第四章 串

串也叫字符串，它是由零个或多种字符构成的的字符序列。

基本内容

1　串的有关概念　串的基本操作
2　串的定长顺序存储构造，堆分配存储构造；
3　串的基本操作算法；
4　串的模式匹配算法；

学习要点

1　了解串的基本操作，了解利用这些基本操作实现串的其他操作的措施；
2　掌握在串的堆分配存储构造下，串的基本操作算法；
3　掌握在串的模式匹配算法；

第四章　串

4. 1 串的基本概念

一、串的定义

1 什么是串

串是一种特殊的线性表，它是由零个或多种字符构成的有限序列，一般记作 s = ‘a_1,$a_{2,}$ $a_{3,}$ ... a_n’

其中 s----串名， a_1,$a_{2,}$ $a_{3,}$ ... a_n----串值

串的应用非常广泛，许多高级语言中都把串的作为基本数据类型。在事务处理程序中，顾客的姓名、地址货品的名称、产地可作为字符串处理，文本文件中的每一行字符等也可作为字符串处理。

4. 1 串的基本概念

下面是某些串的例子：

（1）a = ‘ LIMING’

（2）b = ‘ PEJING UNIUERCITY’

（3）c = ‘ DATA STRUCTURE’

（4）d = ‘’

（5）e = ‘ ’

阐明：

1) 串中涉及的字符个数，称为串的长度。

长度为0的串称为空串，它不涉及任何字符，上面（4）中的串d是空串,，(5)中的e 是涉及一种空格符的空格串；

2）串中所涉及的字符能够是字母、数字或其他字符，这依赖于详细计算机所允许的字符集。

2 串的有关术语

1）子串

串中任意连续的字符构成的子序列称为该串的子串

例：c = ‘ DATA STRUCTURE’，f=‘DATA’ f是c的子串

2）子串的位置

子串T 在主串S中的位置是指主串S中第一种与T相同的子串的首字母在主串中的位置。

例：S=‘ababcabcac’，T=‘abc’ 子串T 在主串S中的位置为3

3）串相等

两个串相等，当且仅当两个串长度相同，而且各个相应位置的字符都相同；

3 串的基本操作

串的逻辑构造与线性表一样，都是线性构造。但因为串的应用与线性表不同，串的基本操作与线性表有很大差别。

1）串赋值操作StrAssign(&T, chars)

功能：将串常量char的值赋给串变量T;

2）复制串操作 StrCopy(&T,S)

功能：由串变量S复制得到串变量T;

3）判空操作 StrEmpty(S)

功能：若为空串，则返回TRUE，不然返回FALSE

4） 串比较操作 StrCompare(S, T)

功能若S>T，则返回值>0；若S=T，则返回值=0；若S<T，则返回值<0

5）串置空操作 ClearString(&S)

功能：将S清为空串

7） 串连接操作 Concat(&T, S1, S2)

功能：由S1和S2连接构成的新串，用T返回新串；

8） 求子串操作SubString(&Sub, S, pos, len)

功能：用Sub返回串S的第pos个字符起长度len 为子串；

9）求子串位置操作Index(S, T, pos)

功能：返回S中第pos个字符之后与T相同的子串的位置，若不存在返回0

10) 串插入操作 StrInsert(&S, pos , T)

功能：将串T插入到串S的第pos字符之前

11）串删除操作 StrDelete(&S, pos , len)

功能：从串S中删除第pos个字符起长度len 为子串

一、串的存储

1 定长顺序存储构造

定长顺序存储构造类似于C语言的字符数组，以一组地址连续的存储单元存储串值字符序列，其类型阐明如下：

```
#define MAXSTRLEN 255
Typedef unsigned char SString[MAXSTRLEN+1]
```

2、堆分配存储

堆分配存储类似于线性表的顺序存储构造，以一组地址连续的存储单元存储串值字符序列，其存储空间是在程序执行过程中动态分配的。

在C语言中，存在一种称之为“堆”的自由存储区，并由C语言的动态分配函数管理。利用函数malloc( )为每个新产生的串分配一块实际串长所需的存储空间，若分配成功，则返回一种指向起始地址的指针，作为串的基址，同步，为了后来处理以便，将串长也作为存储构造的一部分。串的这种存储构造本教材中称为堆分配存储

堆分配存储的类型阐明

```
Typedef struct {
        char *ch;      //指针域，指向存储串值的存储空间基址
        int   length; // 整型域：存储串长
}Hstring;
```

在这种存储构造下，串操作仍是基于“字符序列的复制”进行的。例如，如串插入操作StrInsert(S,pos ,T)（将串T插入到串S的第pos字符之前）的算法是，为串S重新分配大小等于串S和串T长度之和的存储空间，然后进行将S和T串值复制到新分配存储空间中。

