第六章　串

本章内容

6.1 串的基本概念
6.2 串的基本操作
6.3 串的存储结构
6.4 关于串的几个算法

6.1 串的基本概念

一. 串的定义

串是由n≥0个字符组成的有限序列，通常记为

$S = ' a_1\ a_2\ a_3\ \ldots\ a_{n-1}\ a_n '$

其中，S表示串名(也称串变量)，一对引号括起来的字符序列称为串值，a_i可以是字母、数字或其他允许的字符。n为串的长度，长度为0的串称为空串。

例如

S1= ′ abc ′

S2= ′ FORTRAN_77 ′

S3= ′ ′ = Φ　　(空串)

说明

1. 串值须用一对引号括起来，但引号不属于串值。
2. 要区分空串与由空格字符组成的串的不同。

前者长度为0，后者长度为串中空格字符个数

String = ′ String ′

S1＝′′， S2=′ ′，S1≠S2

二. 几个名词概念

1. 子串：串中若干个连续的字符组成的子序列。

例如：　S=＇Beijing&Shanghai＇

T=＇jing＇

2. 主串：包含子串的串。

3. 位置：(1). 单个字符在主串中的位置被定义为该字符在串中的序号。

(2). 子串在主串中的位置被定义为主串中首次出现的该子串的第一个字符在主串中的位置。

例如：　S=＇Beijing&Nanjing&Shanghai＇

T=＇jing＇

位置为4

4. 两个字符串相等的充分必要条件为两个字符串的长度相等，并且对应位置上的字符相同。

′abcd′ ≠ ′bacd′

′abcd′ = ′abcd′

6.2 串的基本操作

C函数

1. 给串变量赋值 ASSIGN(S1,S2) strcpy(S1,S2)
2. 判断两个串是否相等 EQUAL(S1,S2) strcmp(S1,S2)
3. 两个字符串连接 CONCAT(S1,S2) strcat(S1,S2)
4. 求字符串的长度 LEN(S) strlen(S)
5. 求子串 SUBSTR(S,i,k)
6. 求子串在主串中的位置 INDEX(S1,S2) strstr(S1,S2)
7. 串的替换 REPLACE(S,S1,S2)
8. 串的复制 COPY(S1,S2) strcpy(S2,S1)
9. 串的插入 INSERTS(S1,i,S2)
10. 串的删除 DELETES(S,i,k)

……

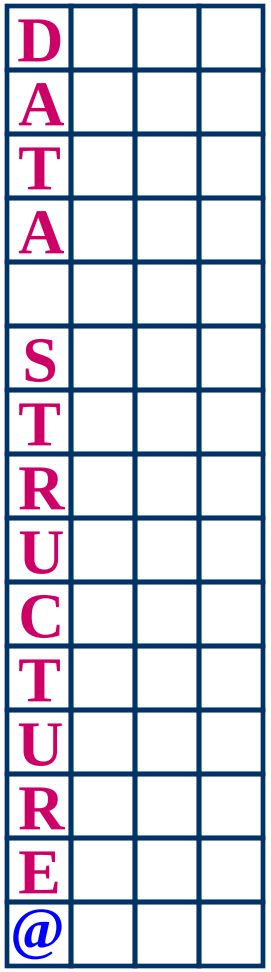
模式匹配

6.3 串的存储结构

一. 串的顺序存储结构

1. 非紧缩格式(设每个字有4个字节)

