C++语言基础——char 类型字符串

一、 字符类型

计算机程序处理的数据类型也可以概括的分为数值型数据和非数值型数据两大类，前面学习的内容多以数值型数据的运算或处理为主，接下来我们开始了解一种最基本的非数值型数据的处理知识——字符类型；

字符类型为由一个单个字符组成的数据。通常使用一对单引号将单个字符括起来的表示形式，如’a’ 、 ‘ ’ 、 ‘.’ 等。

程序设计中常用到的是字符常量或字符变量。

1、字符常量定义：

const 字符常量标识符=’单个字符’；

例如：const char ch1=’A’；

说明：本语句说明或定义了一个字符类型的常量，该常量表示的字符数据为 ‘A’

字符变量是用来存放单个字符的变量，即只能表示一个单个字符；

2、字符变量定义：

char 字符变量标识符;

例如：char ch2,ch2;

表示定义了两个字符变量 ch1 和 ch2，每个变量进可以表示或者存放一个字符，因此可以利用如下语句赋值：

ch1=’a’ ; ch2=’b’ ; ch1=’ ’

也可以相互赋值 ch2=ch1;

3、字符与整数的通用性（对应性）关系

前面我们知道，字符数据在内存中是以该字符对应的 ASCII 编码存储，也即是说每

一个常见字符在内存中都与一个唯一的序值一一对应。

在实际程序设计中，一个字符数据既可以以字符形式输出，也可以以整数（ASCII表中的序值）形式输出，还可以对它们进行算术运算。

ASCII码值	字符	ASCII码值	字符
0	nul（空）	65	A
32	space（空格）	90	Z
48	0	97	a
57	9	122	z

二、 字符数组

我们前面学习数组时已经知道:无论数组的元素有多少个，数组中每个元素的类型必须相同，而且数组元素的类型可以是任何已定义的数据类型，如整形、字符型、实型等。当一个数组的元素类型为字符型时，我们称这个数组为字符数组。字符数组的每一个元素都是一个字符。(连续的单个字符)

字符数组是计算机非数值处理的基本实现方式之一

例如：

```
char c[5];
c[0]='H';c[2]='e';c[3]='l';c[4]='l';c[5]='o';
```

其在内存中的存储形式如图所示：

H	e	l	l	o

三、 字符串

字符串是一对双引号括起来的字符序列（**字符组成的整体**）

如”Hello world!” 、 “a b c d e f g ”、” a” 、” ” 等。

特别的是，每个字符串的结尾都有一个“字符串结束标志”，以便系统据此判断字符串是否结束。C++语言中用字符’**\0**’作为字符串结束的标志，’\0’是一个 ASCII

码为 0 的字符，占用一个字节，例如有一个字符串”Hello”，在内存里的实际存储如下图所示：

说明：这个字符串占用的不是 5 个字节，而是 6 个字节，最后一个字节里面存放的是字符串结束标志’\0’。但是在输出时不会输出’\0’。

四、**字符常量和字符串常量区别**

①两者的定界符不同，字符常量由单引号括起来，字符串常量由双引号括起来。

②字符常量只能是单个字符，字符串常量则可以是多个字符。

③可以把一个字符常量赋给一个字符变量，但不能把一个字符串常量赋给一个字符变量。

④字符常量占一个字节，而字符串常量占用字节数等于字符串的字节数加 1。增加的一个字节中存放字符串结束标志 ‘\0’。例如：字符常量 ‘a’ 占一个字节，字符串常量 “a” 占二个字节。

五、字符数组的定义与赋值

（1）字符数组的定义格式

字符数组定义格式同一般数组，所不同的是数组元素类型是字符型，第一个元素同样是从 ch1[0]开始，而不是 ch1[1]。具体格式如下：

char 数组名[常量表达式 1]…

例如：char ch1[5]; //数组 ch1 是一个具有 5 个字符元素的一维字符数组

（2）字符数组的赋值

字符数组赋值类似于一维数组，赋值分为数组的初始化和数组元素的赋值。初始化的方式有用字符初始化和用字符串初始化两种，也有用初始值表进行初始化的。

1）用字符初始化数组

例如： char chr1[5]={ ‘a’ , ‘b’ , ‘c’ , ‘d’ , ‘e’ };

