字符串

主要内容

字符数组

字符串

指针与字符串

字符数组

字符数组的定义

与前述数组的定义相同,

如 **char c[10];**

字符数组的初始化

1）按单个字符进行

a[10]改为 a[9]?
观察

```
char a[10]={'c', ' ', 'p', 'r', 'o', 'g', 'r', 'a', 'm'};
```

c[0]	c[1]	c[2]	c[3]	c[4]	c[5]	c[6]	c[7]	c[8]	c[9]
c	␣	p	r	o	g	r	a	m	\0

2）按字符串进行

```
char c[11]={“I am happy”}
```

c[11]改为 c[10]?
观察

自动加一个′ \0′

字符数组的引用

例　输出一个字符数组。

程序如下：

```
#include <stdio.h>
void main()
{
   char c[10]={'I', ' ' ,'a','m','a',' ', 'b', 'o', 'y'};
   int i;
   for(i=0;i<10;i++)
      printf("%c",c[i]);
   printf("\n");
}
```

运行结果：I am a boy

printf(“%s” ,c) ;
输出字符串

字符串和字符串结束标志

为了测定字符串的实际长度，C语言规定了一个“字符串结束标志”——‘\0’。

字符数组并不要求它的最后一个字符为′\0′，甚至可以不包含′\0′。

但是由于系统对字符串常量自动加一个′\0′。因此，为了使处理方法一致，在字符数组中也常人为地加上一个′\0′ 例如：char c[6]={′C′,′h′,′i′,′n′,′a′,′\0′};

字符数组的输入输出

1) 逐个字符输入输出。如 用格式符“%c” 或

例如： char c[6]={"China"};

```
printf("%c%c%c%c%c", c[0],c[1],c[2],c[3],c[4]);
putchar(c[2]);
```

2) 将整个字符串一次输入或输出。如 用“%s”格式符，意思是对字符串的输入输出。

例如： char c[10];
scanf("%s", c);

以下程序的输出？

```
#include<stdio.h>
void main()
{
    char tmp[7] = {'C','h','i','n','a'};
    printf("%s" ,tmp);
}
```

以下程序的输出？

```
#include<stdio.h>
void main()
{
    char tmp[5] = {'C','h','i','n','a'};
    printf("%s" ,tmp);
}
```

说明：

用“%s”格式符输出字符数组(字符串)时

（1）如果数组长度大于字符串实际长度，也只输出到遇′ \0′结束。

（2）输出字符不包括结束符′ \0′ 。

（4）如果一个字符数组中包含一个以上′ \0′ ，则遇第一个′ \0′时输出就结束。

（5）可以用scanf函数输入一个字符数组(字符串)

如果利用一个**scanf**函数输入多个字符串，则在输入时以空格分隔。

例如：

```
char str1[5],str2[5],str3[5];
scanf("%s%s%s",str1,str2,str3);
```

输入数据：

How are you? 璇

数组中未被赋值的元素的值自动置′ \0′ 。

Str1	H	o	w	\0	\0
Str2	a	r	e	\0	\0
str3	y	o	u	?	\0

如果利用一个**scanf**函数输入多个字符串，则在输入时以空格分隔。

例如：

```
char str1[5],str2[5],str3[5];
scanf("%s%s%s",str1,str2,str3);
```

输入数据：

How are you? 凝

数组中未被赋值的元素的值自动置′ \0′ 。

Str1	H	o	w	\0	\0
Str2	a	r	e	\0	\0
str3	y	o	u	?	\0

注意：scanf函数中的输入项如果是字符数组名则不需要再加地址符&，因为在C语言中数组名代表该数组的起始地址。

分析图中所示的字符数组

输出数组c的起始地址和值

```
printf(" %d" , c); // 输出起始地址
printf(" %s" , c); // 输出字符串值
```


上机练习1

从键盘输入一个字符串（最大长度为50个字符），依次统计出大写字母、小写字母、数字和其他字符的个数。在屏幕上打印输入的字符串，且依次打印以上各种字符的个数以及字符串的总字符数

输入输出的方式：

(1)一个字符一个字符的输入输出 scanf(printf) (%c)