例如：S = ′ DATA STRUCTURE ′



2. 紧缩格式

D	A	T	A
	S	T	R
U	C	T	U
R	E	@	

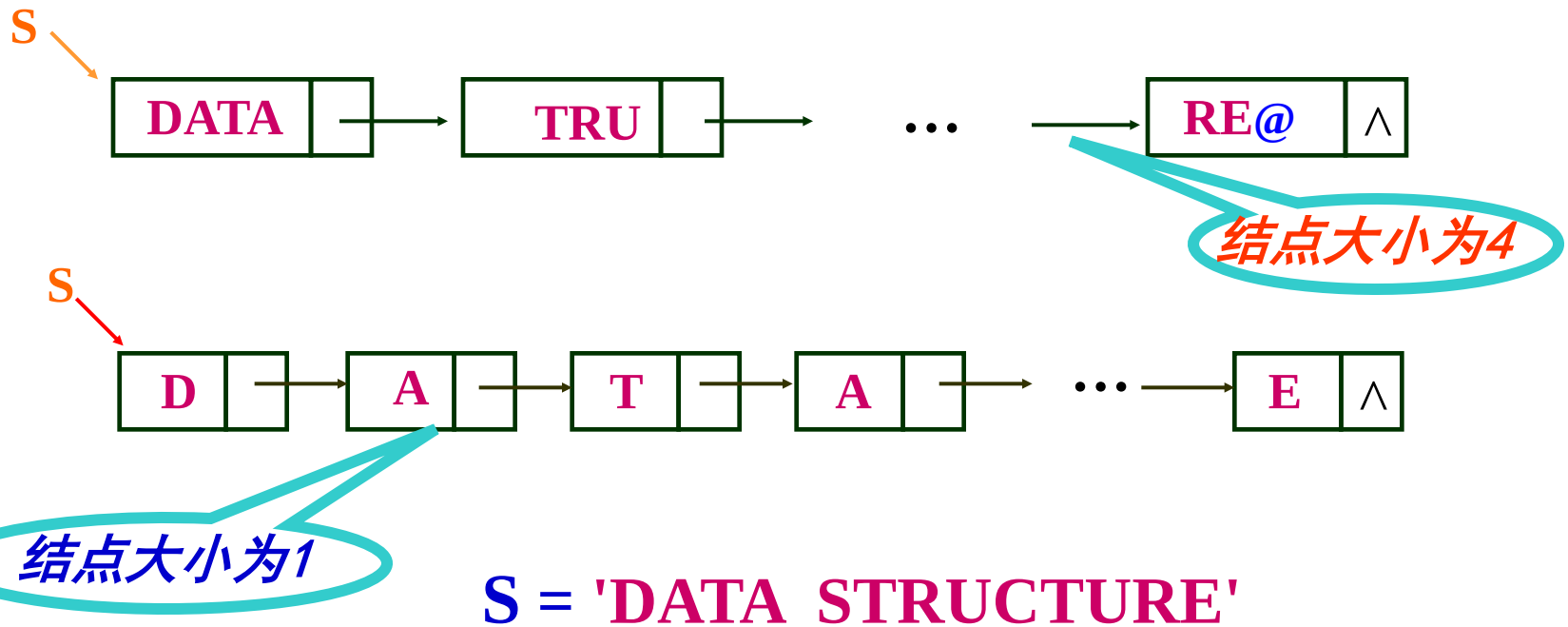
3. 单字节方式

D	A	T	A		S	T	R	U	C	T	U	R	E	@

二. 串的链式存储结构

说明：

所谓链结点大小是指每个链结点的数据域中存放的字符的个数。

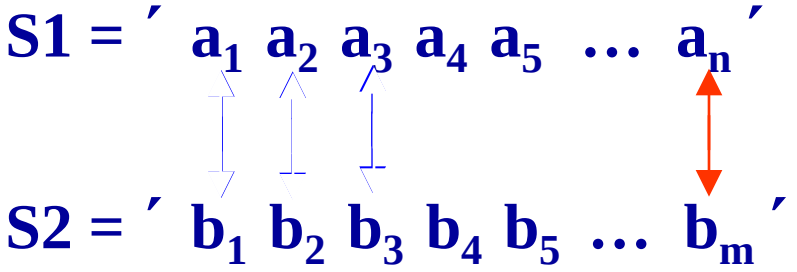


S = 'DATA STRUCTURE'