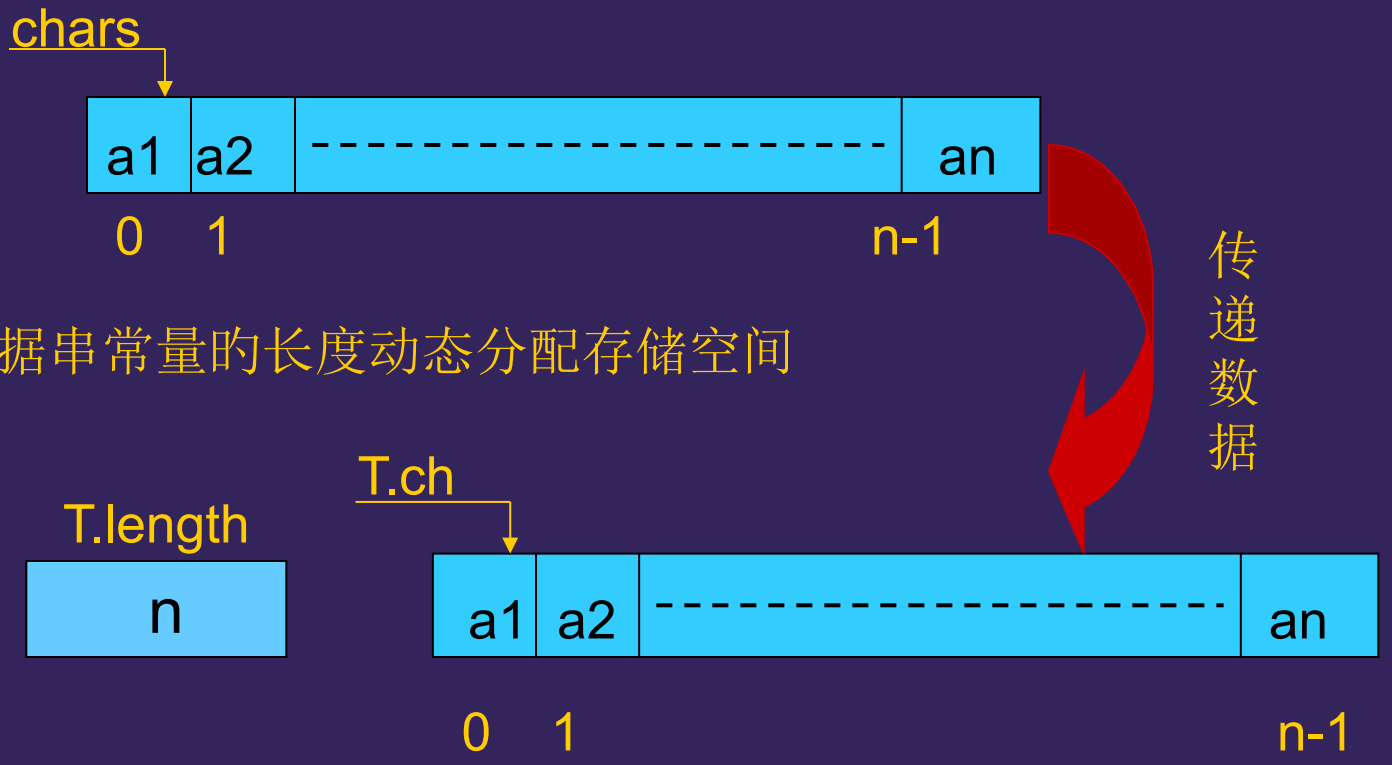
以上两种存储表达一般为高级程序设计语言所采用。因为堆分配存储构造的串既有顺序存储构造的特点，处理以便，操作中对串长又没有任何限制，更显灵活，所以在串处理的应用程序中常被采用。下列我们给出在堆分配存储构造下，串的部分基本操作算法。

二 串基本操作算法

串赋值算法

```
Status StrAssign(HString &T, char * chars) {
//生成一种其值等于串常量chars 的串T
   if (T.ch) free(T.ch);    //若若T.ch非空，释放T.ch所指向的存储空间
   for (i=0, c=chars; c; ++i, ++c);    //求chars 的长度i
   if (!i)  {T.ch =NULL;  T.length=0;}     //若i=0,生成空串T
   else{
        if (!(T.ch=(char * )malloc(i * sizeof(char))))
             exit (OVERFLOW);
        T.ch[0..i-1]=chars[0..i-1];
        T.length=i;
   }
  return OK;
}//StrAssign
```

串常量


chars
a1
a2
an
0
1
n-1
传递数据
根据串常量的长度动态分配存储空间
T.ch
T.length
n
a1
a2
an
0
1
n-1

串赋值操作图示

串比较算法

```
int StrCompare(HString S, HString T){
//若S>T，则返回值>0；若S=T，则返回值=0；若S<T，则返回值<0
for (i=0; i<S.length && i<T. length; ++i)
        if (S.ch[i]!=T.ch[i])   return S.ch[i]-T.ch[i];
return S. length-T. length;
}//StrCompare
```