NAME

easterg, **easterog**, **easteroj**, **gdate**, **jdate**, **ndaysg**, **ndaysj**, **week**, **weekday** - Calendar arithmetic for the Christian era

LIBRARY

Calendar Arithmetic Library (libcalendar, -lcalendar)

SYNOPSIS

#include <calendar.h>

*struct date **
easterg(*int year*, *struct date *dt*);

*struct date **
easterog(*int year*, *struct date *dt*);

*struct date **
easteroj(*int year*, *struct date *dt*);

*struct date **
gdate(*int nd*, *struct date *dt*);

*struct date **
jdate(*int nd*, *struct date *dt*);

int
ndaysg(*struct date *dt*);

int
ndaysj(*struct date *dt*);

int
week(*int nd*, *int *year*);

int
weekday(*int nd*);

DESCRIPTION

These functions provide calendar arithmetic for a large range of years, starting at March 1st, year zero (i.e., 1 B.C.) and ending way beyond year 100000.

Programs should be linked with **-lcalendar**.

The functions **easterg**(), **easterog**() and **easteroj**() store the date of Easter Sunday into the structure pointed at by *dt* and return a pointer to this structure. The function **easterg**() assumes Gregorian Calendar (adopted by most western churches after 1582) and the functions **easterog**() and **easteroj**() compute the date of Easter Sunday according to the orthodox rules (Western churches before 1582, Greek and Russian Orthodox Church until today). The result returned by **easterog**() is the date in Gregorian Calendar, whereas **easteroj**() returns the date in Julian Calendar.

The functions **gdate**(), **jdate**(), **ndaysg**() and **ndaysj**() provide conversions between the common "year, month, day" notation of a date and the "number of days" representation, which is better suited for calculations. The days are numbered from March 1st year 1 B.C., starting with zero, so the number of a day gives the number of days since March 1st, year 1 B.C. The conversions work for nonnegative day numbers only.

The **gdate**() and **jdate**() functions store the date corresponding to the day number *nd* into the structure pointed at by *dt* and return a pointer to this structure.

The **ndaysg**() and **ndaysj**() functions return the day number of the date pointed at by *dt*.

The **gdate**() and **ndaysg**() functions assume Gregorian Calendar after October 4, 1582 and Julian Calendar before, whereas **jdate**() and **ndaysj**() assume Julian Calendar throughout.

The two calendars differ by the definition of the leap year. The Julian Calendar says every year that is a multiple of four is a leap year. The Gregorian Calendar excludes years that are multiples of 100 and not multiples of 400. This means the years 1700, 1800, 1900, 2100 are not leap years and the year 2000 is a leap year. The new rules were inaugurated on October 4, 1582 by deleting ten days following this date. Most catholic countries adopted the new calendar by the end of the 16th century, whereas others stayed with the Julian Calendar until the 20th century. The United Kingdom and their colonies switched on September 2, 1752. They already had to delete 11 days.

The function **week**() returns the number of the week which contains the day numbered *nd*. The argument **year* is set with the year that contains (the greater part of) the week. The weeks are numbered per year starting with week 1, which is the first week in a year that includes more than three days of the year. Weeks start on Monday. This function is defined for Gregorian Calendar only.

The function **weekday**() returns the weekday (Mo = 0 .. Su = 6) of the day numbered *nd*.

The structure *date* is defined in <*calendar.h*>. It contains these fields:

```
int y;      /* year (0000 - ????) */
int m;      /* month (1 - 12) */
int d;      /* day of month (1 - 31) */
```

The year zero is written as "1 B.C." by historians and "0" by astronomers and in this library.

SEE ALSO

ncal(1), strftime(3)

STANDARDS

The week number conforms to ISO 8601: 1988.

HISTORY

The **calendar** library first appeared in FreeBSD 3.0.

AUTHORS

This manual page and the library was written by Wolfgang Helbig <*helbig@FreeBSD.org*>.

BUGS

The library was coded with great care so there are no bugs left.