	EL SE	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

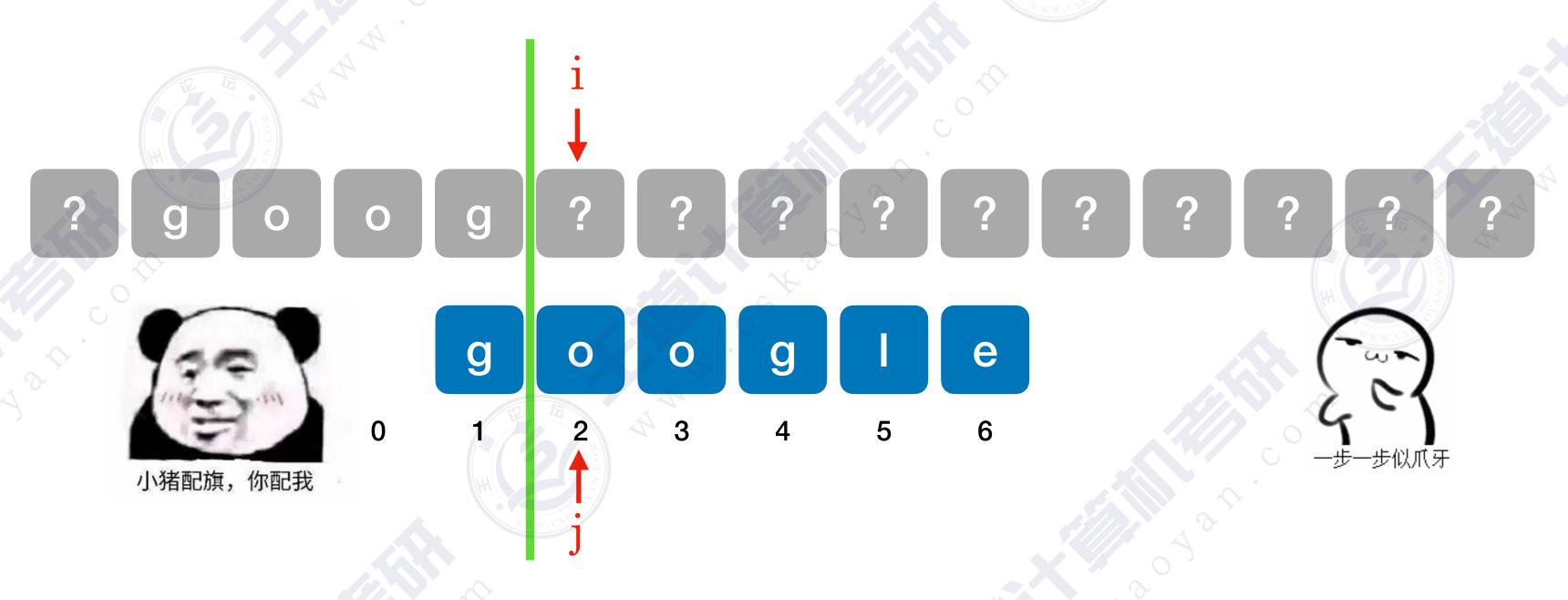


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
404	0	1	1	1	EL SE	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

