



Программирование

[Главная](#)[Скачать](#)[Контакты](#)[Карта сайта](#)

Функции работы с датой и временем

Язык Си / Функции работы с датой и временем

Для работы с системной датой и временем используется библиотека **time.h**.

Типы, определенные в библиотеке **time.h**

Тип	Описание
size_t	Целочисленный тип, возвращаемый операцией sizeof.
clock_t	Арифметический тип, подходящий для представления времени.
time_t	Арифметический тип, подходящий для представления времени.
struct tm	Структурный тип, содержащий компоненты календарного времени.

Рубрики

- ☐ Представление данных и архитектура ЭВМ
- ☐ Создание Windows-приложений
- ☐ Язык Си
- ☐ Язык C++
- ☐ Язык ассемблера
- ☐ Структуры данных
- ☐ Алгоритмизация
- ☐ Алгоритмы сортировки и поиска
- ☐ Задачи и их решение
- ☐ Программирование микроконтроллеров

Свежие записи

- ☐ Односвязный линейный

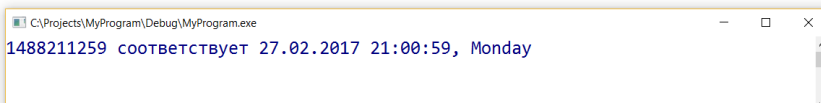
Для определения текущего календарного времени используется функция

```
time_t time(NULL);
```

Данная функция возвращает время в секундах начиная с 1 января 1970 г.

Например,

```
1  #define _CRT_SECURE_NO_WARNINGS
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <string.h>
5  #include <malloc.h>
6  #include <time.h>
7  char * settime(struct tm *u)
8  {
9      char s[40];
10     char *tmp;
11     for (int i = 0; i<40; i++) s[i] = 0;
12     int length = strftime(s, 40, "%d.%m.%Y", u);
13     tmp = (char*)malloc(sizeof(s));
14     strcpy(tmp, s);
15     return(tmp);
16 }
17 int main()
18 {
19     struct tm *u;
20     char *f;
21     system("chcp 1251");
22     system("cls");
23     const time_t timer = time(NULL);
24     u = localtime(&timer);
25     f = settime(u);
26     printf("%ld соответствует ", timer);
27     puts(f);
28     getchar();
29     return 0;
30 }
```



```
C:\Projects\MyProgram\Debug\MyProgram.exe
1488211259 соответствует 27.02.2017 21:00:59, Monday
```

список на
базе классов
ООП

14.12.2018

UML-диаграммы
классов

17.10.2017

Защищено:
Пользовательский
сигнал

28.06.2017

Защищено:
Цифро-
аналоговый
преобразователь

19.06.2017

Граф

19.05.2017

Социальные сети

Группа Вконтакте

Структура `tm` имеет вид

```
struct tm
{
    int tm_sec;        // секунды после минут
    int tm_min;        // минуты после часов [
    int tm_hour;       // часы после полуночи
    int tm_mday;       // день месяца [1,31]
    int tm_mon;        // месяц года (январь =
    int tm_year;       // год (1900 год = 0)
    int tm_wday;       // день недели (вс = 0)
    int tm_yday;       // день года (1 января
    int tm_isdst;      // флаг перехода на лет
```

Функция

```
struct tm *localtime(const time_t *ptm);
```

преобразует календарное время, указанное `ptm`, сохраняет его в структуре `tm` и возвращает указатель на нее.

Функция

```
time_t mktime(struct tm *timeptr);
```

осуществляет обратное преобразование.

Функция

```
clock_t clock (void);
```

возвращает наилучшее приближение процессорного времени, прошедшего с момента запуска программы; для получения времени в секундах значение необходимо разделить на константу, определенную в библиотеке `time.h`:

```
#define CLOCKS_PER_SEC 1000
```

Если время не доступно или не может быть представлено, возвращает `(clock_t) (-1)`.

Функция

```
size_t strftime(
char *restrict s, // указатель на выходную строку
size_t max,      // максимальный размер выходной строки
const char *restrict fmt, // указатель на форматную строку
const struct tm *restrict tmptr); // указатель на структуру времени
```

копирует строку `fmt` в строку `s`, заменяя спецификаторы формата в `fmt` соответствующими данными, взятыми из содержимого структуры времени, на которое указывает `tmptr`; в строку `s` помещается не более `max` символов.

