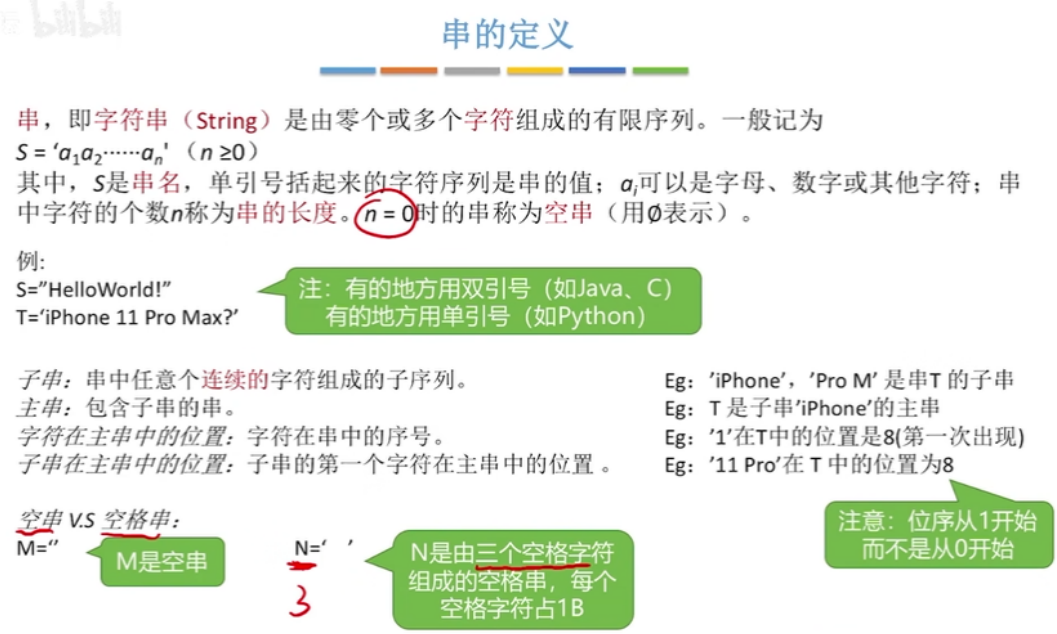
串

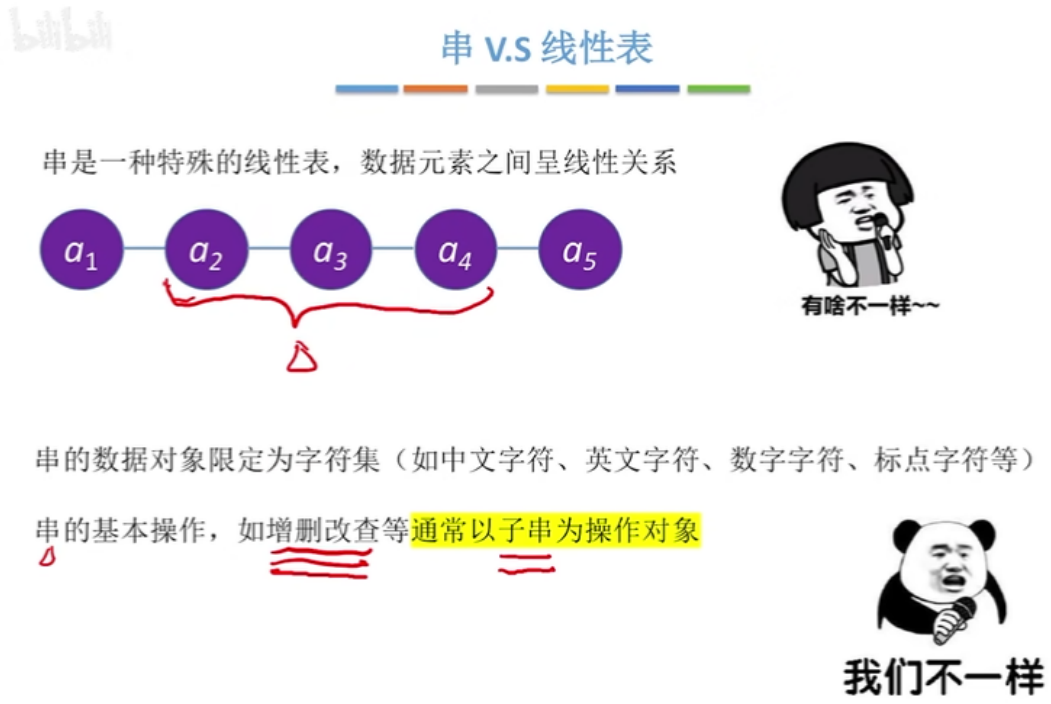
一、串的定义和基本操作

- **串的定义**

  串，即字符串(String)是由零个或多个字符组成的有限序列。一般记为S = 'a_1a_2.....a_n'(n >= 0)；其中，S是串名，单引号括起来的字符序列是串的值；a_i可以是字母、数字或其他字符；串中字符的个数n称为串的长度；n = 0时的串称为空串（用Φ表示）



