第28章　时间、日期和定位函数

标准函数库定义了一些处理日期和时间的函数，还定义了一些处理与程序相关的地理信息的函数。本章将介绍这些函数。

时间和日期函数需要头文件<ctime>（C程序必须使用头文件time.h）。这个头文件定义了3种与时间有关的类型：clock_t、time_t和tm。类型clock_t和time_t可以把系统时间和系统日期表示为某种整型数，我们称之为日历（calendar）时间；结构类型tm则把日期和时间分解为相应的元素来存放。tm结构的定义如下所示：

```
struct tm {
  int tm_sec;   /* seconds, 0-61 */
  int tm_min;   /* minutes, 0-59 */
  int tm_hour;  /* hours, 0-23 */
  int tm_mday;  /* day of the month, 1-31 */
  int tm_mon;   /* months since Jan, 0-11 */
  int tm_year;  /* years from 1900 */
  int tm_wday;  /* days since Sunday, 0-6 */
  int tm_yday;  /* days since Jan 1, 0-365 */
  int tm_isdst /* Daylight Saving Time
                  indicator */
}
```

如果实行夏令时，tm_isdst的值为正；如果不是夏令时，该值为0；如果没有可用信息，该值为负。这种时间和日期形式称为分解（broken-down）时间。

此外，<ctime>定义了一个称为CLOCKS_PER_SEC的宏，这是每秒的系统时钟滴数。

地理环境函数需要头文件<clocale>（C程序必须使用头文件locale.h）。

28.1　asctime函数

```
#include <ctime>
char *asctime(const struct tm *ptr);
```

asctime()函数返回一个指向字符串的指针，该字符串包含存储在ptr所指的结构中的信息，它将被转换为以下形式：

day month date hours:minutes:seconds year\n\0

例如：

```
Fri Apr 15 12:05:34 2005
```

通常情况下，被传递给asctime()的结构指针既可以从localtime()获得，也可以从gmtime()获得。

asctime()用来存放格式输出字符串的缓冲区是一个静态分配的字符数组，它在每次调用该

函数时被重写。如果想要保存该字符串的内容，必须把它复制到其他地方。

与 asctime( )相关的函数有 localtime( )，gmtime( )，time( )和 ctime( )。

28.2　clock 函数

```
#include <ctime>
clock_t clock(void);
```

clock( ) 函数返回一个近似值，该值说明调用程序已经运行了多长时间。为了把这个值转换成秒，可以用它除以 CLOCKS_PER_SEC。如果得不到这个时间，函数返回 -1。

与 clock( )相关的函数有 time( )，asctime( )和 ctime( )。

28.3　ctime 函数

```
#include <ctime>
char *ctime(const time_t *time);
```

假设指针指向日历时间、ctime( )函数将返回一个指向以下形式的字符串的指针：

day month year hours:minutes:seconds year\n\0

日历时间通常可以通过调用 time( )获得。

ctime( )用来存放格式输出字符串的缓冲区是一个静态分配的字符数组，它在每次调用该函数时被重写。如果想要保存该字符串的内容，必须把它复制到其他地方。

与 ctime( )相关的函数有 localtime( )，gmtime( )，time( )和 asctime( )。

28.4　difftime 函数

```
#include <ctime>
double difftime(time_t time2, time_t time1);
```

difftime( )函数返回 time1 与 time2 的差（以秒为单位），也就是说，返回 time2 -time1。

与 difftime( )相关的函数有 localtime( )，gmtime( )，time( )和 asctime( )。

28.5　gmtime 函数

```
#include <ctime>
struct tm *gmtime(const time_t *time);
```

gmtime( )函数返回一个指向过时的 time 形式的指针，这个 time 具有 tm 结构形式。该时间用 UTC (Coordinated Universal Time)表示，它实际上是格林尼治（Greenwich）平均时间。time 的值通常可以通过调用 time( )获得。如果系统不支持 UTC，函数将返回 NULL。

gmtime( )用来存放过时的时间形式的结构是静态分配的并且在每次调用函数时被重写。如果想保存该结构的内容，必须将其复制到其他地方。

与 gmtime( )相关的函数有 localtime( )，time( )和 asctime( )。

28.6 localeconv函数

```
#include <clocale>
struct lconv *localeconv(void);
```

localeconv( )函数返回一个指向lconv类型的结构的指针，该结构包含与数字的格式化方式有关的各种地理环境信息。lconv结构的组织如下所示：

```
struct lconv {
  char *decimal_point;       /* decimal point character
                                for nonmonetary values */
  char *thousands_sep;       /* thousands separator
                                for nonmonetary values */
  char *grouping;            /* specifies grouping for
                                nonmonetary values */
  char *int_curr_symbol;     /* international currency symbol */
  char *currency_symbol;     /* local currency symbol */
  char *mon_decimal_point;   /* decimal point character for
                                monetary values */
  char *mon_thousands_sep;   /* thousands separator for
                                monetary values */
  char *mon_grouping;        /* specifies grouping for
                                monetary values */
  char *positive_sign;       /* positive value indicator for
                                monetary values */
  char *negative_sign;       /* negative value indicator for
                                monetary values */
  char int_frac_digits;      /* number of digits displayed to the
                                right of the decimal point for
                                monetary values displayed using
                                international format */
  char frac_digits;          /* number of digits displayed to the
                                right of the decimal point for
                                monetary values displayed using
                                local format */
  char p_cs_precedes;        /* 1 if currency symbol precedes
                                positive value, 0 if currency
                                symbol follows value */
  char p_sep_by_space;       /* 1 if currency symbol is
                                separated from value by a space,
                                0 otherwise */
  char n_cs_precedes;        /* 1 if currency symbol precedes
                                a negative value, 0 if currency
                                symbol follows value */
  char n_sep_by_space;       /* 1 if currency symbol is
                                separated from a negative
                                value by a space, 0 if
                                currency symbol follows value */
  char p_sign_posn;          /* indicates position of
                                positive value symbol */
  char n_sign_posn;          /* indicates position of
                                negative value symbol */
}
```