Функция возвращает количество символов (исключая нулевой) в результирующей строке. Если результирующая строка (включая нулевой символ) содержит больше, чем `max` символов, функция возвращает 0, а содержимое `s` не определено.

Спец.	Назначение
%a	Локальное сокращенное название дня недели
%A	Локальное полное название дня недели
%b	Локальное сокращенное название месяца
%B	Локальное полное название месяца
%c	Локальный разделитель даты и времени
%d	День месяца в виде десятичного числа (01–31)
%D	Эквивалент %m%d%y
%e	День месяца (десятичное число): однозначные числа дополнены пробелом
%F	Эквивалент %Y-%m-%d
%g	Последние два разряда года (00–99)
%G	Год в виде десятичного числа
%H	Часы (по 24-часовой шкале) в виде десятичного числа (00–23)
%I	Часы (по 12-часовой шкале) в виде десятичного числа (01–12)
%j	День года в виде десятичного числа (001–366)

Спец.	Назначение
%m	Месяц в виде десятичного числа (01–12)
%n	Символ новой строки
%M	Минуты в виде десятичного числа (00–59)
%p	Локальный эквивалент а.м./р.м. для 12-часовой временной шкалы
%r	Локальное 12-часовое время
%R	Эквивалент %H:%M
%S	Секунды в виде десятичного числа (00–61)
%t	Символ горизонтальной табуляции
%T	Эквивалент %H:%M:%S
%u	Номер дня недели (1–7), где 1 соответствует понедельнику
%U	Номер недели в году, считая воскресенье первым днем недели (00–53)
%w	Номер дня недели в виде десятичного числа, начиная с воскресенья (0–6)
%W	Номер недели в году, считая понедельник первым днем недели (00–53)
%y	Год без века в виде десятичного числа (00–99)
%Y	Год с веком в виде десятичного числа
%Z	Смещение от UTC ("–800" означает на 8 ч по Гринвичу западнее).
%Z	Наименование часового пояса (если доступно)
%%	% (то есть знак процента).

Функция, позволяющая вывести день недели на русском языке

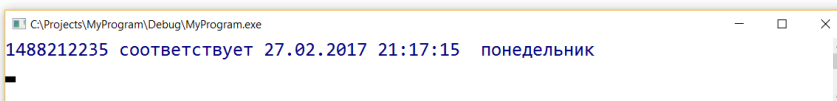
```

1 #define _CRT_SECURE_NO_WARNINGS
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <string.h>
5 #include <malloc.h>

```

```
6  #include <time.h>
7  char * settime(struct tm *u)
8  {
9      char s[40];
10     char *tmp;
11     for (int i = 0; i<40; i++) s[i] = 0;
12     int length = strftime(s, 40, "%d.%m.%Y", u);
13     switch (u->tm_wday)
14     {
15         case 0: strcpy(s + length, " воскресенье");
16         case 1: strcpy(s + length, " понедельник");
17         case 2: strcpy(s + length, " вторник");
18         case 3: strcpy(s + length, " среда");
19         case 4: strcpy(s + length, " четверг");
20         case 5: strcpy(s + length, " пятница");
21         case 6: strcpy(s + length, " суббота");
22     }
23     tmp = (char*)malloc(sizeof(s));
24     strcpy(tmp, s);
25     return(tmp);
26 }
27 int main()
28 {
29     struct tm *u;
30     char *f;
31     system("chcp 1251");
32     system("cls");
33     const time_t timer = time(NULL);
34     u = localtime(&timer);
35     f = settime(u);
36     printf("%ld соответствует ", timer);
37     puts(f);
38     getchar();
39     return 0;
40 }
```

Результат выполнения



```
C:\Projects\MyProgram\Debug\MyProgram.exe
1488212235 соответствует 27.02.2017 21:17:15 понедельник
```

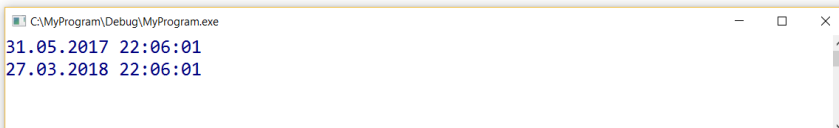
Добавление нескольких дней к текущему времени

Рассмотрим еще один пример. Допустим, требуется добавить несколько дней к текущей дате.