- **串V.S线性表**



- **串的基本操作**

串的基本操作

假设有串T=“”，S=”iPhone 11 Pro Max?”，W=“Pro”

StrAssign(&T,chars)：赋值操作。把串T赋值为chars。
StrCopy(&T,S)：复制操作。由串S复制得到串T。
StrEmpty(S)：判空操作。若S为空串，则返回TRUE，否则返回FALSE。
StrLength(S)：求串长。返回串S的元素个数。
ClearString(&S)：清空操作。将S清为空串。
DestroyString(&S)：销毁串。将串S销毁（回收存储空间）。
Concat(&T,S1,S2)：串联接。用T返回由S1和S2联接而成的新串
SubString(&Sub,S,pos,len)：求子串。用Sub返回串S的第pos个字符起长度为len的子串。
Index(S,T)：定位操作。若主串S中存在与串T值相同的子串，则返回它在主串S中第一次出现的位置；否则函数值为0。
StrCompare(S,T)：比较操作。若S>T，则返回值>0；若S=T，则返回值=0；若S<T，则返回值<0。

Eg：执行基本操作 Concat(&T, S, W) 后，T=“iPhone 11 Pro Max?Pro”　存储空间扩展？
执行基本操作 SubString(&T ,S, 4, 6)后，T=“one 11”
执行基本操作 Index(S, W)后，返回值为 11

- **串的比较操作**

串的比较操作

StrCompare(S,T)：比较操作。若S>T，则返回值>0；若S=T，则返回值=0；若S<T，则返回值<0。

A

abandon/ ə'bændən/ vt.丢弃;放弃，抛弃

aboard/ ə'bɔ:d/ ad.在船(车)上;上船

absolute/ 'æbsəlu:t/ a.绝对的;纯粹的

absolutely/ 'æbsəlu:tli/ ad.完全地;绝对地

absorb/ əb'sɔ:b/ vt.吸收;使专心

abstract/ 'æbstrækt/ n.摘要

abundant/ ə'bʌndənt/ a.丰富的;大量的

abuse/ ə'bju:z, ə'bju:s/ vt.滥用;虐待 n.滥用

academic/ ækə'demik/ a.学院的;学术的

accelerate/ æk'seləreit/ vt.(使)加快;促进

a < o

“abandon” < “aboard”

从第一个字符开始往后依次对比，先出现更大字符的串就更大

长串的前缀与短串相同时，长串更大

“abstract” < “abstraction”　　“abstract”<“abstract ”

只有两个串完全相同时，才相等

“academic”>“abuse”　　“academic”=“academic”

- **字符集编码**

y = f(x)
字符集：函数定义域
编码：函数映射规则 f
y：对应的二进制数

任何数据存到计算机中一定是二进制数。
需要确定一个字符和二进制数的对应规则这就是“编码”

