# Работа с системной датой и временем

Для работы с системной датой и временем используется библиотека time.h (или ctime).

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. |

|  |  |
| --- | --- |
| **Функции управления временем** | |
| [clock](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-ctime-time-h/funkciya-clock/) | Функция возвращает количество тактов процессора, прошедших с момента запуска программы. |
| [difftime](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-ctime-time-h/funkciya-difftime/) | Возвращает разницу между двумя значениями времени. |
| [mktime](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-ctime-time-h/funkciya-mktime/) | Перевод структуры *timeptr в time\_t.* |
| [time](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-ctime-time-h/funkciya-time/) | Получить текущее время. |
| **Функции преобразования** | |
| [asctime](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-ctime-time-h/funkciya-asctime/) | Преобразование значения структуры *timeptr в строку.* |
| [ctime](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-ctime-time-h/funkciya-ctime/) | Преобразование значения типа time\_t в строку. |
| [gmtime](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-ctime-time-h/funkciya-gmtime/) | Принимает значение текущего времени в секундах и инициализирует структуру в формате GMT. |
| [localtime](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-ctime-time-h/funkciya-localtime/) | Принимает значение текущего времени в секундах и инициализирует структуру в местном формате времени. |
| [strftime](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-ctime-time-h/funkciya-strftime/) | Преобразование времени в символьную строку. |

Для определения текущего календарного времени используется функция time\_t time(NULL);

Данная функция возвращает время в секундах начиная с 1 января 1970 г. Структура tm имеет вид

struct tm

{

int tm\_sec; // секунды после минут [0,59]

int tm\_min; // минуты после часов [0,59]

int tm\_hour; // часы после полуночи [0,23]

int tm\_mday; // день месяца [1,31]

int tm\_mon; // месяц года (январь = 0) [0,11]

int tm\_year; // год (1900 год = 0)

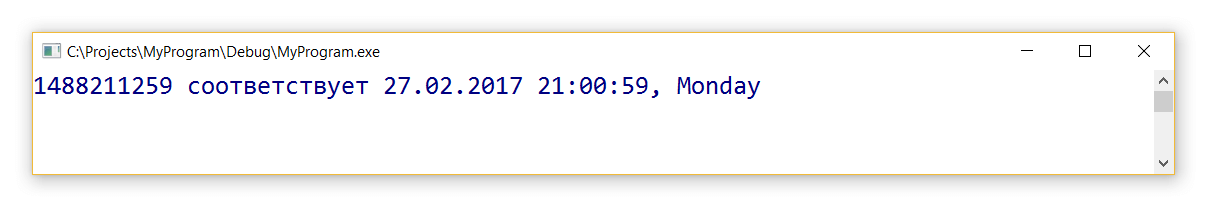
int tm\_wday; // день недели (вс = 0) [0,6]

int tm\_yday; // день года (1 января = 0) [0,365]

int tm\_isdst; // флаг перехода на летнее время (>0- вкл.)

};

|  |  |
| --- | --- |
| 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 %H:%M:%S, %A", 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 | } |



Функция struct tm \*localtime(const time\_t \*ptm);

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

Функция

size\_t strftime(

char \*restrict s, // указатель на выходную строку

size\_t max, // максимальный размер выходной строки

const char \*restrict fmt, // указатель на строку формата

const struct tm \*restrict tmpt); // указатель на структуру tm

копирует строку fmt в строку s, заменяя спецификаторы формата в fmt соответствующими данными, взятыми из содержимого структуры времени, на которое указывает tmpt; 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) |
| %m | Месяц в виде десятичного числа (01-12) |
| %n | Символ новой строки |
| %M | Минуты в виде десятичного числа (00-59) |
| %p | Локальный эквивалент a.m./p.m. 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 clock\_t clock (void);

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

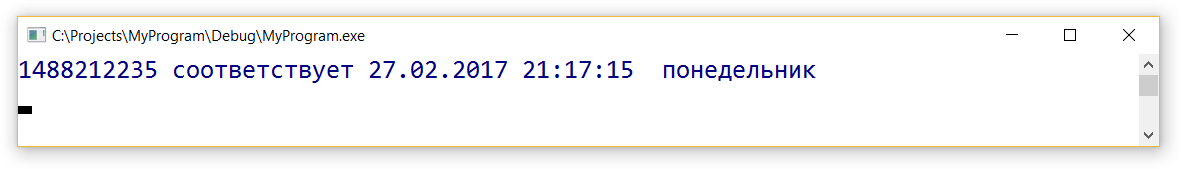
#define CLOCKS\_PER\_SEC 1000

Если время не доступно или не может быть представлено, возвращает -1.