Реализация на Си

```
1  #define _CRT_SECURE_NO_WARNINGS
2  #include <stdio.h>
3  #include <time.h>
4  #define ADD_DAYS 300
5  int main()
6  {
7      struct tm *u;
8      char s1[40] = { 0 }, s2[40] = { 0 };
9      const time_t timer = time(NULL);
10     u = localtime(&timer);
11     strftime(s1, 80, "%d.%m.%Y %H:%M:%S ",
12     printf("%s\n", s1);
13     u->tm_mday += ADD_DAYS;
14     time_t next = mktime(u);
15     u = localtime(&next);
16     strftime(s2, 80, "%d.%m.%Y %H:%M:%S ",
17     printf("%s\n", s2);
18     getchar();
19     return 0;
20 }
```

Результат выполнения



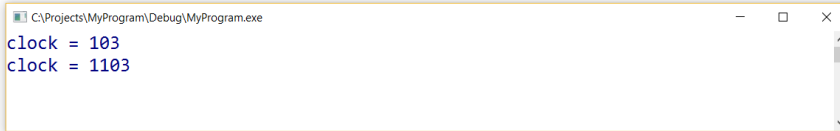
```
C:\MyProgram\Debug\MyProgram.exe
31.05.2017 22:06:01
27.03.2018 22:06:01
```

Реализация функции задержки

```
1  #include <stdio.h>
2  #include <time.h>
3  void delay(int ms) // аргумент - требуемое
4  {
5      int c = clock() + ms;
6      while (clock() < c);
7  }
8  int main()
9  {
10     printf("clock = %d\n", clock());
11     delay(1000);
12     printf("clock = %d\n", clock());
```

```
13  getchar();
14  return 0;
15 }
```

Результат выполнения



```
C:\Projects\MyProgram\Debug\MyProgram.exe
clock = 103
clock = 1103
```

Назад: *Язык Си*

Комментариев к записи: 17



void

11.02.2021 в 11:21

ключевое слово "BUFF_SIZE"

```
1  #define _CRT_SECURE_NO_WARNINGS
2  #include <stdio.h>
3  #include <time.h>
4
5  #define ADD_DAYS  300
6  #define BUFF_SIZE 40
7
8  //////////////////////////////////////
9  int main()
10  //////////////////////////////////////
11  {
12  struct tm *u;
13  char s1[BUFF_SIZE], s2[BUFF_SIZE];
14  time_t timer, next;
15
16  timer      = time(NULL);
17  u          = localtime(&timer);
18  strftime(s1, BUFF_SIZE, "%Y.%m.%d %H:%M", u);
19
20  u->tm_mday += ADD_DAYS;
21  next       = mktime(u);
22  u          = localtime(&next);
23  strftime(s2, BUFF_SIZE, "%Y.%m.%d %H:%M", u);
24
```



```
25 printf("%s\n", s1);
26 printf("%s\n", s2);
27 //getchar();
28 return 0;
29 }
```

Ответить ↓



Роман

10.03.2020 в 13:30

Доброго времени Елена. Подскажите пожалуйста, как получить метку времени UTC, в миллисекундах?

1583835726000 А затем конвертировать в wchar_t*
const time_t timer = time(NULL); Умножение на 1000L,
в VS почему то не срабатывает.

Ответить ↓



Роман

11.03.2020 в 06:15

Кажется разобрался. Я не тот тип указывал в выводимой строке функции wprintf(); Для типа time_t применим %I64d, вместо %ld или %lu. Из за этого не корректно выводилось значение. Ну и конвертацию в wchar_t, выполняет функция _i64tow(timer, buf, 10);

Ответить ↓

**Evgeny**

11.09.2018 в 11:13

Спасибо за ответ. Но я нашел в Интернете формулу расчета разности дат (спасибо опубликовавшему эту формулу). К сожалению, это формула, а не функция C++. Меня удивляет, что нет функции, т.к. по-моему, она должна быть довольно востребована (определение периода, возраста, средних величин за период и т.п.)...