“字符集”：
英文字符——ASCII字符集
中英文——Unicode字符集

基于同一个字符集，可以有多种编码方案，如：UTF-8，UTF-16

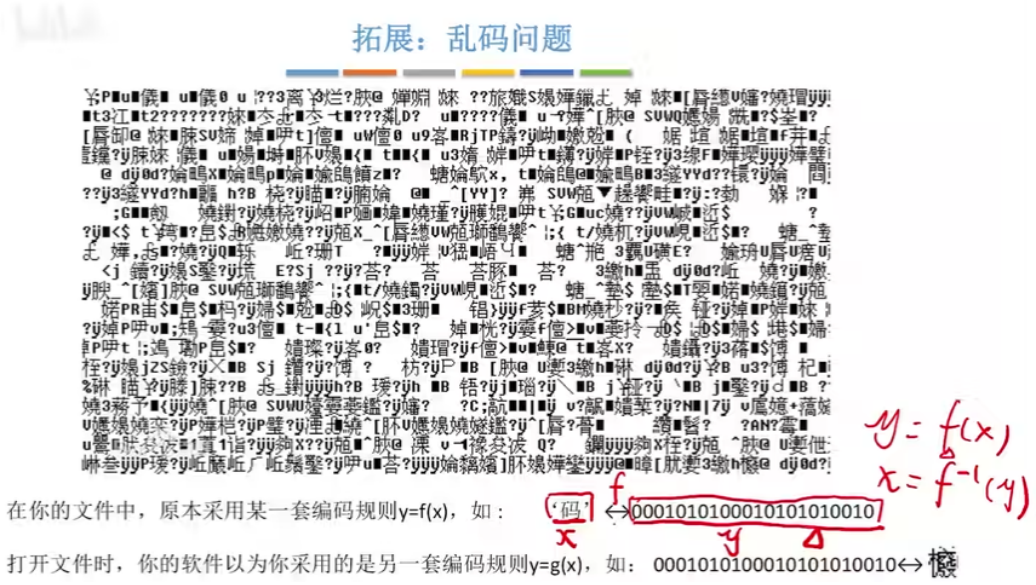
注：采用不同的编码方式，每个字符所占空间不同，考研中只需默认每个字符占1B即可

字符集编码

ASCII 字符代码表 一

高四位 / 低四位	0	1	2	3	4	5	6	7
0000	0	0	16 ►	32 空格	48 0	64 @	80 P	96 `	112 p
0001	1	1 ☺	17 ◄	33 !	49 1	65 A	81 Q	97 a	113 q
0010	2	2 ☻	18 ↕	34 "	50 2	66 B	82 R	98 b	114 r
0011	3	3 ♥	19 ‼	35 #	51 3	67 C	83 S	99 c	115 s
0100	4	4 ♦	20 ¶	36 $	52 4	68 D	84 T	100 d	116 t
0101	5	5 ♣	21 §	37 %	53 5	69 E	85 U	101 e	117 u
0110	6	6 ♠	22 ▬	38 &	54 6	70 F	86 V	102 f	118 v
0111	7	7 •	23 ↨	39 '	55 7	71 G	87 W	103 g	119 w
1000	8	8 ◘	24 ↑	40 (	56 8	72 H	88 X	104 h	120 x
1001	9	9 ○	25 ↓	41)	57 9	73 I	89 Y	105 i	121 y
1010	A	10 ◙	26 →	42 *	58 :	74 J	90 Z	106 j	122 z
1011	B	11 ♂	27 ←	43 +	59 ;	75 K	91 [	107 k	123 {
1100	C	12 ♀	28 ∟	44 ,	60 <	76 L	92 \	108 l	124 |
1101	D	13 ♪	29 ↔	45 -	61 =	77 M	93]	109 m	125 }
1110	E	14 ♫	30 ▲	46 .	62 >	78 N	94 ^	110 n	126 ~
1111	F	15 ☼	31 ▼	47 /	63 ?	79 O	95 _	111 o	127 ⌂ Back Space

注：表中的ASCII字符可以用：ALT + “小键盘上的数字键” 输入



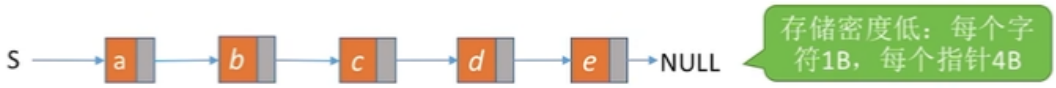
二、串的存储结构

- **串的顺序存储**

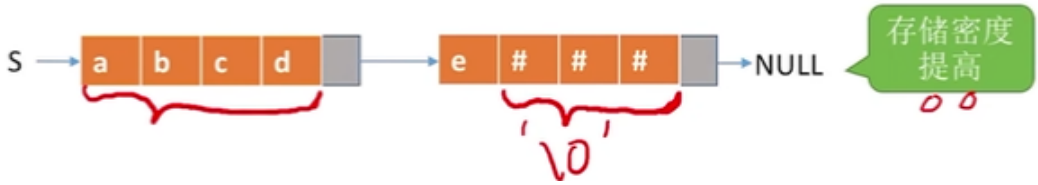
```
#define MAXLEN 255     //预定义最大串长为255
typedef struct
{
    char ch[MAXLEN];   //每个分量存储一个字符
    int length;        //串的实际长度
}SString;

typedef struct
{
    char *ch;          //按串长分配存储区，ch指向串的基地址
    int length;        //串的长度
}HString;

HString S;
S.ch = (char *)malloc(MAXLEN * sizeof(char));
S.length = 0;
```