6.4 串的几个算法

一. 判断两个字符串是否相等

功 能：在单字节方式中，两个字符串分别存放于数组S1与S2中，并且都以@ 作为串的结束标志；判断两个串是否相等，若相等返回信息1，否则，返回信息0。



算法

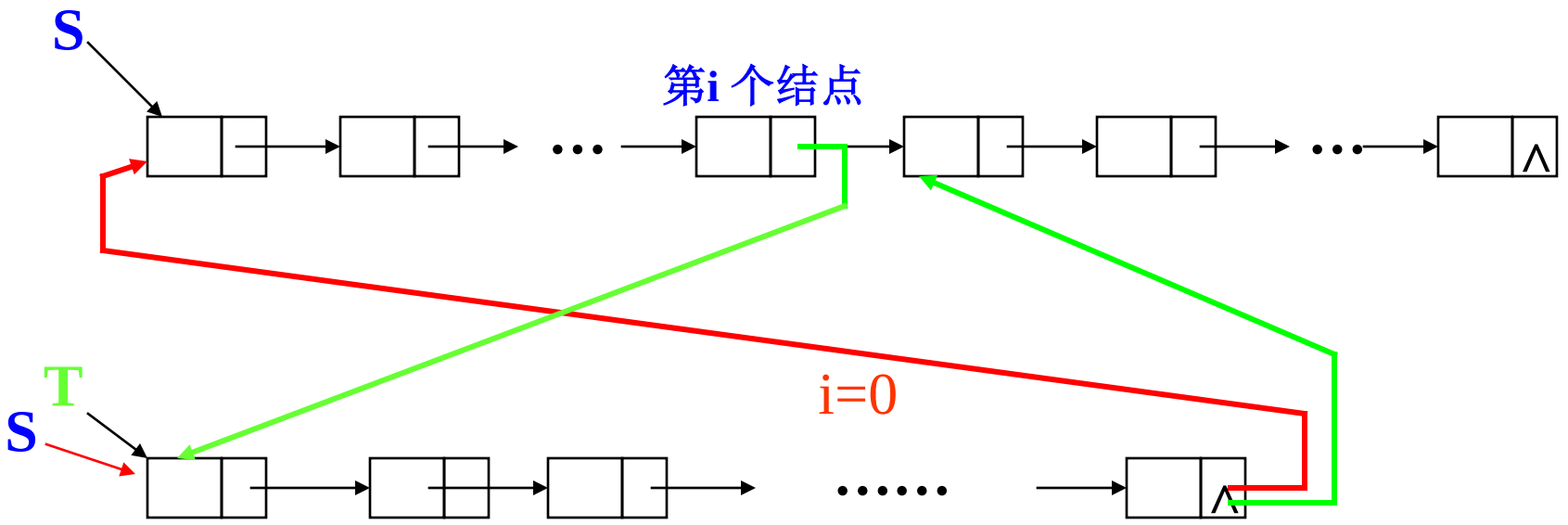
```
int  EQUAL( char S1[ ],  char S2[ ] )
{    int i=0;
     while  ( S1[i]!=′@′  &&  S2[i]!=′@′ ) {
          if ( S1[i]!=S2[i] )
               return  0;
          i++;
     }
     if ( S1[i]==′@′  && S2[i]==′@′ )
           return 1;
     return 0;
}
```

