第26章　字符串与字符函数

标准函数库含有丰富的字符串和字符处理函数。字符串函数操作以null结束的字符数组并需要头文件<cstring>，而字符函数则使用头文件<cctype>。C程序必须使用头文件string.h和ctype.h。

因为C/C++在进行数组操作时没有边界检查功能，所以防止发生数组越界就成了程序员的责任，忽视了这个工作有可能导致程序崩溃。

在C/C++中，一个可打印的字符是指可以在终端上显示的字符。这些字符通常在空格符(0x20)和～字符(0xFE)之间。控制字符的值在(0)和(0x1F)之间，此外还包括DEL (0x7F)。

由于一些历史原因，这些字符函数的参数都是整型的，但是只有低位字节被使用，字符函数自动将它们的参数转换为unsigned char型。然而，我们可以自由地用字符参数调用这些函数，这是因为这些字符在调用时将自动转换为整型。

头文件<cstring>定义了size_t类型，这种类型本质上与unsigned相同。

本章只介绍那些对char类型的字符进行操作的函数。这些函数最初由标准C和C++定义并且到目前为止一直被最广泛地使用和支持。操作wchar_t型字符的宽字符函数将在第31章讨论。

26.1　isalnum函数

```
#include <cctype>
int isalnum(int ch);
```

如果isalnum( )函数的参数是字母表中的字符或是一个数字，该函数将返回一个非0值；如果字符不是字母或数字，函数将返回0。

与isalnum( )相关的函数有isalpha( )，iscntrl( )，isdigit( )，isgraph( )，isprint( )，ispunct( )和isspace( )。

26.2　isalpha函数

```
#include <cctype>
int isalpha(int ch);
```

如果ch是字母表中的字母，isalpha( )函数将返回一个非0值；否则，函数将返回0。组成字母表的字符根据不同的语言而有所不同，对英语来说，这些字符是A~Z的大小写字母。

与isalpha( )相关的函数有isalnum( )，iscntrl( )，isdigit( )，isgraph( )，isprint( )，ispunct( )和isspace( )。

26.3　iscntrl函数

```
#include <cctype>
int iscntrl(int ch);
```

如果ch在0和0x1F之间或者是等于0x7F (DEL)，iscntrl( )函数将返回一个非0值；否则，将返回0。

与iscntrl( )相关的函数有isalnum( )，isalpha( )，isdigit( )，isgraph( )，isprint( )，ispunct( )和isspace( )。

26.4 isdigit函数

```
#include <cctype>
int isdigit(int ch);
```

如果ch是一个数字（即0～9的数），isdigit( )函数将返回一个非0值；否则，将返回0。

与isdigit( )相关的函数有isalnum( )，isalpha( )，iscntrl( )，isgraph( )，isprint( )，ispunct( )和isspace( )。

26.5 isgraph函数

```
#include <cctype>
int isgraph(int ch);
```

如果ch是不包括空格在内的任何可打印字符，isgraph( )函数将返回一个非0值；否则，将返回0。这些字符一般在0x21～0x7E的范围内。

与isgraph( )相关的函数有isalnum( )，isalpha( )，iscntrl( )，isdigit( )，isprint( )，ispunct( )和isspace( )。

26.6 islower函数

```
#include <cctype>
int islower(int ch);
```

如果ch是一个小写字母，islower( )函数将返回一个非0值；否则，将返回0。

与islower( )相关的函数是isupper( )。

26.7 isprint函数

```
#include <cctype>
int isprint(int ch);
```

如果ch是一个包括空格在内的可打印字符，isprint( )函数将返回一个非0值；否则，将返回0。可打印字符通常在0x20～0x7E的范围内。

与isprint( )相关的函数有isalnum( )，isalpha( )，iscntrl( )，isdigit( )，isgraph( )，ispunct( )和isspace( )。

26.8 ispunct函数

```
#include <cctype>
int ispunct(int ch);
```

如果 ch 是一个标点符号，ispunct( )函数将返回一个非 0 值；否则，将返回 0。这个函数定义的“标点符号”包括既不是字母和数字、也不是空格的所有可打印字符。