Ответить ↓**Evgeny**

10.09.2018 в 12:46

Здравствуйте, ищу способ определения разницы в днях между двумя датами на C++ в Win32 GUI. Есть два поля DateTimePicker, вот и хочу знать разницу между ними. Думаю, что должна быть какая-то функция, но найти нигде не смог. Искал во многих форумах, везде объяснения для консольных программ. С помощью определения количества дней в годах и вычислении смог получить нужный результат для любых дат, но это заняло у меня примерно 140 строк. Функция difftime() дает какой-то странный, отличный от моего, результат, постоянный независимо от дат. Пишут, что получаются секунды, но не может быть в разнице в 6 лет около 10000 секунд. Можете помочь функцией, примером или ссылкой?

Ответить ↓**Елена Вставская**

10.09.2018 в 20:12

Не приходилось работать с DateTimePicker в Win32 GUI на C++. Знаю только, как сделать в C#.

Ответить ↓

**Anastasia**

29.03.2018 в 19:27

Здравствуйте, скажите, пожалуйста, что такое массив временных штампов? И как можно вывести каждый понедельник из этого массива, что он был четным днем недели?

Ответить ↓**Владимир**

31.05.2017 в 22:51

спасибо, разберу ваш пример и буду нормально оперировать данными. спасибо за оперативность.

Ответить ↓**Владимир**

31.05.2017 в 21:33

добрый вечер, частенько захожу сюда чтобы разобраться лучше.. и все же.. у меня вопрос следующий. Есть дата, к которой нужно прибавить определенное (пользователем) количество дней и вывести новую дату на экран. Как реализовать этот алгоритм? Информации на странице не достаточно. Не могли бы вы написать функции (и параметры этих функций) для решения моей задачи? Что использовать? У меня есть вариант организовать это все без time, но сами понимаете код там будет очень громоздкий...

Ответить ↓

Елена Вставская

31.05.2017 в 22:08

Ответила в тексте статьи

Ответить ↓**Боб**

28.02.2017 в 14:16

Спасибо большое за ответы. Действительно, я не учел часовой пояс.

Ответить ↓**Боб**

27.02.2017 в 17:16

Не подскажите а как получить доступ к дню недели? Мне надо что бы при понедельнике у меня значение принималось 1, при вторнике 2 и тд.

Ответить ↓**Елена Вставская**

27.02.2017 в 18:35

Не поняла вопроса. В примере так и есть: понедельник – 1, вторник – 2.

Ответить ↓**Боб**

28.02.2017 в 12:34

как можно сделать вывод похожий на этот:
`cout<<(tm->tm_wday);` Нужен только день недели без секунд часов и т.д.

Ответить ↓

**Елена Вставская**

28.02.2017 в 12:46

```
1 #include <time.h>
2 #include <iostream>
3 int main()
4 {
5     const time_t SEC_IN_GOD = 3153600
6     using namespace std;
7     const time_t timer = time(NULL);
8     time_t t = timer - (SEC_IN_GOD *
9
10     cout << endl;
11     cout << ((t / 60) / 60) % 24;
12     cout << ":";
13     cout << ((t / 60) % 60) / 10;
14     cout << ((t / 60) % 60) % 10;
15     cout << ":";
16     cout << (t % 60) / 10;
17     cout << (t % 60) % 10;
18
19     getchar();
20     return 0;
21 }
```

Формат %u выводит номер дня недели, %A - название.

Ответить ↓**Боб**

28.02.2017 в 12:55

И почему нижеприведенный код не правильно показывает время в часах? Влияние високосного

дня должен же влиять только на дату, но не на время же

```
1  #include <time.h>
2  #include <iostream>
3  int main()
4  {
5      const time_t SEC_IN_GOD = 3153600
6      using namespace std;
7      const time_t timer = time(NULL);
8      time_t t = timer - (SEC_IN_GOD *
9
10     cout << endl;
11     cout << ((t / 60) / 60) % 24;
12     cout << ":";
13     cout << ((t / 60) % 60) / 10;
14     cout << ((t / 60) % 60) % 10;
15     cout << ":";
16     cout << (t % 60) / 10;
17     cout << (t % 60) % 10;
18
19     getchar();
20     return 0;
21 }
```

Ответить ↓



Елена Вставская

28.02.2017 в 13:12

Проблем не вижу. Время выводится для часового пояса GMT+0 корректно. Отличается от того, что на компьютере ровно на столько часов, сколько соответствуют часовому поясу.

Добавить комментарий

Ваш e-mail не будет опубликован. Обязательные поля помечены *

Комментарий

Имя *

E-mail *

Сайт

☐ Я не робот

reCAPTCHA

Конфиденциальность - Условия использования

Отправить комментарий