二. 串的插入

功能：在字符串S的第i个字符后面插入字符串T。

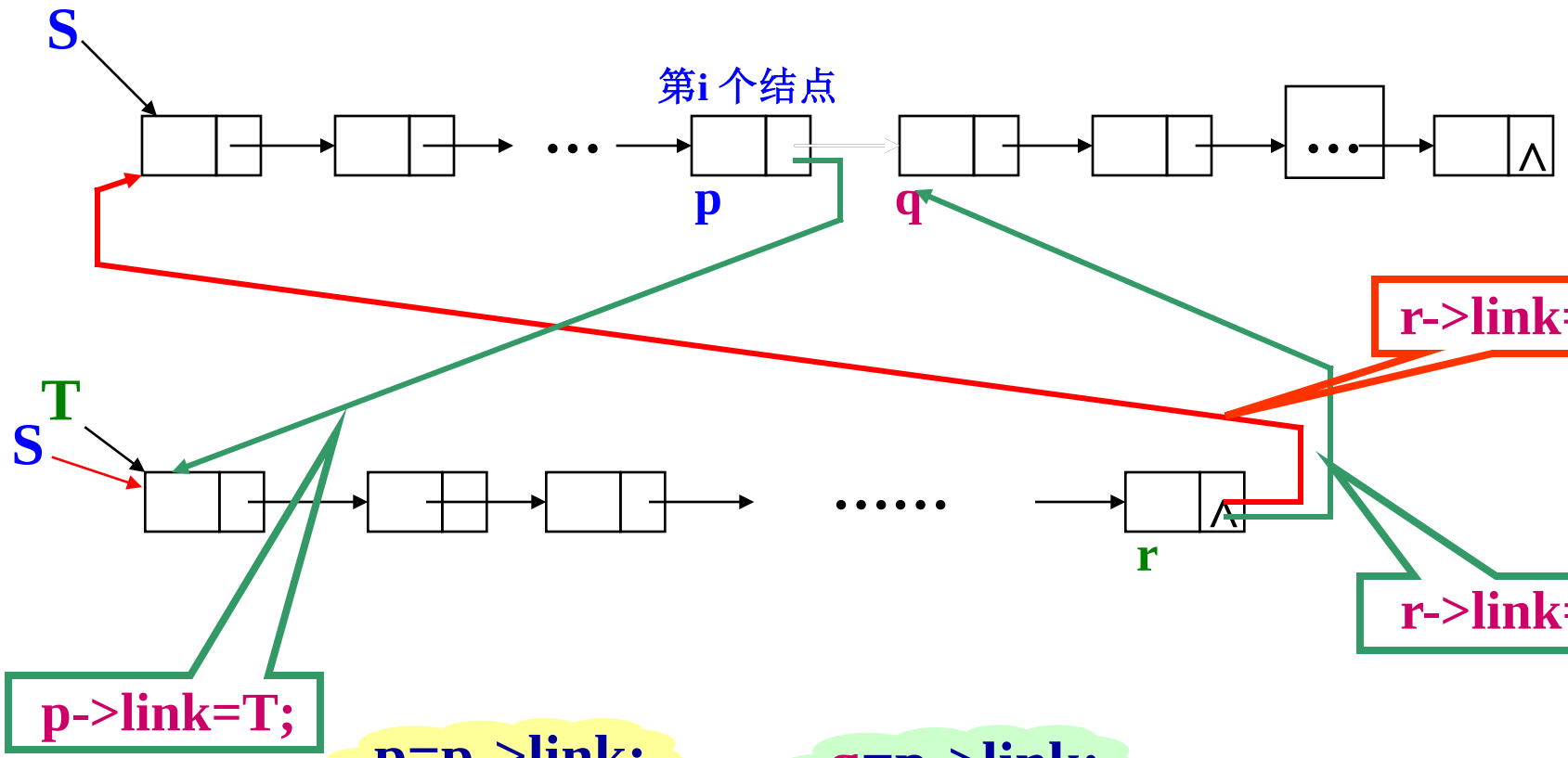
前提：字符串S与T分别采用结点大小为1的线性链表存储结构(设S与T分别指向链表的第一个链结点)。

约定：1. 当i=0 时，将T 插在S 的最前面。
2. 结果串由S 指出。



需要做的工作：

1. 找到S 的第i个字符的位置（即S的第i个链结点的地址）.
2. 找到S 的第i+1个字符的位置（即S 的第i+1个链结点的地址）.
3. 找到T 的最后那个字符的位置（即T 的尾结点的地址）.
4. 插入


S
第i 个结点
p
q
r->link=S;
T
S
r
r->link=q;
p->link=T;
p=p->link;
q=p->link;
做i-1次
while (r->link!=NULL)
r=r->link;