与 ispunct( )相关的函数有 isalnum( )，isalpha( )，iscntrl( )，isdigit( )，isgraph( )和 isspace( )。

26.9　isspace 函数

```
#include <cctype>
int isspace(int ch);
```

如果 ch 是一个空格、横向制表符、纵向制表符、换页符、回车和换行符，isspace( ) 函数将返回一个非 0 值；否则，将返回 0。

与 isspace( )相关的函数有 isalnum( )，isalpha( )，iscntrl( )，isdigit( )，isgraph( )和 ispunct( )。

26.10　isupper 函数

```
#include <cctype>
int isupper(int ch);
```

如果 ch 是一个大写字母，isupper( )函数将返回一个非 0 值；否则，将返回 0。

与 isupper( )相关的函数是 islower( )。

26.11　isxdigit 函数

```
#include <cctype>
int isxdigit(int ch);
```

如果 ch 是一个十六进制的数字，isxdigit( )函数将返回一个非 0 值；否则，将返回 0。一个十六进制数的范围是：A ~ F、a ~ f 或 0 ~ 9。

与 isxdigit( )相关的函数有 isalnum( )，isalpha( )，iscntrl( )，isdigit( )，isgraph( )，ispunct( )和 isspace( )。

26.12　memchr 函数

```
#include <cstring>
void *memchr(const void *buffer, int ch, size_t count);
```

memchr( )函数在 buffer 指定的数字中搜索前 count 个字符中第一次出现 ch 的位置。memchr( )函数返回一个指向 buffer 中第一次出现 ch 处的指针，如果没有发现 ch，则返回一个空指针。

与 memchr( )相关的函数有 memcpy( )和 isspace( )。

26.13　memcmp 函数

```
#include <cstring>
int memcmp(const void *buf1, const void *buf2, size_t count);
```

memcmp( )函数对 buf1 和 buf2 所指的数组的前 count 个字符进行比较。

memcmp( )函数返回一个整型值，其说明如下：

值	含义
小于0	buf1小于buf2
0	buf1等于buf2
大于0	buf1大于buf2

与memcmp( )相关的函数有memchr( )，memcpy( )和strcmp( )。

26.14 memcpy函数

```
#include <cstring>
void *memcpy(void *to, const void *from, size_t count);
```

memcpy( )函数把from所指的数组的count个字符复制到to所指的数组中。如果数组发生重叠，memcpy( )的行为将是不确定的。

memcpy( )函数返回一个指向to的指针。

与memcpy( )相关的函数是memmove( )。

26.15 memmove函数

```
#include <cstring>
void *memmove(void *to, const void *from, size_t count);
```

memmove( )函数把from所指的数组的count个字符复制到to所指的数组中。如果数组发生重叠，复制将以正确的方式进行，函数将正确的内容放入to，但将更改from。memmove( )函数返回一个指向to的指针。

与memmove( )相关的函数是memcpy( )。

26.16 memset函数

```
#include <cstring>
void *memset(void *buf, int ch, size_t count);
```

memset( )函数把ch的低位字节复制到buf所指数组的前count个字符中。

memset( )最常见的用途是把内存中的某个区域初始化为某个已知的值。

与memset( )相关的函数有memcmp( )，memcpy( )和memmove( )。

26.17 strcat函数

```
#include <cstring>
char *strcat(char *str1, const char *str2);
```

strcat( )把str2的一个副本连接到str1并用一个null作为str1的结束符。原来作为str1结束符的null被str2的第一个字符重写。该操作不会改变str2的内容。如果发生数组重叠，strcat( )的行为将是不确定的。

strcat( )函数返回str1。

记住，因为这个函数不进行边界检查，所以为了确保str1的长度足以存放其初始内容和str2的内容，编程人员应进行边界检查。

与 strcat( )相关的函数有 strchr( )，strcmp( )和 strcpy( )。

26.18　strchr 函数

```
#include <cstring>
char *strchr(const char *str, int ch);
```