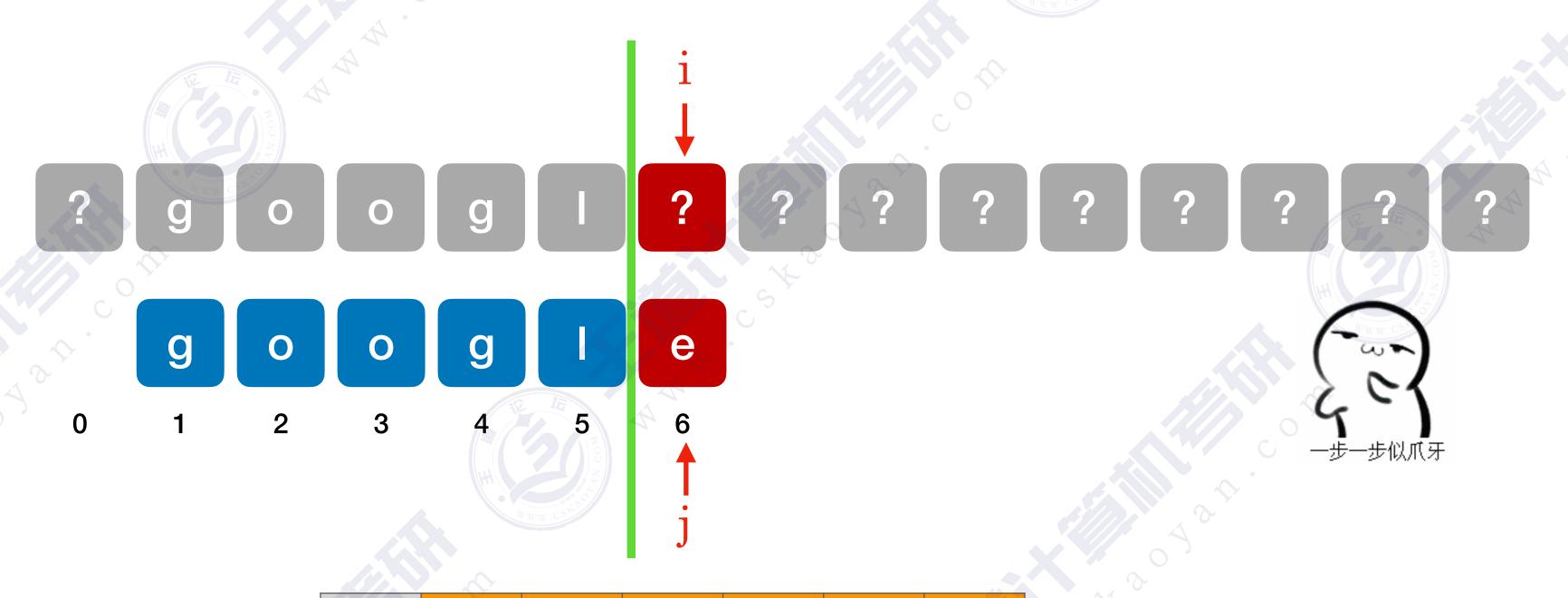


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

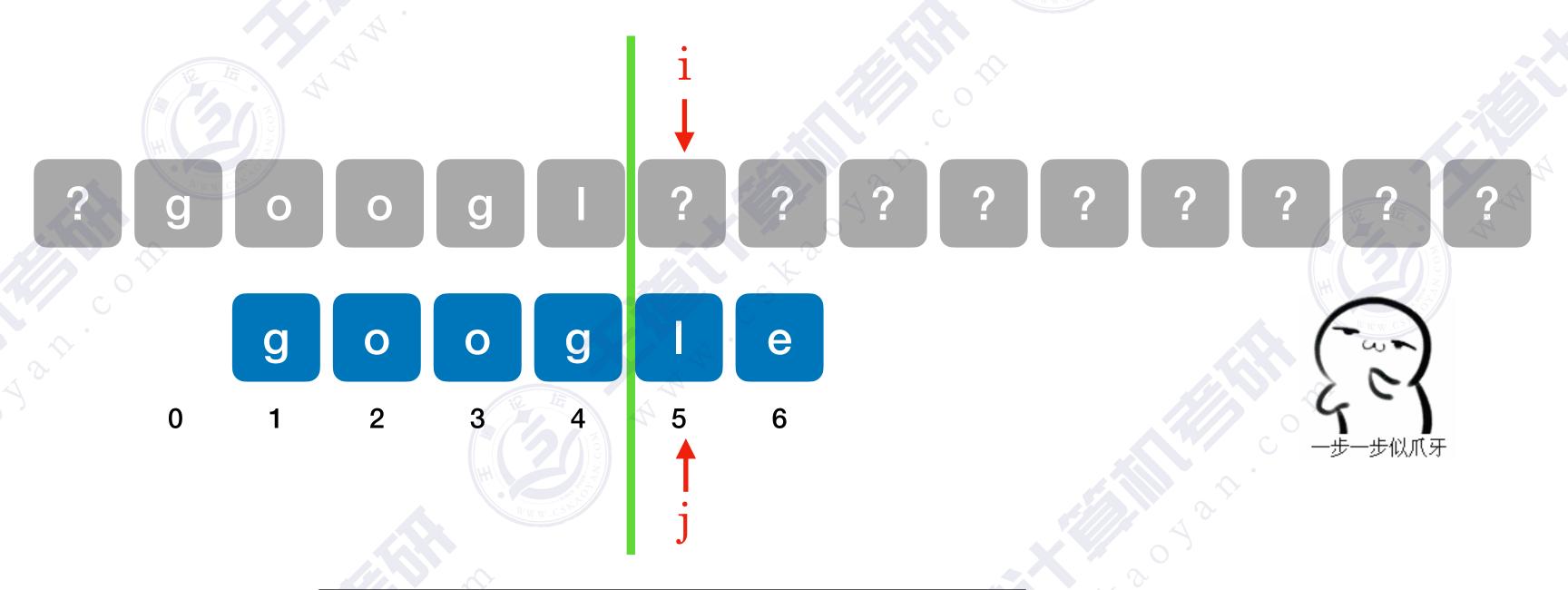


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

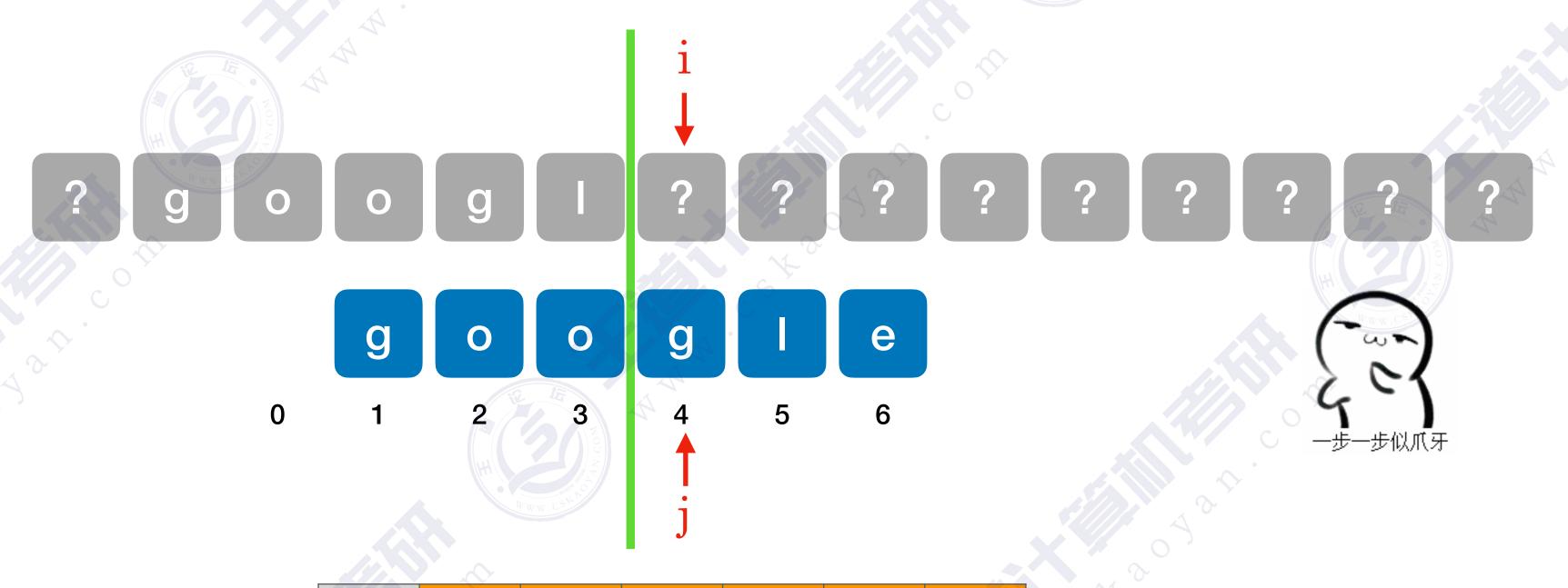


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

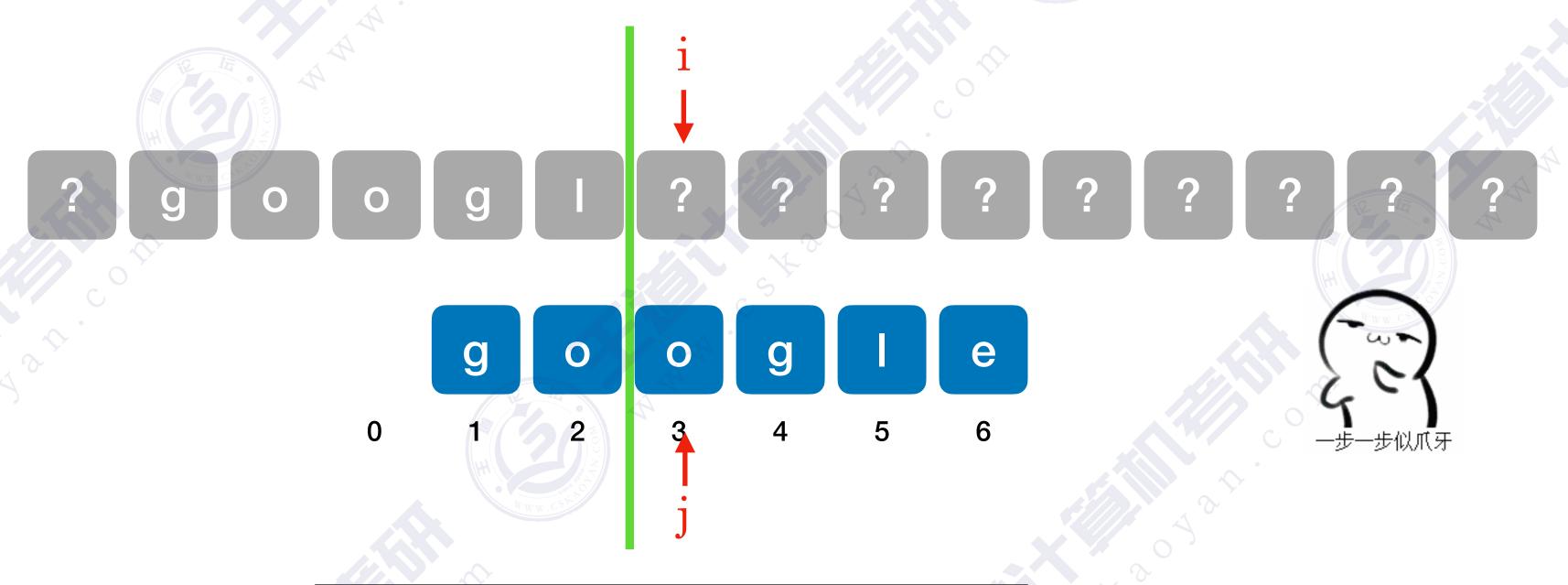


