<https://www.codingunit.com/c-reference-time-h-function-time>

## 1. **time():**

time\_t time ( time\_t \* ptr\_time ); The function time() returns the type time\_t. The time that is returned represents the number of seconds elapsed since 00:00 hours, Jan 1, 1970 UTC. It’s also called UNIX EPOCH time.

**Parameters:**

ptr\_time is a pointer to an object of type time\_t. In this object the time value is stored.

It is also allowed to fill in a null pointer, but the a time\_t object is still returned by the function time().

**Return Value:**

The current calendar time as a time\_t object.

If the argument is not a null pointer, the return value is the same as the one stored in the location pointed by the argument.

If the function time() could not retrieve the calendar time, it will return a -1 value.

#include <stdio.h>

#include <time.h>

int main ()

{

time\_t in\_seconds;

in\_seconds = time (NULL);

printf ("%ld hours since January 1, 1970,

EPOCH time!", in\_seconds/3600);

return 0;

}

[How to use Time and Date in C » CodingUnit Programming Tutorials](https://www.codingunit.com/c-tutorial-how-to-use-time-and-date-in-c)

// This calendar example is provided by:

// http://www.codingunit.com

#include<stdio.h>

#define TRUE 1

#define FALSE 0

int days\_in\_month[]={0,31,28,31,30,31,30,31,31,30,31,30,31};

char \*months[]=

{

" ",

"\n\n\nJanuary",

"\n\n\nFebruary",

"\n\n\nMarch",

"\n\n\nApril",

"\n\n\nMay",

"\n\n\nJune",

"\n\n\nJuly",

"\n\n\nAugust",

"\n\n\nSeptember",

"\n\n\nOctober",

"\n\n\nNovember",

"\n\n\nDecember"

};

int inputyear(void)

{

int year;

printf("Please enter a year (example: 1999) : ");

scanf("%d", &year);

return year;

}

int determinedaycode(int year)

{

int daycode;

int d1, d2, d3;

d1 = (year - 1.)/ 4.0;

d2 = (year - 1.)/ 100.;

d3 = (year - 1.)/ 400.;

daycode = (year + d1 - d2 + d3) %7;

return daycode;

}

int determineleapyear(int year)

{

if(year% 4 == FALSE && year%100 != FALSE || year%400 == FALSE)

{

days\_in\_month[2] = 29;

return TRUE;

}

else

{

days\_in\_month[2] = 28;

return FALSE;

}

}

void calendar(int year, int daycode)

{

int month, day;

for ( month = 1; month <= 12; month++ )

{

printf("%s", months[month]);

printf("\n\nSun Mon Tue Wed Thu Fri Sat\n" );

// Correct the position for the first date

for ( day = 1; day <= 1 + daycode \* 5; day++ )

{

printf(" ");

}

// Print all the dates for one month

for ( day = 1; day <= days\_in\_month[month]; day++ )

{

printf("%2d", day );

// Is day before Sat? Else start next line Sun.

if ( ( day + daycode ) % 7 > 0 )

printf(" " );

else

printf("\n " );

}

// Set position for next month

daycode = ( daycode + days\_in\_month[month] ) % 7;

}

}

int main(void)

{

int year, daycode, leapyear;

year = inputyear();

daycode = determinedaycode(year);

determineleapyear(year);

calendar(year, daycode);

printf("\n");

}

[**Download here the source code of this calendar example.**](https://www.codingunit.com/source-code/calendar.c)