strchr( )函数返回一个指针，该指针指向str 所指的字符串中第一次出现ch 低位字节的位置。如果没有发现匹配的内容，函数返回一个空指针。

与 strchr( )相关的函数有 strpbrk( )，strspn( )，strstr( )和 strtok( )。

26.19　strcmp 函数

```
#include <cstring>
int strcmp(const char *str1, const char *str2);
```

strcmp( )函数根据词典编辑法比较两个字符串并根据以下结果返回一个整型值：

值	含义
小于 0	str1 小于 str2
0	str1 等于 str2
大于 0	str1 大于 str2

与 strcmp( )相关的函数有 strchr( )，strcpy( )和 strcmp( )。

26.20　strcoll 函数

```
#include <cstring>
int strcoll(const char *str1, const char *str2);
```

strcoll( )函数将 str1 指定的字符串和 str2 指定字符串进行比较。这种比较与 setlocale( )函数（详见 setlocale）指定的规则相一致。

strcoll( )函数返回一个整型值，其说明如下：

值	含义
小于 0	str1 小于 str2
0	str1 等于 str2
大于 0	str1 大于 str2

与 strcoll( )相关的函数有 memcmp( )和 strcmp( )。

26.21　strcpy 函数

```
#include <cstring>
char *strcpy(char *str1, const char *str2);
```

strcpy( )函数把str2的内容复制到str1。str2必须是一个指向以null结束的字符串的指针。该函数返回一个指向 str1 的指针。

如果 str1 和 str2 发生重叠，strcpy( )的行为将是不确定的。

与 strcpy( )相关的函数有 memcpy( )、strchr( )、strcmp( )和 strncmp( )。

26.22 strcspn 函数

```
#include <cstring>
size_t strcspn(const char *str1, const char *str2);
```

strcspn( )函数返回 str1 指定的字符串的初始子串长度，该子串只由不包含在 str2 所指的字符串中的字符组成。换句话说，strcspn( )返回 str1 所指的字符串中第一个与 str2 所指的字符串中任何字符相匹配的字符索引。

与 strcspn( )相关的函数有 strrchr( )，strpbrk( )，strstr( )和 strtok( )。

26.23 strerror 函数

```
#include <cstring>
char *strerror(int errnum);
```

strerror( )函数返回一个指向与errnum值相关联的由实现定义的字符串的指针。该字符串不能被修改。

26.24 strlen 函数

```
#include <cstring>
size_t strlen(const char *str);
```

strlen( )函数返回 str 所指的以 null 结束的字符串长度，该长度不包括作为结束符的 null。

与 strlen( )函数相关的函数有 memcpy( )，strchr( )，strcmp( )和 strncmp( )。

26.25 strncat 函数

```
#include <cstring>
char *strncat(char *str1, const char *str2, size_t count);
```

strncat( )函数把 str2 所指的字符串中数量不超过 count 的字符连接到 str1 所指的字符串中并用一个 null 结束 str1。最初结束 str1 的 null 终止符被 str2 的首字符覆盖。该操作不改变 str2 所指的字符串。如果字符串发生重叠，行为将是不确定的。

strncat( )函数返回 str1。

记住，该函数不进行边界检查，所以为了确保 str1 的长度足以存放其初始内容和 str2 的内容，编程人员应进行边界检查。

与 strncat( )相关的函数有 strcat( )，strnchr( )，strncmp( )和 strncpy( )。

26.26 strncmp 函数

```
#include <cstring>
int strncmp(const char *str1, const char *str2, size_t count);
```

strncmp( )函数根据词典编辑法比较两个以 null 结束的字符串中数量不超过 count 的字符，然后根据下面的结果返回一个整型值：

值	含义
小于 0	str1 小于 str2
0	str1 等于 str2
大于 0	str1 大于 str2

如果两个字符串中任何一个字符串的字符数小于 count，那么当遇到 null 时，比较操作结束。

与 strncmp( )相关的函数有 strcmp( )，strnchr( )和 strncpy( )。

26.27　strncpy 函数

```
#include <cstring>
char *strncpy(char *str1, const char *str2, size_t count);
```

strncpy( )函数把 str2 所指的字符串中的 count 个字符复制到 str1 所指的字符串中。str2 必须是指向一个以 null 结束的字符串的指针。