(2)整个字符串的输入输出scanf(printf) (%s)

字符串与指针

(1) 用字符数组存放一个字符串，然后输出该字符串。

```
#include <stdio.h>
void  main（）
 {
        char string［］＝{"I love China!"};
        printf("%s\n",string);
  }
```

(2) 用字符指针指向一个字符串。

可以不定义字符数组，而定义一个字符指针。用字符指针指向字符串中的字符。

例 定义字符指针

```
#include <stdio.h>
void  main（）
 {
        char*  string＝"I love China!";
        printf("%s\n", string);
 }
```

实例 将字符串 a 复制为字符串 b 。

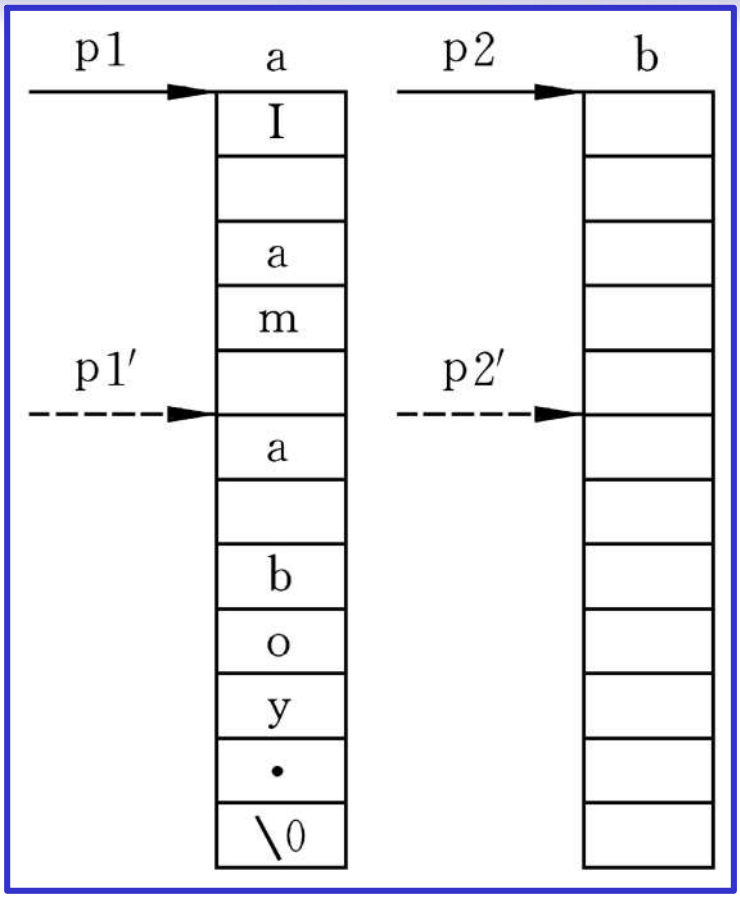
```
#include <stdio.h>
void main()
{
        char a[ ] = "I am a boy.", b[20];
        int i;
        for(i=0; *(a+i)! = '\0'; i++)
             *(b+i) = *(a+i);
        *(b+i)='\0';
        printf("string a is:%s\n",a);
        printf("string b is:");
        for(i=0;b[i]!='\0';i++)
                printf("%c",b[i]);
        printf("\n");
}
```

也可以设指针变量，用它的值的改变来指向字符串中的不同的字符。

例 指针变量来处理实例6 的字符串复制问题。

```
#include <stdio.h>
void main()
{
        char a[]="I am a boy.",b[20],*p1,*p2;
        int i;
        p1=a;p2=b;
        for(;*p1!='\0';p1++,p2++)
                *p2=*p1;
        *p2='\0';
        printf("string a is:%s\n",a);
        printf("string b is:");
        for(i=0;b[i]!='\0';i++)
                printf("%c",b[i]);
        printf("\n");

}
```