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

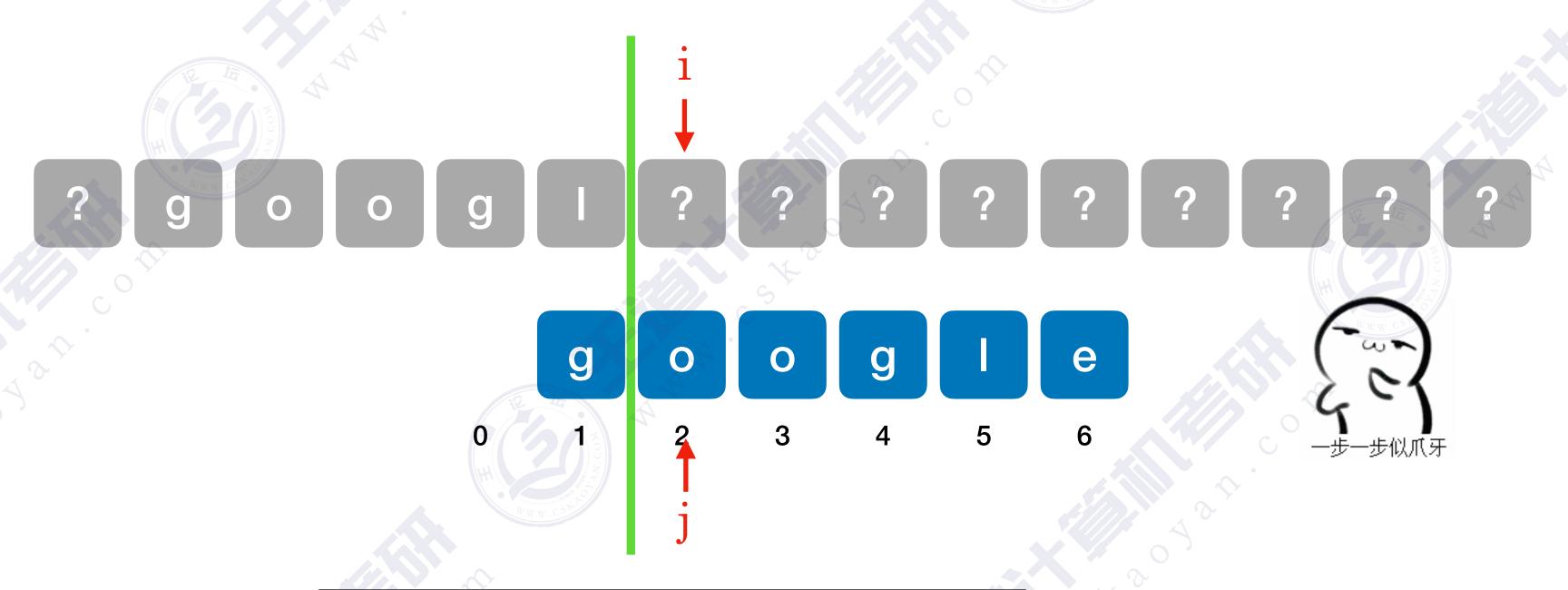


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
404	0	1	1	1	2	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

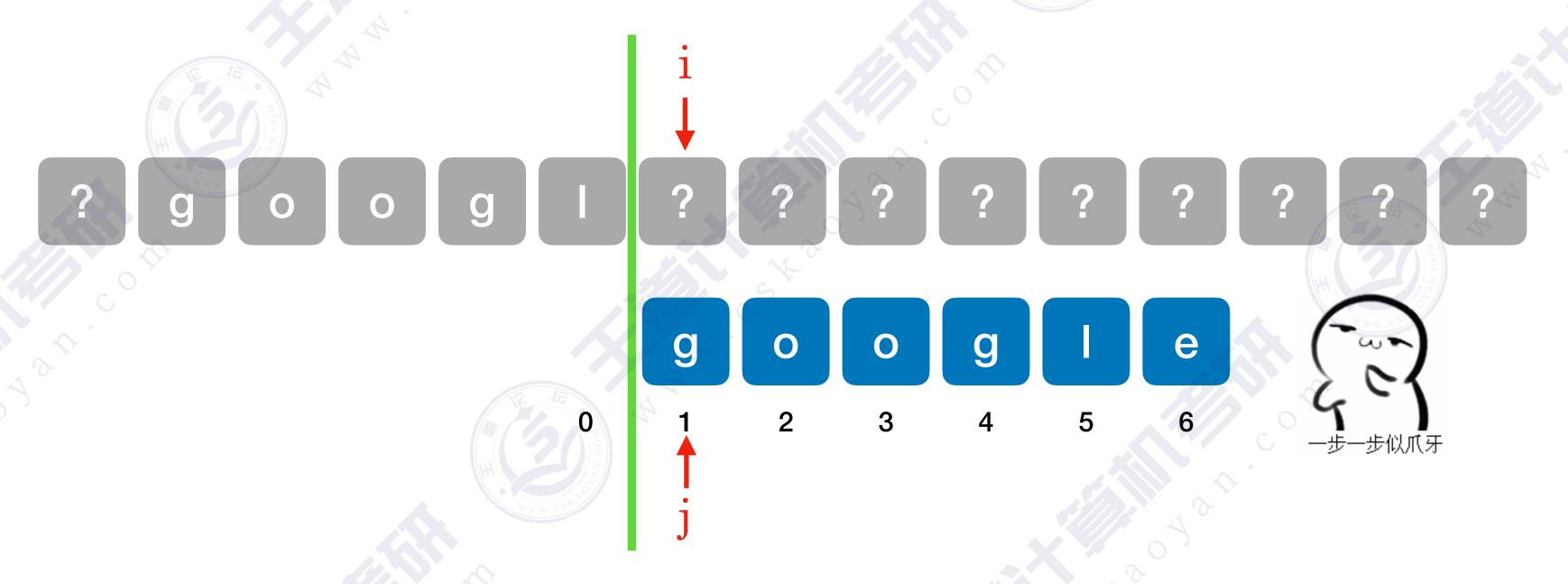


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
404	0	1	1	1	2	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配