localeconv( )函数返回一个指向 lconv 结构的指针。不必修改这个结构的内容，而是应查阅所用的编译器文档，以了解与 lconv 结构相关的具体实现信息。

与 localeconv( )相关的函数是 setlocale( )。

28.7 localtime 函数

```
#include <ctime>
struct tm *localtime(const time_t *time);
```

localtime( ) 函数返回一个指向 time 的分解形式的指针，这个 time 具有 tm 结构形式。该时间用当地时间表示。time 的值通常可以通过调用 time( )获得。

localtime( )用来存放过时的时间形式的结构是静态分配的并且在每次调用该函数时被重写。如果想保存该结构的内容，必须将其复制到其他地方。

与 localtime( )相关的函数有 gmtime( )，time( )和 asctime( )。

28.8 mktime 函数

```
#include <ctime>
time_t mktime(struct tm *time);
```

mktime( )函数返回与 time 所指结构中的故障时间相对应的日历时间。因为 tm_wday 和 tm_yday 元素由函数设置，所以调用时不需要定义这些元素。

如果 mktime( )不能把信息表示为有效的日历时间，函数将返回 -1。

与 mktime( )相关的函数有 time( )，gmtime( )，asctime( )和 ctime( )。

28.9 setlocale 函数

```
#include <clocale>
char *setlocale(int type, const char *locale);
```

setlocale( )函数允许对某些参数进行查询和设置，它们是对程序执行的地理环境敏感的参数。如果 locale 为空，setlocale( )将返回一个指向当前定位字符串的指针；否则，setlocale( )将试图利用 locale 指定的字符串将 locale 设置为 type 指定的参数。可以查阅相关的编辑器文档，来找到它所支持的定位字符串。

调用这个函数时，type 必须是下面列出的宏中的一个：

LC_ALL
LC_COLLATE
LC_CTYPE
LC_MONETARY
LC_NUMERIC
LC_TIME

LC_ALL 涉及所有的定位类别，LC_COLLATE 影响 strcoll( )函数的操作，LC_CTYPE 改变字符函数的工作方式，LC_MONETARY 确定货币格式，LC_NUMERIC 改变格式化输入/输出函数的小数点字符，最后，LC_TIME 确定 strftime( ) 函数的行为。

setlocale( )函数返回一个指向与 type 参数相关联的字符串的指针。

与 setlocale( )相关的函数有 localeconv( )，time( )，strcoll( )和 strftime( )。

28.10 strftime 函数

```
#include <ctime>
size_t strftime(char *str, size_t maxsize, const char *fmt,
                const struct tm *time);
```

strftime( )函数根据fmt 所指字符串中的格式指令并使用分解时间time把时间和日期信息与其他信息一起放入 str 所指的字符串，而且将 maxsize 字符的最大值放入 str。

strftime( )函数有点像 sprintf( )，它识别以百分号（%）开头的一组格式指令并把格式化的输出放入一个字符串。这些格式指令被用于指定在str 中表示各种时间和日期信息的精确方法，而格式字符串中的所有其他字符将原封不动地放入str，时间和日期将以当地时间显示。下表形式了这些格式指令，其中许多指令是大小写敏感的。

strftime( )函数返回放入 str 所指字符串中的字符个数；如果发生错误，则返回 0。

指令	取代物
%a	简写的每星期各天的名字
%A	完整的每星期各天的名字
%b	简写的月份名
%B	完整的月份名
%c	标准日期和时间字符串
%d	按十进制数字计算的每月的第几天（1～31）
%H	小时（0～23）
%I	小时（1～12）
%j	按十进制数字计算的每年的第几天（1～366）
%m	按数字计算的月份数（1～12）
%M	按十进制数字计算的分钟数（0～59）
%p	AM 或 PM 的地方对应值
%S	按十进制数字计算的秒数（0～60）
%U	每年的第几个星期，星期日为第一天（0～53）
%w	按十进制数字计算的每个星期的天数（0～6，星期日为0）
%W	每年的第几个星期，星期一为第一天（0～53）
%x	标准日期字符串
%X	标准时间字符串
%y	不包括世纪在内的按十进制数字计算的年数（0～99）
%Y	包括世纪在内的按十进制数字计算的年数
%Z	时区名
%%	百分号

与 strftime( )相关的函数有 time( )，localtime( )和 gmtime( )。

28.11 time 函数

```
#include <ctime>
```

```
time_t time(time_t *time);
```

time( )函数返回当前的系统日历时间。如果系统没有时间，则返回 -1。

time( )函数既可以用一个空指针调用，也可以用一个指向time_t类型的变量的指针调用。如果是后者，函数还将把日历时间赋给该变量。

与 time( )相关的函数有 localtime( )，gmtime( )，strftime( )和 ctime( )。