初始值表中的每个数据项是一个字符，用字符给数组 chr1 的各个元素初始化。当初始值个数少于元素个数时，从首元素开始赋值，剩余元素默认为空字符。

（3）字符串赋值

字符数组中也可以存放若干个字符，也可以来存放字符串。两者的区别是字符串有一结束符（‘\0’）。反过来说，在一维字符数组中存放着带有结束符的若干个字符可以称为字符串。字符串是一维字符数组，但是一维字符数组不等于字符串。

例如：

char chr2[5]={‘a’，‘b’，‘c’，‘d’，‘\0’}; 即在数组 chr2 中存放着一个字符串“abcd”。

2）用字符串初始化数组

用一个字符串初始化一个一维字符数组，可以写成下列形式：

char chr2[5]=“abcd”；

使用此格式均要注意字符串的长度应小于字符数组的大小或等于字符数组的大小减 1

3）字符数组元素赋值

字符数组的赋值是给该字符数组的各个元素赋一个字符值。

例如：

char chr[3];

chr[0]=‘a’；

chr[1]=‘b’；

chr[2]=‘c’；

六、字符串的基本操作

（1）字符串的输入与输出

字符串可以作为一维字符数组来处理，那么字符串的输入和输出也可以按照数组元素来处理，本节不再做介绍。本节仅介绍将字符串作为一个整体进行输入和输出的语句。

（1）输入

从键盘输入一个字符数组可以使用 scanf 语句或 gets 语句。

1) scanf 语句

格式：scanf(“%s”,字符串名称);

说明：

①这里的字符串名称之前不加&这个取地址符。例如：scanf(“%s”,&s1)是错误的。

②系统会自动在输入的字符串常量后添加‘\0’标志，因此输入时，仅输入字符串的内容即可。

③输入多个字符串时，以空格分隔。

```
#include<cstring>
#include<cstdio>
using namespace std;
char a[10],b[10],c[10];
int main(){
    scanf("%s %s %s",a,b,c);
    printf("%s %s %s",a,b,c);
    return 0;
}
```

2) gets 语句 整行读入

格式：gets(字符串名称)；

说明：使用 gets 只能输入一个字符串。

例如：gets(s1,s2)；是错误的。使用 gets，是从光标开始的地方读到换行符也就是说读入的是一整行，而使用 scanf 是从光标开始的地方到空格，如果这一行没有空格，才读到行尾。

例如：scanf(“%s”,s1)；gets(s2)；对于相同的输入 Hello World!。s1 获取的结果仅仅是 Hello，而 s2 获取的结果则是 Hello World!

（2）字符串的输出

向屏幕输出一个字符串可以使用 printf 语句或 puts 语句。

1) printf 语句

格式：printf(“%s”,字符串名称);

说明：

①用%s 格式输出时，printf 的输出项只能是字符串(字符数组)名称，而不能是数组元素。例如：printf(“%s”,a[5]);是错误的。可以是 printf(“%s”,a);

②输出的内容不包括字符串结束标志符 ‘\0’。

2) puts 语句

格式：puts(字符串名称);

说明：puts 语句输出一个字符串和一个换行符。对于已经声明过的字符串 a，printf(“%s\n”,a)和 puts(a)是等价的。

七、常见 char 字符串函数

函数格式	函数功能
strcat(st1,st2)	将 st2 连接到 st1 后边，返回 st1 的值。(+)
strncat(st1,st2,n)	将 st2 前 n 个字符连接到 st1 后边，返回 st1 的值。(有选择的+)
strcpy(st1,st2)	将 st2 复制到 st1 中，返回 st1 的值。(赋值)
strncpy(st1,st2,n)	将 st2 前 n 个字符复制到 st1 中，返回 st1 的值。(前 n 个字符复制替换，其他不影响)

strcmp(st1,st2)	比较 st1 和 st2 的大小，比较的结果由函数带回； 如果 st1>st2，返回一个正整数； 如果 st1=st2，返回 0； 如果 st1<st2，返回一个负整数； （首先逐位按照字符 ASCII 码值比较，如果对应位都相同，则长串大）
strncmp(st1,st2,n)	比较 st1 和 st2 的前 n 个字符进行比较，函数返回值的情况同 strcmp 函数；
strlen(st)	计算 st 的长度，终止符‘\0’不算在长度之内
strlwr(st)	将 st 中大写字母换成小写字母
strupr(st)	将 st 中小写字母换成大写字母