如果 str1 和 str2 重叠，strncpy( )的行为将是不确定的。

如果 str2 所指的字符串中的字符数小于 count，函数将把一些 null 添加到 str1 的末尾，直到 count 个字符被复制。

如果 str2 所指的字符串中的字符数大于 count，str1 所指的结果字符串将不以 null 结束。

strncpy( )函数返回一个指向 str1 的指针。

与 strncpy( )相关的函数有 memcpy( )，strchr( )，strncat( )和 strncmp( )。

26.28　strpbrk 函数

```
#include <cstring>
char *strpbrk(const char *str1, const char *str2);
```

strpbrk( )函数返回一个指向字符的指针，该字符在 str1 所指的字符串中，是第一个与 str2 所指的字符串中任何一个字符相匹配的字符，null 终止符不包括在内。如果没有找到匹配的字符，函数将返回一个空指针。

与 strpbrk( )相关的函数有 strspn( )，strrchr( )，strstr( )和 strtok( )。

26.29　strrchr 函数

```
#include <cstring>
char *strrchr(const char *str, int ch);
```

strrchr( )函数返回一个指向 ch 的低位字节的最后一个位置的指针，该 ch 在 str 中。如果没有找到匹配字符，函数返回一个空指针。

与 strrchr( )相关的函数有 strpbrk( )，strspn( )，strstr( )和 strtok( )。

26.30　strspn 函数

```
#include <cstring>
size_t strspn(const char *str1, const char *str2);
```

strspn( )函数返回str1指定的字符串的初始子串的长度，该子串只由包含在str2 指定的字符串中的字符组成。换句话说，strspn( )返回str1所指的字符串中与str2所指的字符串中任何字符不匹配的第一个字符的索引。

与strspn( )相关的函数有strpbrk( )，strrchr( )，strstr( )和strtok( )。

26.31 strstr 函数

```
#include <cstring>
char *strstr(const char *str1, const char *str2);
```

strstr( )函数返回一个指针，该指针指向str2所指的字符串中第一次出现str1所指字符串的位置。如果没有发现匹配字符串，函数返回一个空指针。

与strstr( )相关的函数有strchr( )，strcspn( )，strpbrk( )，strspn( )，strtok( )和strrchr( )。

26.32 strtok 函数

```
#include <cstring>
char *strtok(char *str1, const char *str2);
```

strtok( )函数返回一个指向str1所指字符串中下一个标记（token）的指针。构成str2所指字符串的字符是确定该标记的定界符。当没有标记返回时，函数返回一个空指针。

为了标记一个字符串，第一次调用strtok( )时str1所指的字符串必须被加上标记，其后的调用必须将一个空指针用于str1。这样整个字符串就可以精简为它的标记。

每次调用strtok( )时可以使用一组不同的定界符。

与strtok( )相关的函数有strchr( )，strcspn( )，strpbrk( )，strrchr( )和strspn( )。

26.33 strxfrm 函数

```
#include <cstring>
size_t strxfrm(char *str1, const char *str2, size_t count);
```

strxfrm( )函数对str2所指的字符串进行转换，以使它可以被strcmp( )函数所用，该函数还将转换结果放入str1所指的字符串中。转换之后，使用str1的strcmp( )产生的结果和使用str2所指的初始字符串的strcoll( )产生的结果是相同的，没有数量多于count的字符被写到str1所指的字符串。

strxfrm( )函数返回被转换的字符串长度。

与strxfrm( )相关的函数有strcoll( )。

26.34 tolower 函数

```
#include <cctype>
int tolower(int ch);
```

如果ch是一个字母，tolower( )函数返回与其相对应的小写字母；否则，返回ch。

与tolower( )相关的函数有toupper( )。

26.35　toupper 函数

```
#include <cctype>
int toupper(int ch);
```

如果 ch 是一个字母，toupper( )函数返回与其相对应的大写字母；否则，返回 ch。

与 toupper( )相关的函数有 tolower( )。