

- [Главная](#)
- [История ▼](#)
  - [Домеханический этап ►](#)
    - [Зарождение счета](#)
    - [Бирки](#)
    - [Китайские счетные палочки](#)
    - [Узелковая письменность](#)
    - [Абак](#)
    - [Китайская счетная доска](#)
    - [Суаньпань](#)
    - [Соробан](#)
    - [Счет на линиях](#)
    - [Счет костями](#)
    - [Дощанный счет](#)
    - [Счеты](#)
    - [Палочки Непера](#)
    - [Логарифмы](#)
    - [Логарифмическая линейка](#)
  - [Механический этап ►](#)
    - [Счетная машина Леонардо](#)
    - [Вычисляющие часы](#)
    - [Паскалин](#)
    - [Счетная машина Морленда](#)
    - [Калькулятор Лейбница](#)
    - [Рабдологический абак](#)
    - [Арифмометр Полени](#)
    - [Машина Перейры](#)
    - [Счетная машина Якобсона](#)
  - [Первое поколение ЭВМ](#)
  - [Второе поколение ЭВМ](#)
  - [Третье поколение ЭВМ](#)
  - [Персональные ЭВМ](#)
- [Устройство ПК ▼](#)
  - [Материнская плата](#)
  - [Процессор ►](#)
    - [Устройство процессора](#)
    - [Хронология ЦП Intel](#)
  - [Оперативная память ►](#)
    - [Типы ОЗУ](#)
    - [Динамическая оперативная память](#)
    - [Статическая оперативная память](#)
    - [Магниторезистивная оперативная память](#)
  - [Постоянная память](#)
  - [Периферия](#)
- [ВС и Сети ▼](#)
  - [Надежность ВС ►](#)
    - [Надежность](#)
    - [Повышение надежности](#)
    - [Резервирование оборудования](#)
    - [Контроль исправности](#)
    - [Высоконадежные ВС](#)
    - [Программное резервирование](#)
    - [Избыточное кодирование](#)
    - [Контрольная сумма CRC](#)
    - [Коды Хемминга](#)
  - [Бортовые ВС](#)
  - [Сети](#)
- [Разработка ПО ▼](#)
  - [Системы контроля версий ►](#)
    - [Введение в СКВ](#)
    - [Обзор СКВ](#)
    - [Начинаем работать с СКВ GIT](#)
  - [Виртуальные ЭВМ ►](#)
    - [Виртуальные машины](#)
    - [Технология виртуализации](#)
    - [Начинаем работать с VMware](#)
    - [Начинаем работать с VirtualBox](#)
  - [Язык Си](#)
  - [Стиль Си](#)

- [Поиск](#)
- [Содержание](#)

- [Полезное ▼](#)
  - [Бесплатное ПО первой необходимости](#)
  - [Заставки на рабочий стол ▶](#)
    - [Абстракция](#)
    - [Архитектура](#)
    - [Природа](#)

[Главная](#) ▶ [Разработка ПО](#) ▶ [функции языка Си](#)

## Описание функций языка Си

" L "

[All](#) | [\\_](#) | [A](#) | [B](#) | [C](#) | [D](#) | [E](#) | [F](#) | [G](#) | [H](#) | [I](#) | [J](#) | [K](#) | [L](#) | [M](#) | [N](#) | [O](#) | [P](#) | [Q](#) | [R](#) | [S](#) | [T](#) | [U](#) | [V](#) | [W](#) | [X](#) | [Y](#) | [Z](#)

[ldexp, ldexpf, ldexpl](#) ◀

[localtime](#) ◀

[localtime\\_r](#) ◀

[log, logf, logl](#) ◀

[logb, logbf, logbl](#) ◀

[log2, log2f, log2l](#) ◀

[log10, log10f, log10l](#) ◀

[log1p, log1pf, log1pl](#) ◀

[lround, lroundf, lroundl](#) ◀

[llround, llroundf, llroundl](#) ◀

[llrint, llrintf, llrintl](#) ◀

[lrint, lrintf, lrintl](#) ◀

**localtime – преобразование системного времени в местное.**

### Синтаксис:

```
#include < time.h >
```

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

### Аргументы:

s\_time – указатель на переменную, содержащую время в секундах с 0 часов 1 января 1970 года.

### Возвращаемое значение:

Указатель на структуру, содержащую преобразованное системное время в дату и местное время.

### Описание:

Функция localtime () преобразует время в секундах, истекшее с 0 часов 1 января 1970 года (показание системных часов CLOCK\_REALTIME) в местное (с учетом часового пояса) время и дату. Результат помещается в структуру типа tm и функция возвращает указатель на эту структуру.

Структура tm содержит элементы:

```
int tm_sec – секунды (отсчет с 0);
int tm_min – минуты (отсчет с 0);
int tm_hour – часы (отсчет с 0);
int tm_mday – день месяца (отсчет с 1);
int tm_mon – месяц (отсчет с 0);
int tm_year – год (за начала отсчета принят 1900 год);
int tm_wday – день недели (воскресенье - 0);
int tm_yday – день в году (отсчет с 0);
int tm_isdst – признак "летнее время" (больше нуля если «летнее время», ноль если «зимнее время», меньше нуля если нет информации.
```

Внимание! Не рекомендуется использовать функцию localtime () в многопоточных приложениях, так как данные функции используют общую структуру для сохранения преобразованного времени и одновременный вызов функции из разных потоков может привести к неверному результату работы. Для работы в многопоточных приложениях используйте функцию localtime\_r().

### Пример:

В примере определяется текущее системное время в секундах, преобразуется в локальное время (с учетом часового пояса) с помощью функции localtime, затем локальное время преобразуется в текстовую строку с помощью функции asctime и результат выводится в консоль.

```
#include < stdio.h > //Для printf
#include < time.h > //Для time, localtime, asctime

int main (void)
{
    //Переменная для сохранения текущего системного времени
    long int s_time;
    //Указатель, в который будет помещен адрес структуры с
    //преобразованным временем
    struct tm *m_time;

    //Считываем текущее системное время
    s_time = time (NULL);

    //Преобразуем системное время в локальное
    m_time = localtime (&s_time);

    // С помощью функции asctime преобразуем локальное время в строку
    // и выводим результат на консоль
    printf ("Время: %s\n", asctime (m_time) );

    return 0;
}
```

### Результат:

Время: Sat May 17 01:17:08 2014

**Смотри так же:**

[asctime](#), [asctime\\_r](#), [clock\\_getres](#), [clock\\_gettime](#), [clock\\_gettime](#), [ctime](#), [ctime\\_r](#), [difftime](#), [gmtime](#), [gmtime\\_r](#), [localtime](#), [localtime\\_r](#), [mktime](#), [strftime](#), [time](#)