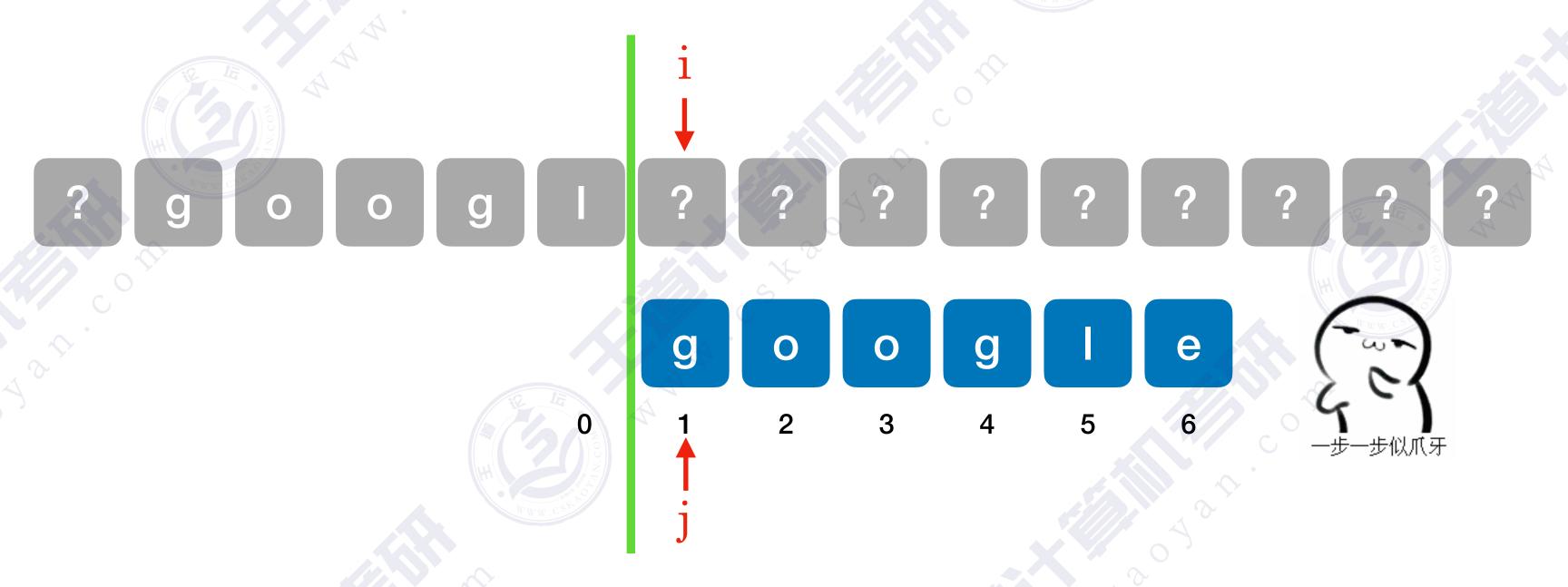


next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止

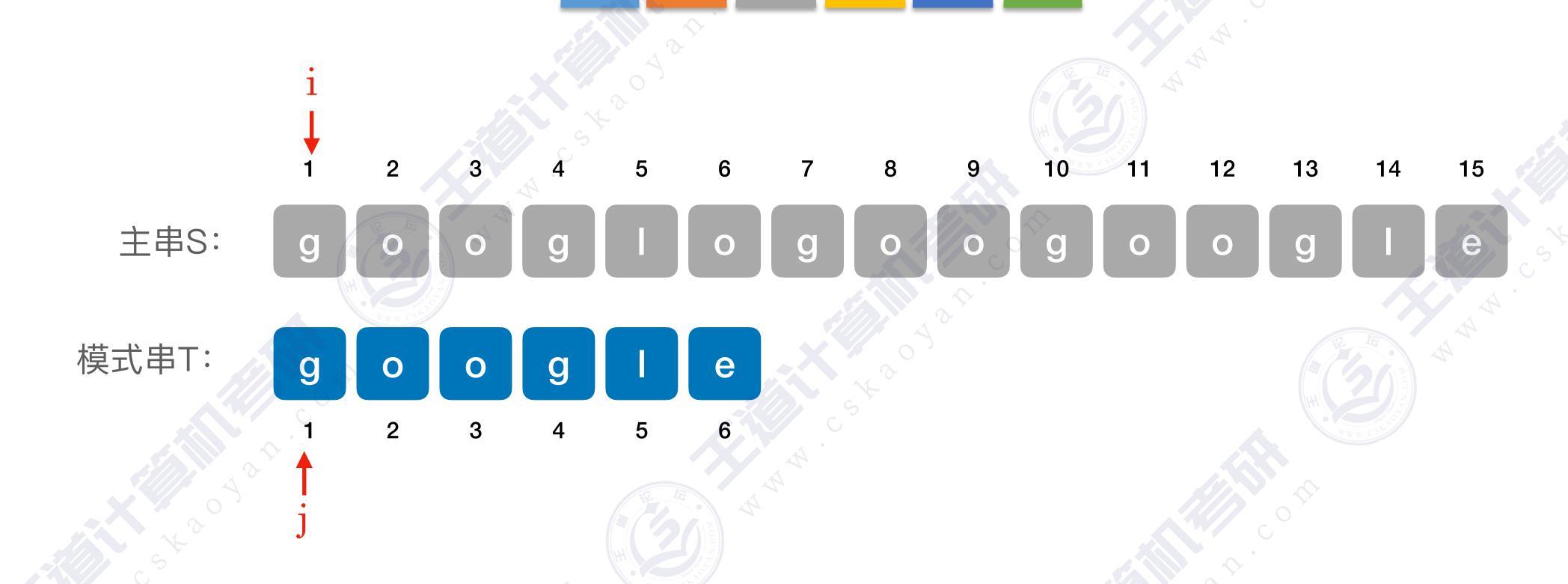
next数组的作用: 当模式串的第 j 个字符失配时, 从模式串的第 next[j] 的继续往后匹配



next数组:

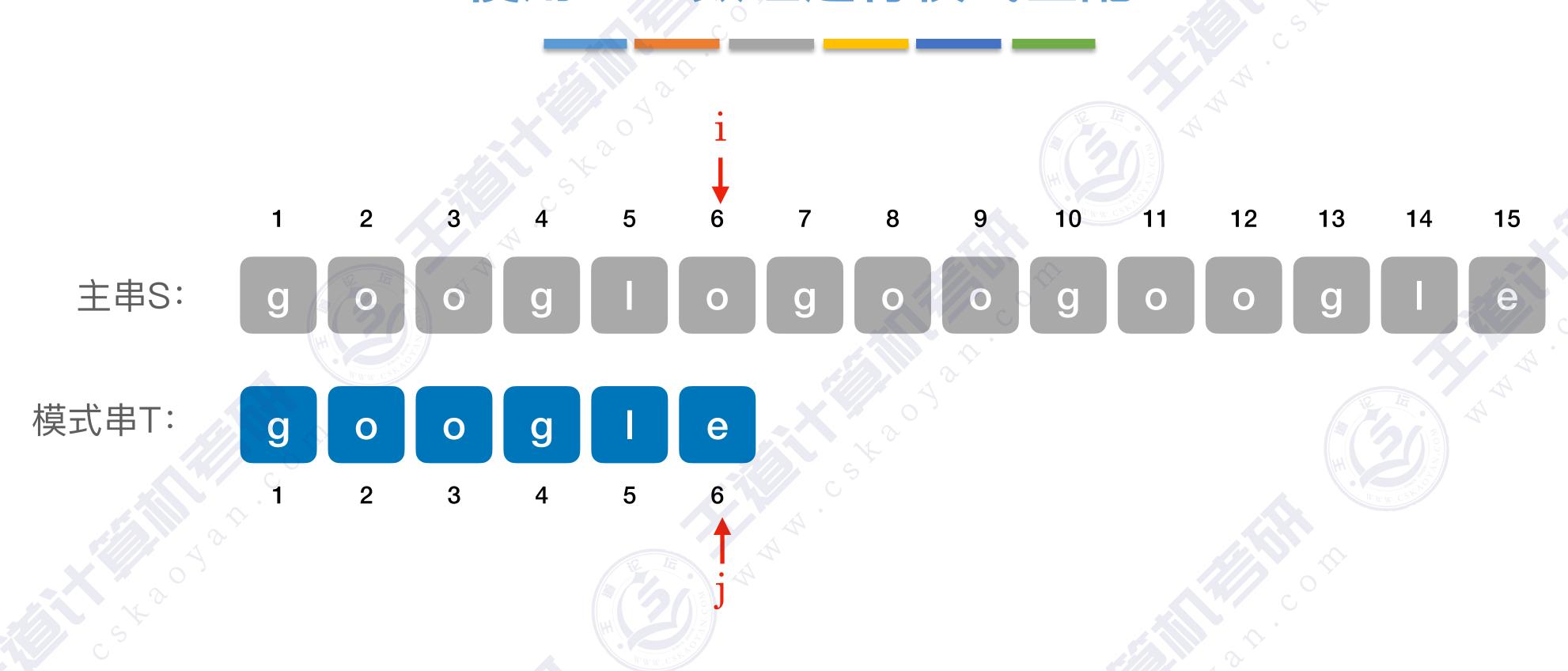
next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	1.6

在不匹配的位置前边,划一根美丽的分界线 模式串一步一步往后退,直到分界线之前 "能对上",或模式串完全跨过分界线为止



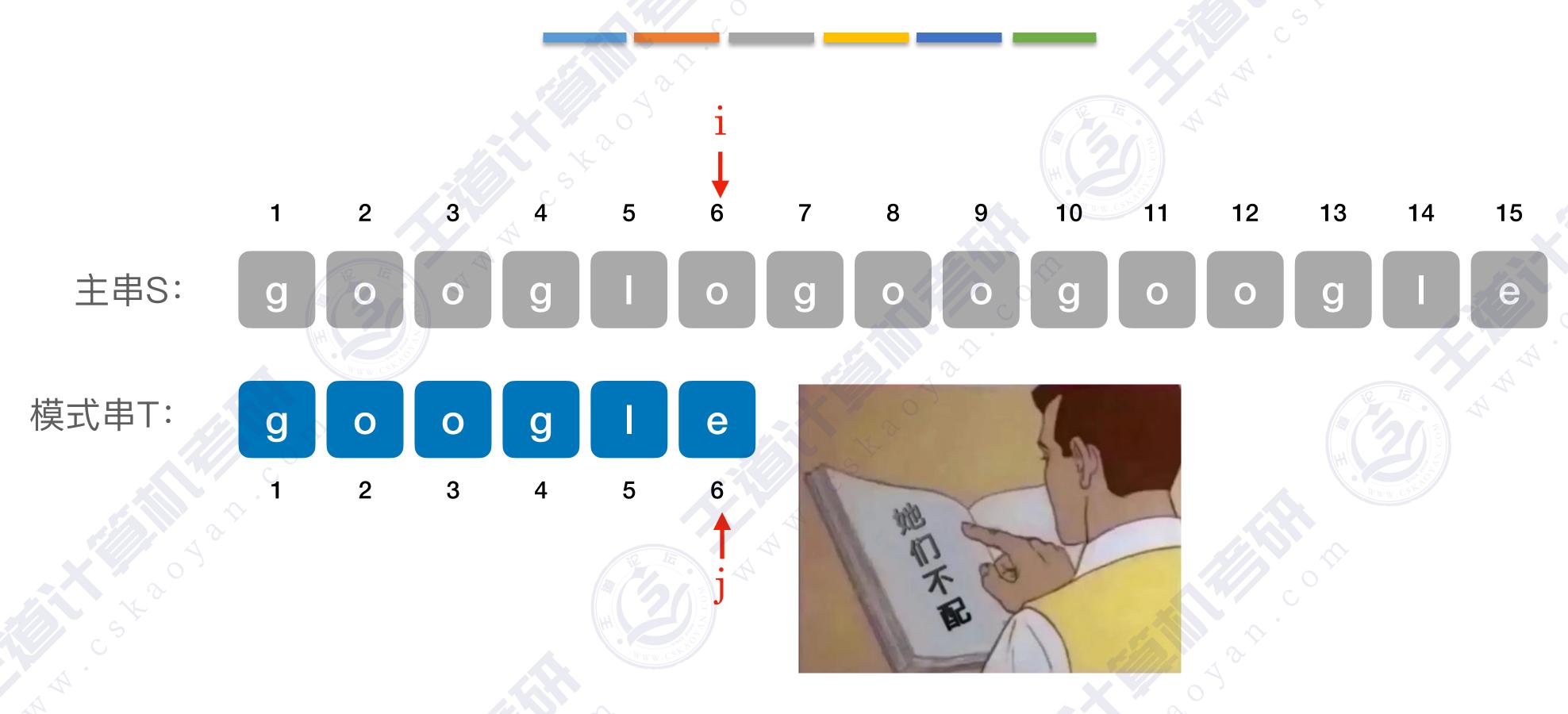
next数组:	next[0]

