Table 26.4 E- and O-Modified Conversion Specifiers for the strftime Function (C99 only)

Conversion	*Replacement*
%Ec	Alternative date and time representation
%EC	Name of base year (period) in alternative representation
%Ex	Alternative date representation
%EX	Alternative time representation
%Ey	Offset from %EC (year only) in alternative representation
%EY	Full alternative year representation
%Od	Day of month, using alternative numeric symbols (filled with leading zeros or with leading spaces if there is no alternative symbol for zero)
%Oe	Day of month, using alternative numeric symbols (filled with leading spaces)
%OH	Hour on 24-hour clock, using alternative numeric symbols
%OI	Hour on 12-hour clock, using alternative numeric symbols
%Om	Month, using alternative numeric symbols
%OM	Minute, using alternative numeric symbols
%OS	Second, using alternative numeric symbols
%Ou	ISO 8601 weekday as a number in alternative representation, where Monday is 1
%OU	Week number, using alternative numeric symbols
%OV	ISO 8601 week number, using alternative numeric symbols
%Ow	Weekday as a number, using alternative numeric symbols
%OW	Week number, using alternative numeric symbols
%Oy	Last two digits of year, using alternative numeric symbols

second step is to convert the time to string form and print it. The easiest way to do the second step is to call ctime, which returns a pointer to a string containing a date and time, then pass this pointer to puts or printf.

So far, so good. But what if we want the program to display the date and time in a particular way? Let's assume that we need the following format, where 06 is the month and 03 is the day of the month:

```
06-03-2007    5:48p
```

The ctime function always uses the same format for the date and time, so it's no help. The strftime function is better; using it, we can almost achieve the appearance that we want. Unfortunately, strftime won't let us display a one-digit hour without a leading zero. Also, strftime uses AM and PM instead of a and p.

When strftime isn't good enough, we have another alternative: convert the calendar time to a broken-down time, then extract the relevant information from the tm structure and format it ourselves using printf or a similar function. We might even use strftime to do some of the formatting before having other functions complete the job.

The following program illustrates the options. It displays the current date and time in three formats: the one used by ctime, one close to what we want (created using strftime), and the desired format (created using printf). The ctime version is easy to do, the strftime version is a little harder, and the printf version is the most difficult.