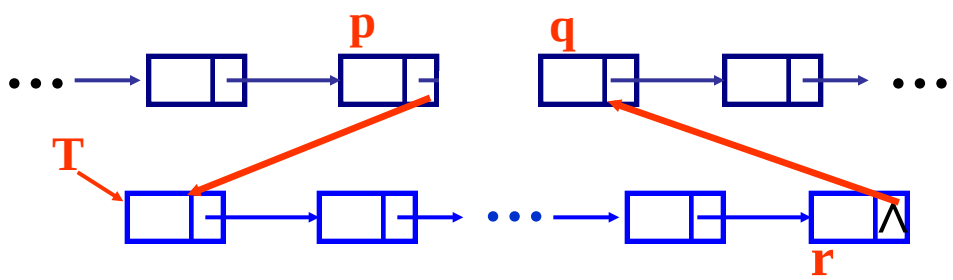

```
StrLink   INSERTS( StrLink S, StrLink T, int i )
{
      StrLink  p, r;
      int  j;
      if ( T!=NULL ) {
           if ( S==NULL ) {
                S=T;
                return S;
           }
       p=S;
       for( j=1; j<i; j++ ) {
            p=p->link;
            if ( p==NULL )
                 return  NULL;
       }                                  /*p 指向S 的第i个字符*/
       q=p->link;                        /*q 指向S 的第i+1个字符*/
```

若S串中不存在第i个字符

```
        r=T;
        while ( r->link!=NULL )
            r=r->link;                  /*r指向T 的最后那个字符*/
        if ( i==0 ) {
            r->link=S;
            S=T;                        /*将T 插在S 的最前面*/
        }
        else {
            p->link=T;
            r->link=q;                  /*将T 插在S 的第i个字符后面*/
        }
        return S;
    }
}
```

寻找T串最后字符的位置

r->link=p->link;
p->link=T;



三. 串的模式匹配

也称子串定位。设S与PAT分别为两个字符串，串的模式匹配就是以PAT为模式，查看字符串源S中有无PAT这样的子串。

书上的匹配算法很简单，就是从S的第一个字符开始依次与PAT进行比较，若不同，则从S的第二个字符再与PAT进行比较，…，算法时间复杂度为O（m x n），改进的算法是先从PAT的尾部与S中相应的尾部比，相同后再从S中相应的头部与PAT从头到尾挨个比。清华严蔚敏的教材介绍了KMP模式匹配算法，利用模式中的特征向量来控制比较字符的位置

功能：在单字节方式中，两个字符串分别存放于数组S1 与S2中，S1中可能带有元字符“*”和“?”，判断两个串是否相等，若相等返回信息1，否则，返回信息0。

a?cd -> abcd axcd a2cd a+cd

a*cd -> acd abcd alotofcd a!$cd


```
function   MATCH( S1, S2, p1, p2 )
    test ←0                    // 是否匹配的标记，初值为0//
    if S1[p1]='*' then                    //S1中有元字符*//
      [ if p2<length(S2)  then          //没到S2的结尾//
           test ←test OR MATCH(S1,S2,p1,p2+1)
        p1 ←p1+1]
    if p1<length(S1) AND p2<length(S2) then
       if S1[p1]=S2[p2] OR S1[p1]='?' then
          test ←test OR MARCH(S1,S2,p1+1,p2+1)
    else
        if p1=length(S1) AND p2=length(S2) then
          test ←1
     return (test)
end
```

算法

```
int   MATCH( i,j )
    char *i, *j;
   { int test =0;  //  是否匹配的标记，初值为0//
     if (*i==‘*’ ) {  //在字符串中有元字符//
         while (i[1]==‘*’)  i++;  //跳过重复的元字符//
         if (*j)  test |= MATCH (i,j+1);
          i++ ;
     }
    if (*i &&*j)
        if (*i==*j || *i==‘?’ )
           test |= MATCH (i+1,j+1);
    else
          if (*i==*j )
              test=1;
     return test;
  }
```

本章内容小结

字符串

字符串的基本概念

- 字符串的定义
- 基本的名词概念

子串、主串、位置、两串相等

字符串的存储结构

- 顺序存储结构

紧缩格式、非紧缩格式、单字节格式

- 链式存储结构

关于链结点大小

关于字符串的几个基本算法

- 判断两串相等（顺序存储结构）
- 串的插入（链式存储结构）
- 模式匹配（链式存储结构）