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
3	0	1	1	1	2	1.4



		next数组

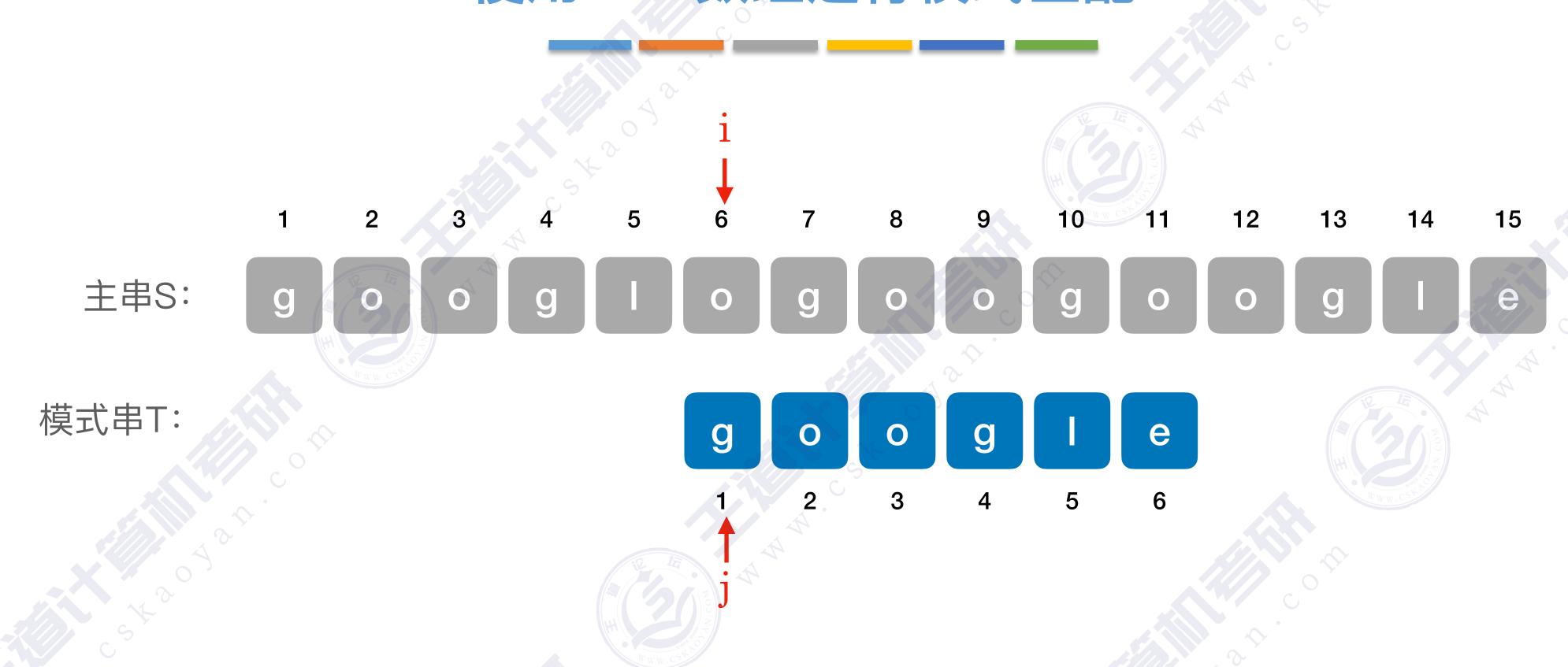
next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	1.6





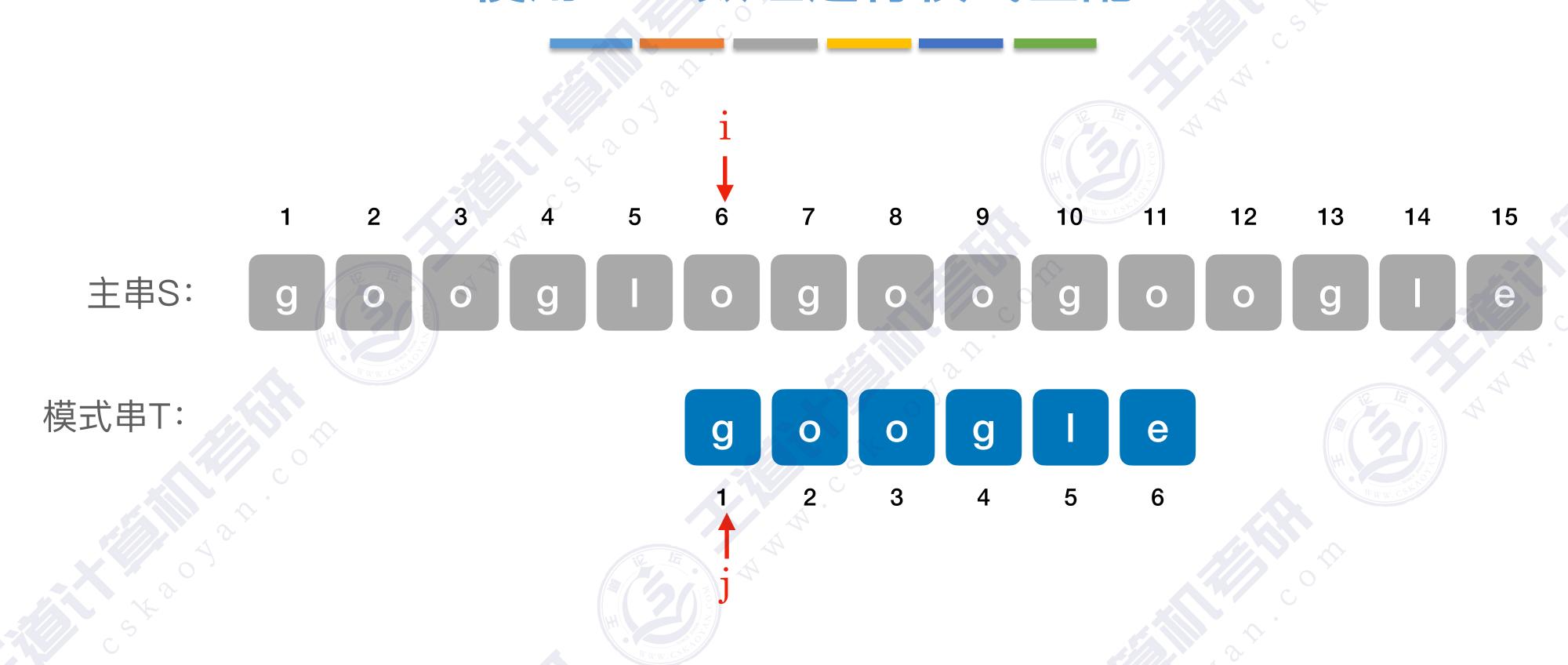
组:	next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
	3	0	1	1	1	2	14

j=next[j]