程序必须保证使 p 1 和 p 2 同步移动


p1
a
p2
b
I
a
m
p1′
p2′
a
b
o
y
•
\0

字符指针作函数参数

(1) 用字符数组作参数

实例7　用函数调用实现字符串的复制

```
void copy_string(char from[ ], char to[ ]);
void main()
{
        char a[ ]="I am a teacher.";
        char b[ ]="you are a student";
        printf("string a=%s\nstring b=%s\n",a,b);
        printf("copy string a to string b:\n");

        copy_string(a,b);

        printf("\nstring a=%s\nstring b=%s\n",a,b);
        return ;
}
```

程序运行结果如下：

```
string  a＝I am a teacher.
string  b ＝you are a student.
copy string a to string b:
string a ＝I am a teacher.
string b ＝I am a teacher.
```

```
void copy_string(char from[ ], char to[ ])
{
        int i=0;
        while(from[i]!='\0')
        {
                to[i]=from[i];
                i++;
        }
        to[i]='\0';
        return;
}
```

```
#include <stdio.h>
void copy_string（const char * from， char *to）;
void  main（）
{
  char from[ ]=″ I am a teacher. ″;
  char to[ ]=″ you are a student. ″;

  char * a ＝from;
  char * b ＝to;
  printf("string a=%s\n string b ＝%s\n″, a, b);
  printf("copy string a to string b:\n ");
  copy_string （a， b）;
   printf(“\nstring a=%s\n string b=%s＼n"， a， b);
}
```

```
void copy_string（const char *from，char * t o ）
{
    for（；*from！＝′ \ 0 ′ ；from＋＋，to＋＋）
        *to＝*from;
   *to＝′ \ 0 ′ ；
 }
```

•copy_string函数的函数体还可写成

```
{
    while（*from！＝′ \ 0 ′ ）
        *to++＝*from++;
   *to＝′ \ 0 ′ ；
 }
```

对使用字符指针变量和字符数组的讨论

字符数组和字符指针变量二者之间的区别：

(1) 字符数组由若干个元素组成，每个元素中放一个字符，而字符指针变量中存放的是地址（字符串第1个字符的地址），决不是将字符串放到字符指针变量中。

(2)赋值方式。对字符数组只能对各个元素赋值，不能用以下办法对字符数组赋值。

```
char  str[14];
str＝〃 I love China！〃；
```

而对字符指针变量，可以采用下面方法赋值：

```
char玏* a；
a ＝〃 I love China！〃；
```

(3)对字符指针变量赋初值：

char * a ＝″ I love China！″；等价于

char玳* a；

a ＝″ I love China！″；

而对数组的初始化：

char str［14］＝{″ I love China！″}；

不能等价于

char str［14］；

str［ ］＝″ I love China！″；

(4) 定义了一个字符数组，在编译时为它分配内存单元，它有确定的地址。而定义一个字符指针变量时，给指针变量分配内存单元，在其中可以放一个字符变量的地址。

例如：**char str［１０］；**

scanf（″％s″，str）；

字符串处理函数

1) puts函数 (<stdio.h>)

形式：**int puts(const char *_string_);** 作用：将一个以'\0'结束的字符串输出到终端。

例如：

```
char str[]={" China\nBeijing" };
puts(str);
```

输出结果：

China
Beijing

在输出时，将字符串结束标志'\0'转换成'\n'，即输出完字符串后换行。

2) gets函数 (<stdio.h>)

形式： **char *gets(char *buffer);** 作用：从终端输入一个字符串到字符数组，并且得到一个函数值。该函数值是字符数组的起始地址。

如执行下面的函数：

gets(str)

从键盘输入：

Computer↙ 凝 *将输入的字符串"Computer"送给字符数组str*

3) strcat函数

形式： char *strcat(char *strDestination, const char *strSource);

作用：把字符串strSource接到字符串strDestination的后面，结果放在字符串strSource中，函数调用后返回一个函数值——字符串strSource的地址。

例如：char str1［30］={″ People′ s Republic of ″ };
char str2[]={″ China″ };
printf(″ %s″ ， strcat(str1， str2));
输出结果： People′ s Republic of China

```
#include<string.h>
```

4） strcpy函数

一般形式： char *strcat(char **strDestination*, const char **strSource*);