- **串的链式存储**

```
typedef struct StringNode
{
    char ch;                //每隔结点存1个字符
    struct StringNode * next;
}StringNode, * String;
```




```
typedef struct StringNode
{
    char ch[4];             //每个结点存多个字符
    struct StringNode * next;
}StringNode, * String;
```



- **基本操作的实现**

基本操作的实现


```
#define MAXLEN 255        //预定义最大串长为255
typedef struct{
    char ch[MAXLEN];      //每个分量存储一个字符
    int length;           //串的实际长度
}SString;
```

StrAssign(&T,chars)：赋值操作。把串T赋值为chars。
StrCopy(&T,S)：复制操作。由串S复制得到串T。
StrEmpty(S)：判空操作。若S为空串，则返回TRUE，否则返回FALSE。
StrLength(S)：求串长。返回串S的元素个数。
ClearString(&S)：清空操作。将S清为空串。
DestroyString(&S)：销毁串。将串S销毁（回收存储空间）。
Concat(&T,S1,S2)：串联接。用T返回由S1和S2联接而成的新串

SubString(&Sub,S,pos,len)：求子串。用Sub返回串S的第pos个字符起长度为len的子串。


```
#define MAXLEN 255    //预定义最大串长为255
typedef struct
{
    char ch[MAXLEN];  //每个分量存储一个字符
    int length;       //串的实际长度
}SString;

//求子串
bool SubString(SString &Sub,SString S,int pos,int len)
{
    //子串范围越界
    if(pos + len - 1 > S.length)
        return false;
    for(int i = pos;i < pos + len;i++)
        Sub.ch[i - pos + 1] = S.ch[i];
    Sub.length = len;
    return true;
}
```

基本操作的实现

StrCompare(S,T)：比较操作。若S>T，则返回值>0；若S=T，则返回值=0；若S<T，则返回值<0。

S.ch="wangdao"
S.length=7

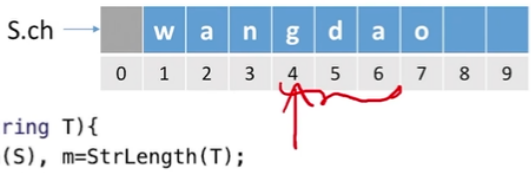

```
//比较操作。若S>T，则返回值>0；若S=T，则返回值=0；若S<T，则返回值<0
int StrCompare(SString S, SString T) {
    for (int i=1; i<=S.length && i<=T.length; i++){
        if (S.ch[i]!=T.ch[i])
            return S.ch[i]-T.ch[i];
    }
    //扫描过的所有字符都相同，则长度长的串更大
    return S.length-T.length;
}
```


```
1  //比较操作。若S > T，则返回值>0；若S = T，则返回值=0；若S < T，则返回值<0
2  int StrCompare(SString S,SString T)
3  {
4      for(int i = 1;i <= S.length && i <= T.length;i++)
5      {
6          if(S.ch[i] != T.ch[i])
7              return S.ch[i] - T.ch[i];
8      }
9      //扫描过的所有字符都相同，则长度长的串更大
10     return S.length - T.length;
11 }
```

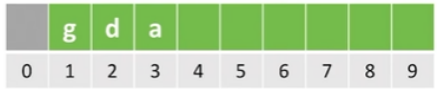
基本操作的实现

Index(S,T)：定位操作。若主串S中存在与串T值相同的子串，则返回它在主串S中第一次出现的位置；否则函数值为0。

S.ch="wangdao"
S.length=7




```
int Index(SString S, SString T){
    int i=1, n=StrLength(S), m=StrLength(T);
    SString sub;    //用于暂存子串
    while(i<=n-m+1){
        SubString(sub,S,i,m);
        if(StrCompare(sub, T)!=0)  ++i;
        else return i;  //返回子串在主串中的位置
    }
    return 0;   //S中不存在与T相等的子串
}
```




```
1  int Index(SString S,SString T)
2  {
3      int i = 1,n = StrLength(S),m = StrLength(T);
4      SString sub;                    //用于暂存子串
5      while(i <= n - m + 1)
6      {
7          SubString(sub,Si,m);
8          if(StrCompare(sub,T) != 0)
9              ++i;
```

```
        else
            return i;           //返回子串在主串中的位置
    }
    return 0;                   //S中不存在与T相等的子串
}
```

三、知识总结