	next数组

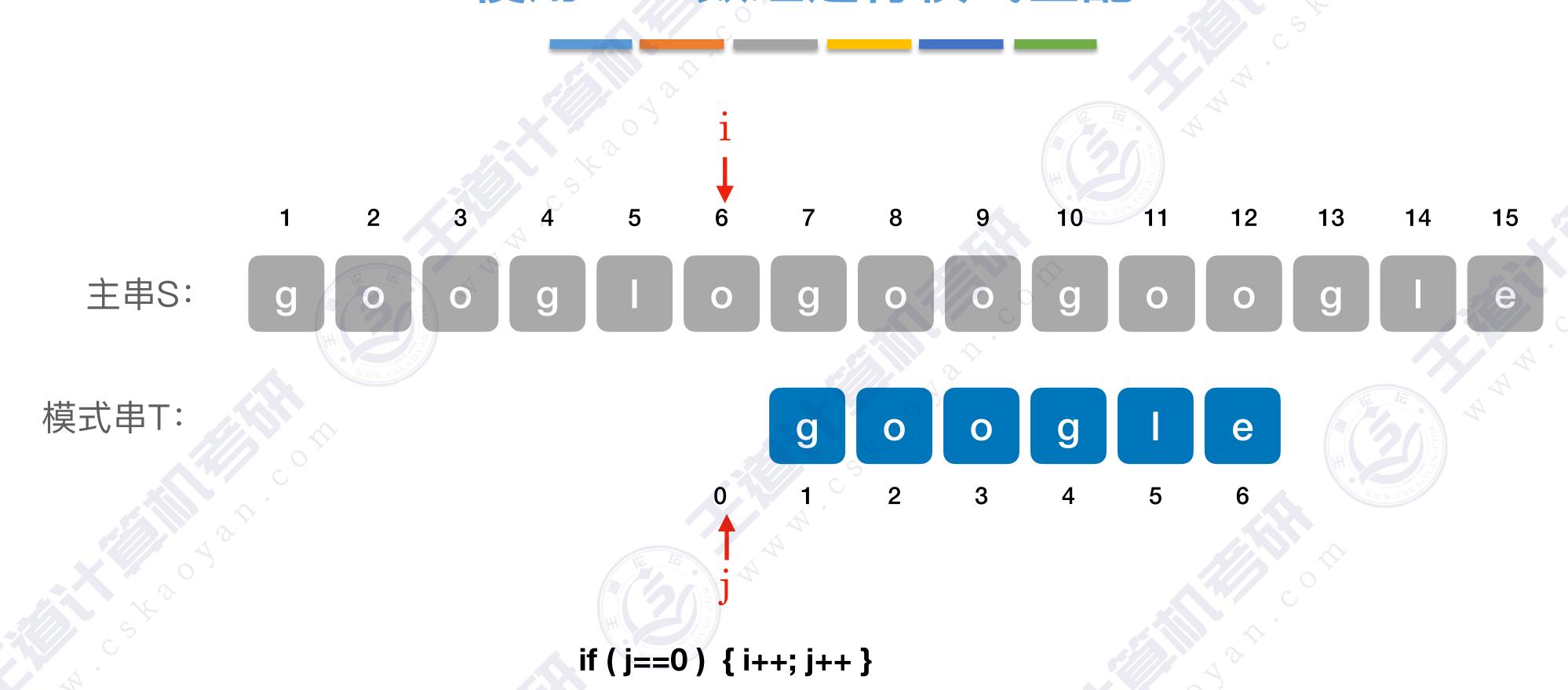
next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
3	0	1	1	1	2	1.6



next数组:

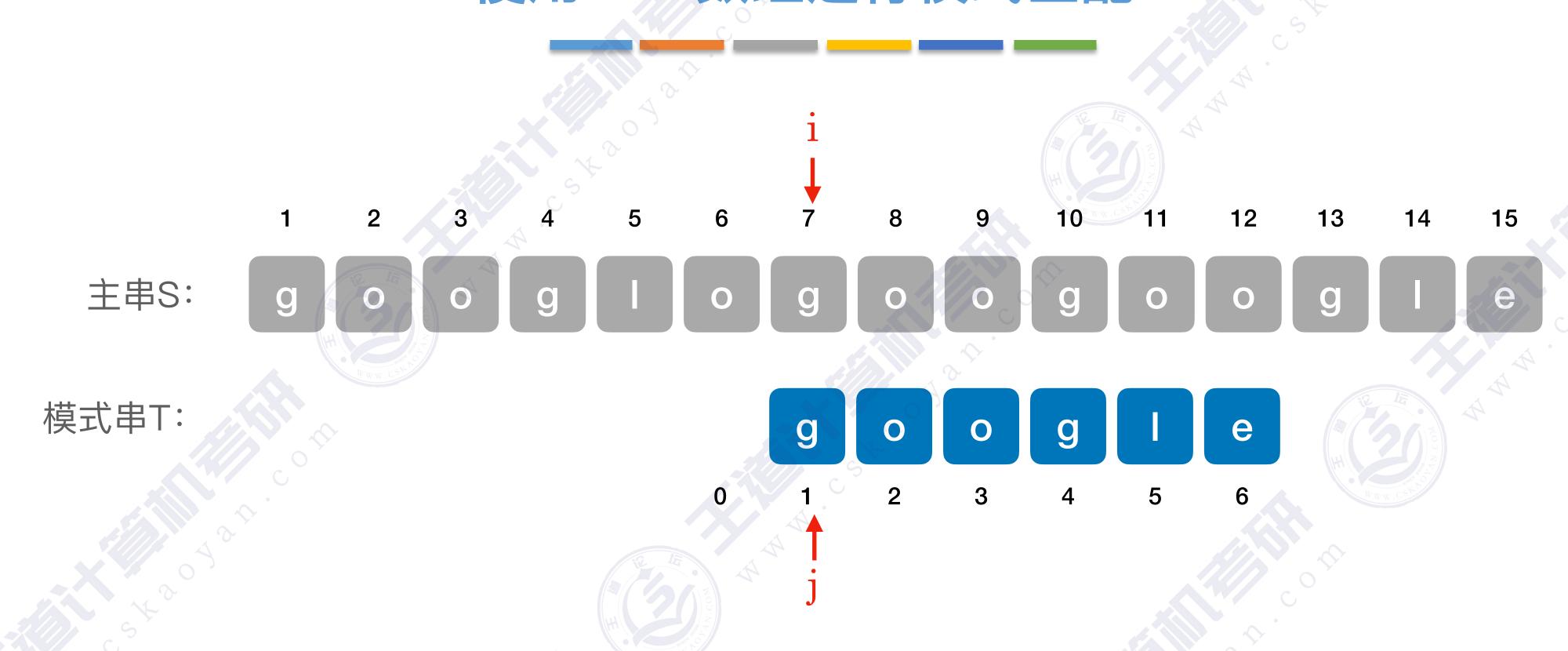
next[0	] next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
707	0	1	1	1	2	1.6

j=next[j]