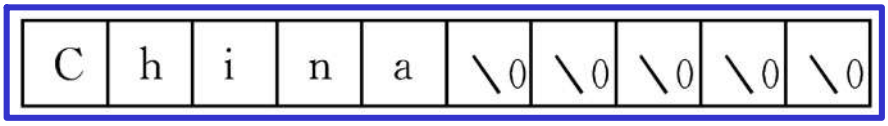
作用： 将字符串strSource复制到字符串strDestination中去。

例如：

char str1[10]，str2[]={″China″};

strcpy(str1，str2);

str1



关于strcpy函数的几点说明

1. “*strDestination*”必须定义得足够大，以便容纳被复制的字符串。

2. 复制时连同字符串后面的′ \0′一起复制。

关于strcpy函数的几点说明

4. strcpy的变形：strcpy(str1，str2，n); //将str2中前面n个字符复制到str1中去，然后再加一个‘\0’。

5. 只能用strcpy函数给一个字符串赋值，而不能用＝将一个字符串常量直接赋给另一个字符串。如：

（char *str2=″China″;） str1=str2; 

用strcpy函数能将一个字符串复制到另一个字符数组中去。
用赋值语句只能将一个字符赋给一个字符型变量或字符数组元素。

如： char a[6]，c1，c2;
strcpy(a,”China”); c1=′A′; c2=′B′; 
a[0]=‘C’; a[1]=‘h’; a[2]=‘i’; a[3]=‘n’; a[4]=‘a’;

5） strcmp函数

形式： int strcmp(const char *string1, const char *string2);

作用：比较字符串1和字符串2。

返回值：如果字符串1==字符串2，函数值为0。 如果字符串1>字符串2，函数值为一正整数。 如果字符串1<字符串2，函数值为一负整数

例如：strcmp(str1，str2);

strcmp(″ China″ ，″ Korea″);

strcmp(str1，″ Beijing″);

if(str1>str2) printf(″ yes″);

6) strlen函数

形式： size_t strlen(const char *_string_);

作用： 测试字符串长度的函数。函数的值为字符串中的实际长度(不包括′ \0′ 在内)。

例如：char str[10]={″ China″ };

printf(″ %d″ ， strlen(str));

输出结果不是10，也不是6，而是5。也可以直接测试字符串常量的长度，如strlen(″ China″);

上机练习2

1、上机练习1中的输入输出的方式用 gets (puts)

上机练习3

从键盘输入三个字符串，然后按从大到小的顺序输出这三个字符串

上机练习4

自己编写函数实现字符串的拷贝功能，并测试

char *Mystrcpy(char *_strDestination_, const char *_strSource_);

动态分配内存

```
void copy_string(const char *from, char *to);
void  main()
{
        char from[]="you are a student";

        char *a = from;
        char *b =(char *) malloc(strlen(a) + 1);

        printf("string a=%s\nstring b=%s\n",a,b);
        printf("copy string a to string b:\n ");

        copy_string(a, b);

        printf("\nstring a=%s\nstring b=%s\n",a,b);
        free(b);
        return;
}
```

```
void copy_string(const char *from, char *to);
void  main()
{
        char from[]="you are a student";

        char *a = from;
        char *b =(char *) malloc(strlen(a) + 1);

        printf("string a=%s\nstring b=%s\n",a,b);
        printf("copy stri

        copy_string(a, b);

        printf("\nstring a
        free(b);
        return;
}
```

```
string a=you are a student
string b=屯屯屯屯屯屯屯屯屯屯
copy string a to string b:

string a=you are a student
string b=you are a student
Press any key to continue
```

```
void copy_string(const char* from, char* to)
{
        for(;*from !='\0';from++,to++)
                *to = *from;

        *to='\0';
}
```