**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 %H:%M:%S ", u); |
| 13 | switch (u->tm\_wday) |
| 14 | { |
| 15 | case 0: strcpy(s + length, " воскресенье"); break; |
| 16 | case 1: strcpy(s + length, " понедельник"); break; |
| 17 | case 2: strcpy(s + length, " вторник"); break; |
| 18 | case 3: strcpy(s + length, " среда"); break; |
| 19 | case 4: strcpy(s + length, " четверг"); break; |
| 20 | case 5: strcpy(s + length, " пятница"); break; |
| 21 | case 6: strcpy(s + length, " суббота"); break; |
| 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 | } |

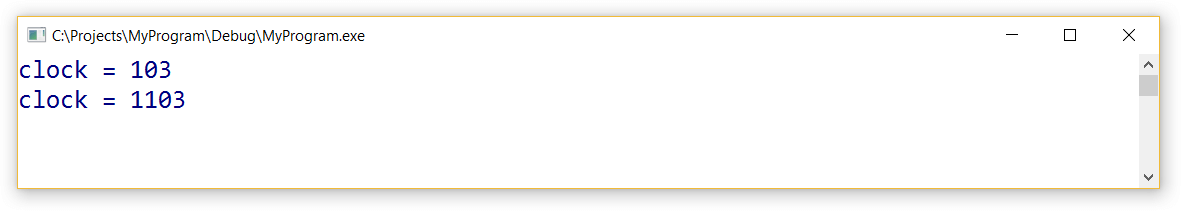
Результат выполнения



**Реализация функции задержки**

|  |  |
| --- | --- |
| 1 | #include <stdio.h> |
| 2 | #include <time.h> |
| 3 | void delay(int ms) // аргумент - требуемое время задержки в миллисекундах |
| 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 | } |

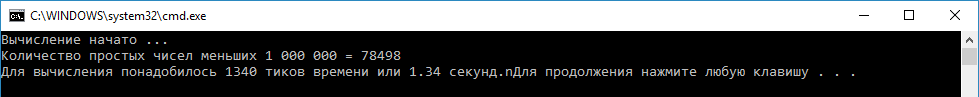
Результат выполнения



**Реализация функции, считающей время выполнения программы**

|  |  |
| --- | --- |
|  | #include "stdafx.h" **// Попробуйте оптимизировать функцию и уменьшить время в 10 раз** |
|  | #include <iostream> // для оператора cout |
|  | #include <ctime> // для функции clock |
|  | #include <cmath> // для функции sqrt |
|  | using namespace std; |
|  | int primes(int number) // функция поиска простых чисел |
|  | { |
|  | int quantity = number - 1; |
|  | for (int i = 2; i <= number; ++i) |
|  | for (int j = sqrt((float)i); j > 1; --j) |
|  | if (i % j == 0) |
|  | { |
|  | --quantity; |
|  | break; |
|  | } |
|  | return quantity; |
|  | } |
|  | int main() |
|  | { |
|  | setlocale(0, "Ru"); |
|  | cout << "Вычисление начато ..." << endl; |
|  | int f = primes(1000000); // ищем количество простых чисел в интервале от 2 до 1000000 |
|  | int t = clock(); // получаем количество тиков времени |
|  | cout << "Количество простых чисел меньших 1 000 000 = " << f << endl; |
|  | cout << "Для вычисления понадобилось " << t << " тиков времени или " |
|  | << ((float)t) / CLOCKS\_PER\_SEC << " секунд.n"; |
|  | return 0; |
|  | } |

Результат выполнения



**Реализация функции, вычисляющей разность в секундах между timestart и timeend**

|  |  |
| --- | --- |
|  | #include "stdafx.h" |
|  | #include <iostream> |
|  | #include <ctime> |
|  | using namespace std; |
|  | int main () |
|  | { |
|  | setlocale(0, "Ru"); |
|  | time\_t start, end; |
|  | char name[80]; |
|  |  |
|  | time (&start);                                 // начальное время |
|  | cout << "Пожалуйста, введите своё имя: "; |
|  | cin.getline(name, 80);                         // считать имя в name |
|  | time (&end);                                   // конечное время |
|  |  |
|  | cout << "Здравствуйте, " << name << endl; |
|  | cout << "Вам понадобилось  " |
|  | << difftime(end, start) |
|  | << " секунд для ввода своего имени\n"; |
|  |  |
|  | return 0; |
|  | } |

Результат выполнения