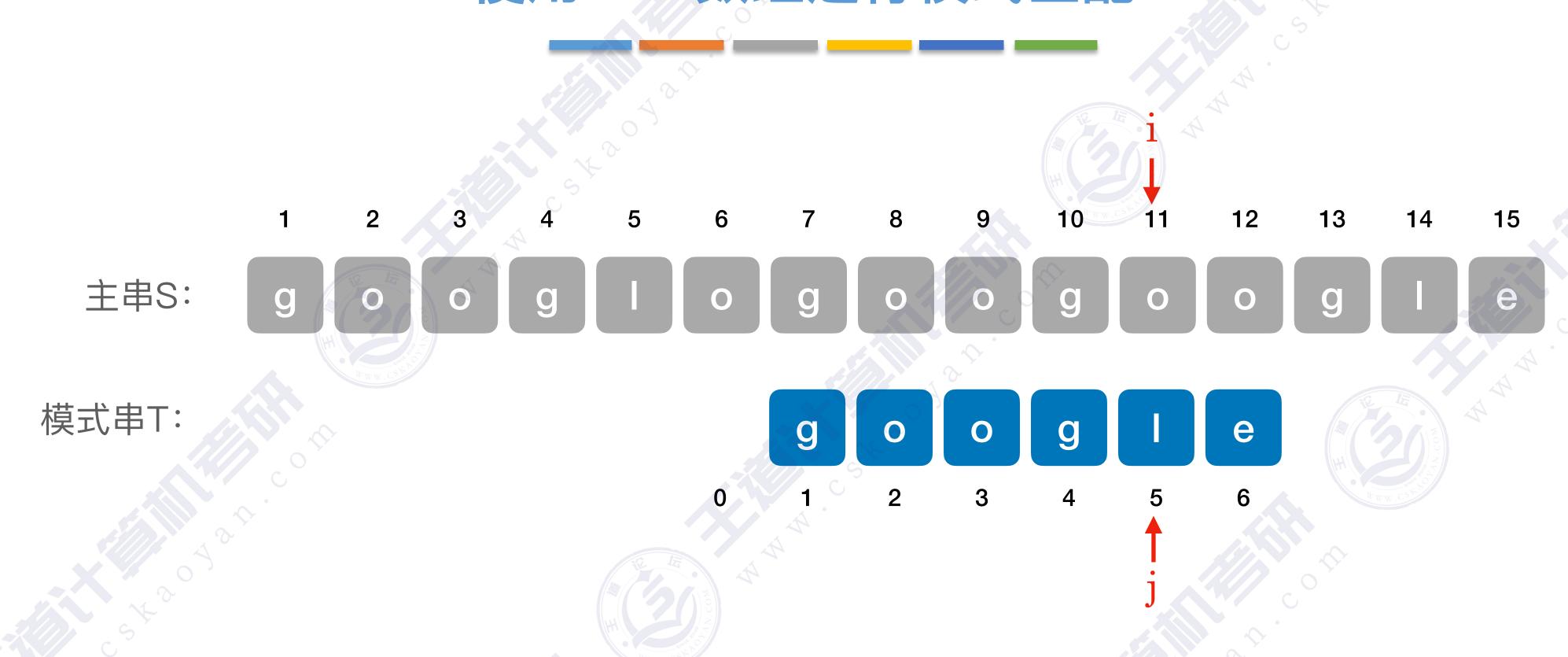
next数组:	
---------	--

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
3	0	1	1	1	2	1.6



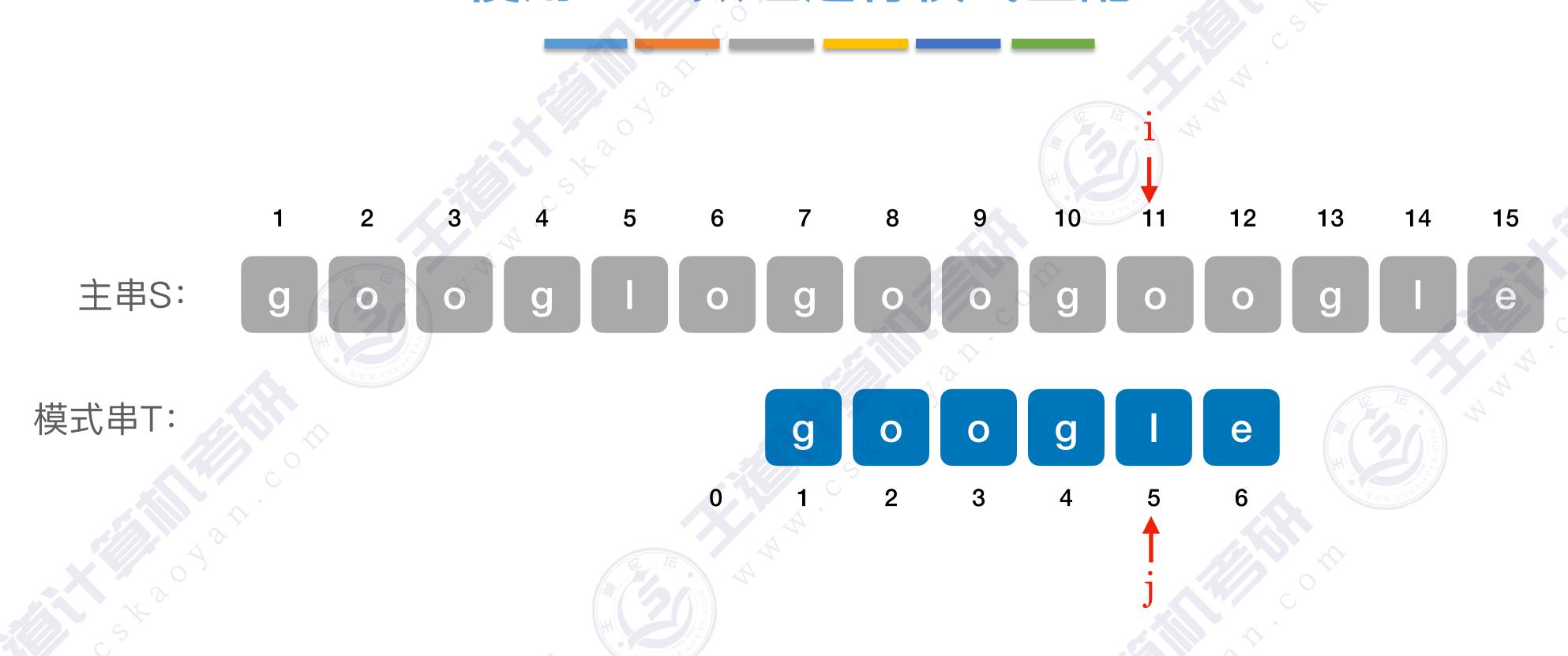
next数组:	
---------	--

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	1 🔊



next数组

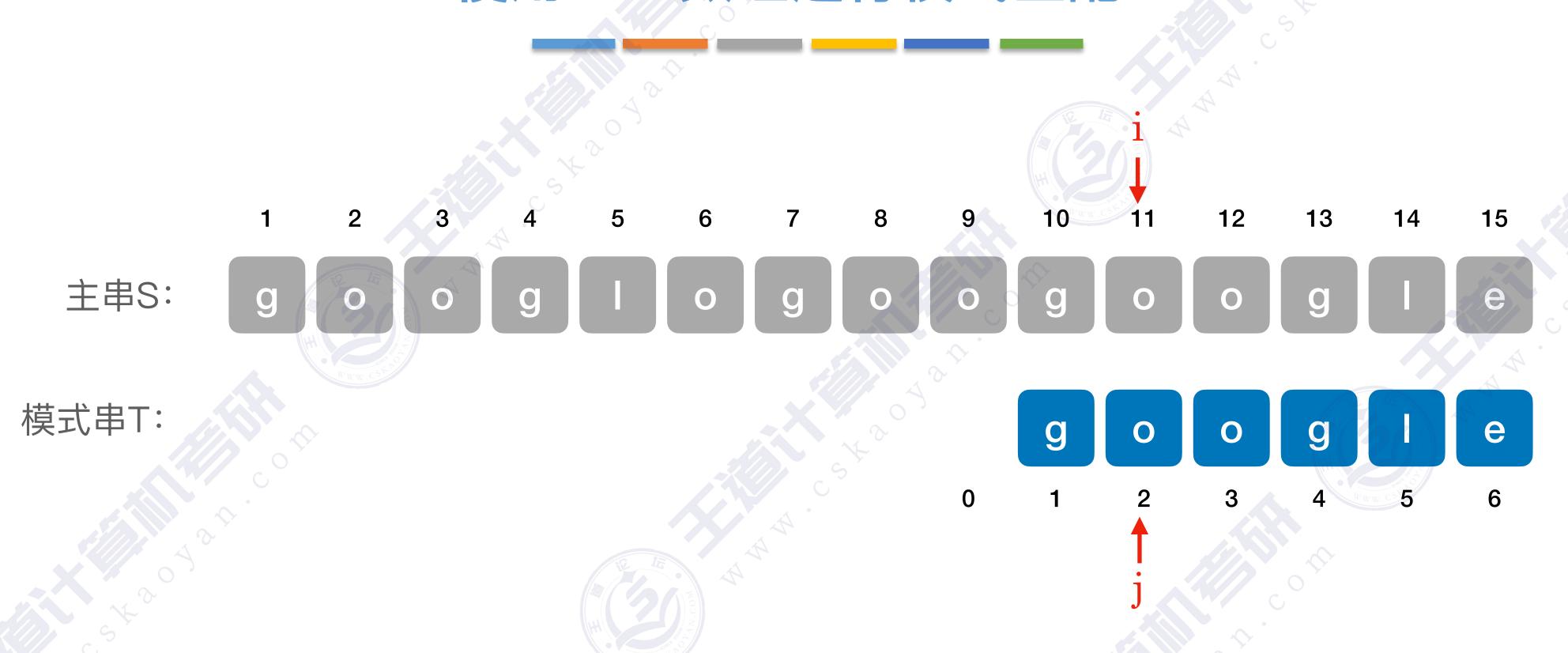
next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
3	0	1	1	1	2	1.6



next数组:

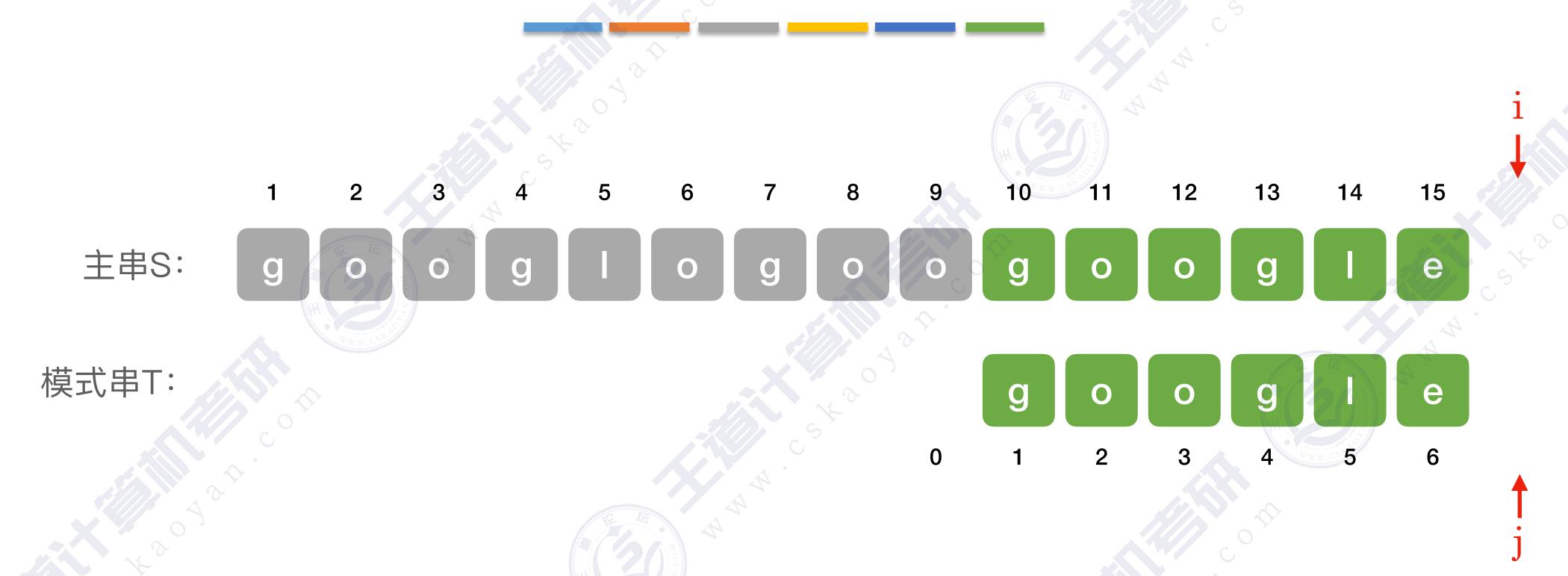
next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	1.5

j=next[j]



next数组:

next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	1.6





next[0]	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
400	0	1	1	1	2	1.3

模式串 T = ababaa

序号j	1	2	3	4	5	6
模式串	а	b	а	b	a	а
next[j]						

next[1]都无脑写 0 next[2]都无脑写 1



模式串 T = ababaa

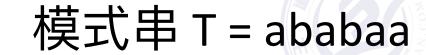
# ababaa

序号j	1	2	3	4	5	6
模式串	a	b	а	b	а	а
next[j]	0	1				

next[1]都无脑写 0 next[2]都无脑写 1



ab???abaaa

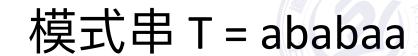


序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	a
next[j]	0	1			NOO!	

next[1]都无脑写 0 next[2]都无脑写 1



ab???
ababaa



序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	а
next[j]	0	1			COM T	

next[1]都无脑写 0 next[2]都无脑写 1



ab???
ababaa

模式串 T = ababaa

序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	a
next[j]	0	1			TOO!	

next[1]都无脑写 0 next[2]都无脑写 1

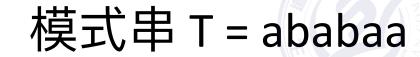


ab??? ababaa 模式串 T = ababaa

序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	a
next[j]	0	1	1 /		COM A	

next[1]都无脑写 0 next[2]都无脑写 1



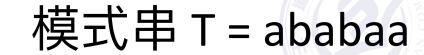


序号j	1	2	3	4	5	6
模式串	a	b	а	b	a	a
next[j]	0	1	1		NOO!	

next[1]都无脑写 0 next[2]都无脑写 1



aba??
abaaaa

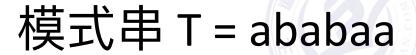


序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	а
next[j]	0	1	1 (		Y V	

next[1]都无脑写 0 next[2]都无脑写 1



aba???
ababaa



序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	а
next[j]	0	1	1 (		COM T	

next[1]都无脑写 0 next[2]都无脑写 1



aba???
ababaa

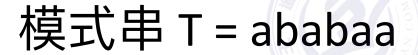
模式串 T = ababaa

序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	а
next[j]	0	1	1 (	2	NOO!	

next[1]都无脑写 0 next[2]都无脑写 1



abab?aabaa

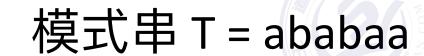


序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	а
next[j]	0	1	1 /	2	NOO!	

next[1]都无脑写 0 next[2]都无脑写 1



abab??
ababaaa

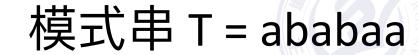


序号j	1	2	3	4	5	6
模式串	a	b	а	b	a	a
next[j]	0	1	1 /	2	COM T	

next[1]都无脑写 0 next[2]都无脑写 1



abab??
ababaa



序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	а
next[j]	0	1	1 (	2	NOO!	

next[1]都无脑写 0 next[2]都无脑写 1



abab??
ababaa

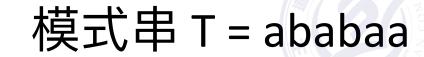
模式串 T = ababaa

序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	а
next[j]	0	1	1 (	2	3	

next[1]都无脑写 0 next[2]都无脑写 1



ababa?
ababaa

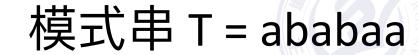


序号j	1	2	3	4	5	6
模式串	a	b	a	b	а	a
next[j]	0	1	1 /	2	3	

next[1]都无脑写 0 next[2]都无脑写 1



ababa?
ababaa



序号j	1	2	3	4	5	6
模式串	a	b	a	b	а	a
next[j]	0	1	1 /	2	3	

next[1]都无脑写 0 next[2]都无脑写 1



ababa?
ababaa

模式串 T = ababaa

序号j	1	2	3	4	5	6
模式串	а	b	a	b	а	а
next[j]	0	1	1 (	2	3	

next[1]都无脑写 0 next[2]都无脑写 1



ababa?
ababaa

模式串 T = ababaa

序号j	1	2	3	4	5	6
模式串	a	b	а	b	a	a
next[j]	0	1	1	2	3	4

next[1]都无脑写 0 next[2]都无脑写 1





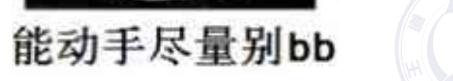
能动手尽量别bb





序号j	1	2	3	4	5
模式串	a	a	а	a	b
next[j]			04		









序号j	1	2	3	4	5
模式串	a	а	а	a	b
next[j]	0	1	2	3	4



#### KMP算法——求next数组

#### 根据模式串T,求 出 next 数组

#### T = 'abaabc'

#### next数组:

next[0] n	next[1]	next[2]	next[3]	next[4]	next[5]	next[6]
	0	10	1	2	2	3

if (j==0) { i++; j++ }

#### KMP算法,最坏时间复杂度 O(m+n)

其中,求 next 数组时间复杂度 O(m) 模式匹配过程最坏时间复杂度 O(n) next[1]都无脑写 0 next[2]都无脑写 1