

串的定义和实现【2023年考点不考这个，了解即可】

定义	<div><ul style="list-style-type: none">串：字符串简称串，是由零个或多个字符组成的有限序列子串：串中任意多个连续的字符组成的子序列主串：包含子串的串某个字符在串中的序号称为该字符在串中的位置空格串：由一个或多个空格组成的串</div>
定长顺序存储表示	<div><pre>#define MAXLEN 255 typedef struct { char ch[MAXLEN]; int length; } SString;</pre></div>
堆分配存储表示	<div><pre>typedef struct { char *ch; int length; } SString;</pre></div>
块链存储表示	<div><ul style="list-style-type: none">类似于线性表的链式存储结构也可采用链表方式存储串值其中节点称为块最后一个节点占不满时，用#填充</div>
基本操作	<div><ul style="list-style-type: none">StrAssign(&T,chars):赋值操作StrCopy(&T,S):赋值操作StrEmpty(S):判空操作StrCompare(S,T):比较操作StrLength(S):求串长StrString(&Sub,S,pos,len):求子串Concat(&T,S1,S2):串联接Index(S,T):定位操作ClearString(&S):清空操作DestroyString(&S):销毁串</div>

串的模式匹配

- 模式匹配：子串的定位操作，求的是子串（模式串）在主串中的位置

	解释说明	代码
暴力匹配算法-BF算法	<div><p>① 最好情况,平均时间复杂度 $O(m+n)$ [i,j不用回退]</p><p>② 最坏情况,平均时间复杂度 $O(mn)$ [i,j一直在回退]</p><div><div><div><div>S</div><div>a b a b c a b c a c b a b</div></div><div><div>t</div><div>a b c</div></div></div><div><div>i=3</div><div>↓</div><div>↑</div><div>j=3</div></div><div>Round 1</div></div><div><div><div>S</div><div>a b a b c a b c a c b a b</div></div><div><div>T</div><div>a</div></div></div><div><div>i=3-3+2=2</div><div>↓</div><div>↑</div><div>j=1</div></div><div>Round 2</div></div>	<pre>int Index_BF(string s, string t) { int i = 1, j = 1; int lens = s.length(); int lent = t.length(); while (i < lens && j < lent) { if (s[i] == t[j]) { i++, j++; continue; } else { i = i - j + 2; // i指示主串S正在比较的字符位置 j = 1; // j指示子串t正在比较的字符位置 } } if (j == lent) { return i - j; } return -1; }</pre>
模式匹配算法-KMP算法	<div><div>主串S A B A B A B C A A</div><div>模式T A B A B C</div><div><div>Round 1</div><div><div>S</div><div>A B A A B A B C A A</div></div><div><div>next数组</div><div>A B A B C</div><div>0 1 1 2 3</div></div><div>↓ 跳过2个字符</div></div><div><div>Round 2</div><div><div>S</div><div>A B A A B A B C A A</div></div><div><div>next数组</div><div>A B A B C</div><div>0 1 1 2 3</div></div><div>↓ 跳过1个字符</div></div><div><div>Round 3</div><div><div>S</div><div>A B A A B A B C A A</div></div><div><div>next数组</div><div>A B A B C</div><div>0 1 1 2 3</div></div></div></div> <div><div>next数组的值 = 寻找子串中最长相同前后缀的长度+1</div><div>求 ababaaababaa 的 next 数组 (位置从1开始)</div><div><div>aba 相同前后缀为 a, 值=1+1=2</div><div>ababaaababaa</div><div>0 1 1 2 3 4 2 2 3 4 5 6</div><div>固定的 ↓</div><div>ab 元相同前后缀 值=0+1=1</div><div>ababaaababaa</div><div>相同前后缀为 ababa</div><div>值=5+1=6</div></div></div>	<pre>void get_next(String T, int next[]) //计算next数组 { int i = 1, j = 0; next[1] = 0; while (i < T.length) { if (j == 0 T.ch[i] == T.ch[j]) { ++i; ++j; next[i] = j; } else { j = next[j]; } } } int Index_KMP(String S, String T, int next[]) { int i = 1, j = 1; while (i <= S.length && j <= T.length) //i永远不递减 { if (j == 0 S.ch[i] == T.ch[j]) //字符匹配成功,指针后移 { ++i; ++j; } else //字符匹配失败,根据next跳过前面的一些字符 { j = next[j]; } } if (j > T.length) // 匹配成功 { return i - T.length; } else { return 0; } }</pre>