【例 1】输出字母表

【问题描述】：按字母表顺序和逆序每隔一个字母打印。即打印出：

a c e g i k m o q s u w y

z x r v t p n l j h f d b

【问题分析】：利用了字符类型是顺序类型这一特性，灵活利用字符变量当作循环控制变量，使程序处理起来比较直观

```
#include<cstdio>
#include<cstring>
using namespace std;
int main(){
    for(char ch='a';ch<='z';ch+=2)
        printf("%c ",ch);
    printf("\n");
```

```
    for(char ch='z';ch>='a';ch-=2)
        printf("%c ",ch);
    return 0;
}
```

【例 2】**输出字符串**

【问题描述】：输入一个字符串，分别输出字符串中的每一个字符

【问题分析】：可以充分利用字符数组和字符串的相似性，以串的形式读入，以字符数组形式对其中的某些字符进行引用、表示、处理，本例即体现了一个字符串中的字符可以通过其对应的下标灵活使用特点。

```
#include<cstdio>
#include<cstring>
using namespace std;
int main(){
    char st[100];
    gets(st);
    for(int i=0;i<strlen(st);i++)
        printf("%c",st[i]);
    return 0;
}
```

【例 3】**数字统计**

【问题描述】

请统计某个给定范围[L, R]的所有整数中，数字 2 出现的次数。

比如给定范围[2, 22] ，数字 2 在数 2 中出现了 1 次，在数 12 中出现 1 次，在数 20 中出现 1 次，在数 21 中出现 1 次，在数 22 中出现 2 次，所以数字 2 在该范围内一共出现了 6 次。

【数据范围】

1 ≤L ≤R ≤10000

【算法分析】

枚举[L,R]区间的所有整数，对于每个整数 x：

1.将整数 x 转化成字符串 s，可以用 sprintf(s,"%d",x)来实现；

2.枚举字符串 s 的每个字符判断是否为 2。

```
#include<cstdio>
#include<cstring>
using namespace std;
int main(){
    char st[100];
    int i,j,l,r,len,ans=0;
    scanf("%d %d",&l,&r);
    for(i=l;i<=r;i++){
        sprintf(st,"%d",i);
        sprintf(st,"%d",i);
        len=strlen(st);
        for(j=0;j<len;j++)
            if(st[j]=='2')
                ans++;
    }
    printf("%d",ans);
    return 0;
}
```

【例 4】**数字反转**

【问题描述】

给定一个整数，请将该数各个位上数字反转得到一个新数。新数也应满足整数的常见形式，即除非给定的原数为零，否则反转后得到的新数的最高位数字不应为零

【数据范围】

-1,000,000,000 ≤ N≤ 1,000,000,000。

【算法分析】

1.将整数 N 转化成字符串 s，可以用 sprintf(s,"%d",N)来实现；

2.对字符串进行反转操作；

3.将字符串 s 转换成数字 N,可以用 sscanf(s,"%d",&N)来实现。

4.输出数字 N。

```
#include <cstdio>
#include <cstring>
using namespace std;
char s[100],c[100];
int main(){
      int n,l;
      scanf("%d",&n);
      sprintf(s,"%d",n);
      l=strlen(s);
      for (int i=0; i<=l-1; ++i)
          c[l-i-1]=s[i];
      if(n<0)
         printf("-");
      sscanf(c,"%d",&n);
      printf("%d",n);
      return 0;
}
```

【例 5】**回文字符串**

输入一个字符串，输出该字符串是否回文。回文是指顺读和倒读都一样的字符串。

输入:

输入为一行字符串（字符串中没有空白字符，字符串长度不超过 100)。

输出:

如果字符串是回文，输出 yes；否则，输出 no。

样例输入:

abcdedcba

样例输出:

yes

【问题分析】

设置三个变量表示：i 字串的头指针，其从字串的第一个字符依次后移，其值依次增大；j 字串尾指针，其从字串尾向前依次前移，其值逐渐减小；s 标志其是否为回文串，其初始值为 true，一旦发现字串从首尾开始相对的字符不一致了，其值就变化为 false，最后根据 s 的取值即可判断是否为回文串

```
# include <cstdio>
# include <cstring>
using namespace std;
char a[101];
int main(){
    int i=0,j;
    bool f=true;
    gets(a);
    j=strlen(a)-1;
    while (i<j && s==true){
        if (a[i]!=a[j]) s=false;
        i++;
        j--;
    }
    if (s==true) printf("yes\n");
    else printf("no\n");
    return 0;
}
```