(C++)new和delete使用示例。

```
int main()
{
    int i, n;
    printf("请输入数组长度： ");
    scanf("%d",&n);
    int *p = new int[n]; (or int *p = (int *) malloc(n * sizeof(int)))   // 动态分配
    if ( p == NULL)
        return 1;
    printf("请输入 %d个元素： ", n);
    for (i=0; i<n; i++)
        scanf("%d",&p[i]);
    // 求和
    int sum = 0;
    for (i=0; i<n; i++)
        sum += p[i];
    printf("n个元素的和为：%d\n", sum);
    delete []p;     (free (p))
    return 0;
}
```

```
char * substring(const char * const s, int start, int end);
int main()
{
        char *s = "Chengdu, Sichuan";
        printf("%s\n",substring(s, 0, 7));
        return 0;
}
char * substring(const char * const s, int start, int end)
{
       char * pNewString = new char[end - start + 1 + 1];
        int j = 0;
        for (int i = start; i < end; i++, j++)
        {
            pNewString[j] = s[i];
        }
        pNewString[j] = '\0'; // Set a 0 terminator
        return pNewString;
}
```

改为空间足够大的数组怎样？

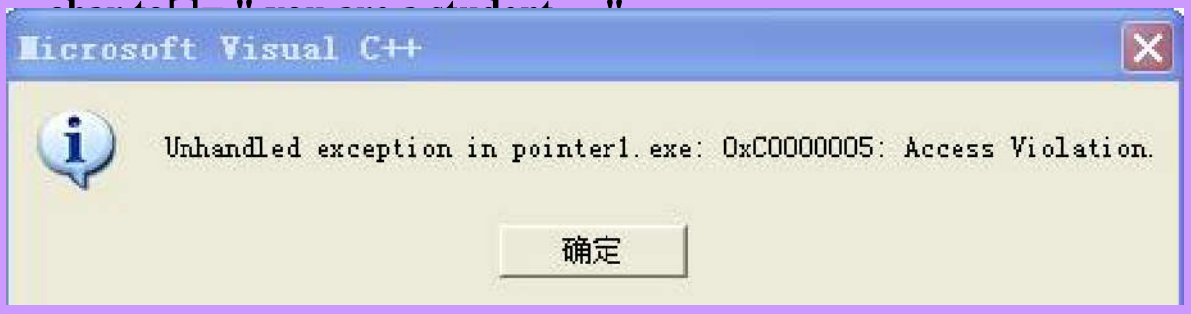
上机练习5

1. 用动态内存分配的方式，实现拷贝输入的字符串中的前面n个字符（提示：调用函数strncpy）
2. 修改前面的例子（数组-》上机练习3-》第4题），输入全班人数，动态分配内存用于存储全班同学的成绩，并将成绩从高到低输出，且输出平均分。

```
#include <stdio.h>
void copy_string（const char * from， char *to）;
void  main（）
{
        char from[ ]= " I  am a teacher. " ;
        char to[ ]= " you are a student.   " ;

        char * a ＝from;
        char * b ＝to;

        printf("string a=%s\n string  b ＝%s\n″ , a, b);

        printf("copy string a to string b:\n ");
        copy_string （ a ， b ）;

      printf(“\nstring a=%s\n string b=%s ＼ n"， a， b);
}
void copy_string(const char* from, char* to)
{
          for(;*from !='\0';from++,to++)
                    *to = *from;
          *to='\0';
}
```

```
#include <stdio.h>
void copy_string（const char * from， char *to）;
void  main（）
{
        char from[ ]= " I  am a teacher. " ;
        char to[ ]= " you are a student.   " ;

        char * a ＝from;
        char * b ＝to;

        printf("string a=%s\n string  b ＝%s\n″ , a, b);

        printf("copy string a to string b:\n ");
        copy_string （b，a）;

      printf(“\nstring a=%s\n string b=%s ＼ n"， a，b);
}
void copy_string(const char* from, char* to)
{
          for(;*from !='\0';from++,to++)
                    *to = *from;
          *to='\0';
}
```

```
#include <stdio.h>
void copy_string（const char * from， char *to）;
void  main（）
{
        char from[ ]= " I  am a teacher. " ;
```




```
        printf("copy string a to string b:\n ");
        copy_string （b，a）;

      printf(“\nstring a=%s\n string b=%s \ n"， a，b);
}
void copy_string(const char* from, char* to)
{
            for(;*from !='\0';from++,to++)
                        *to = *from;
            *to='\0';
}
```