Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Операционные системы

Студент: Немкович А.В.

ФИТ 3 курс 1 группа

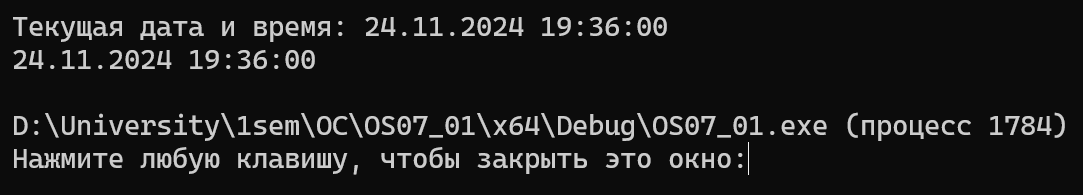
Преподаватель: Бернацкий П.В

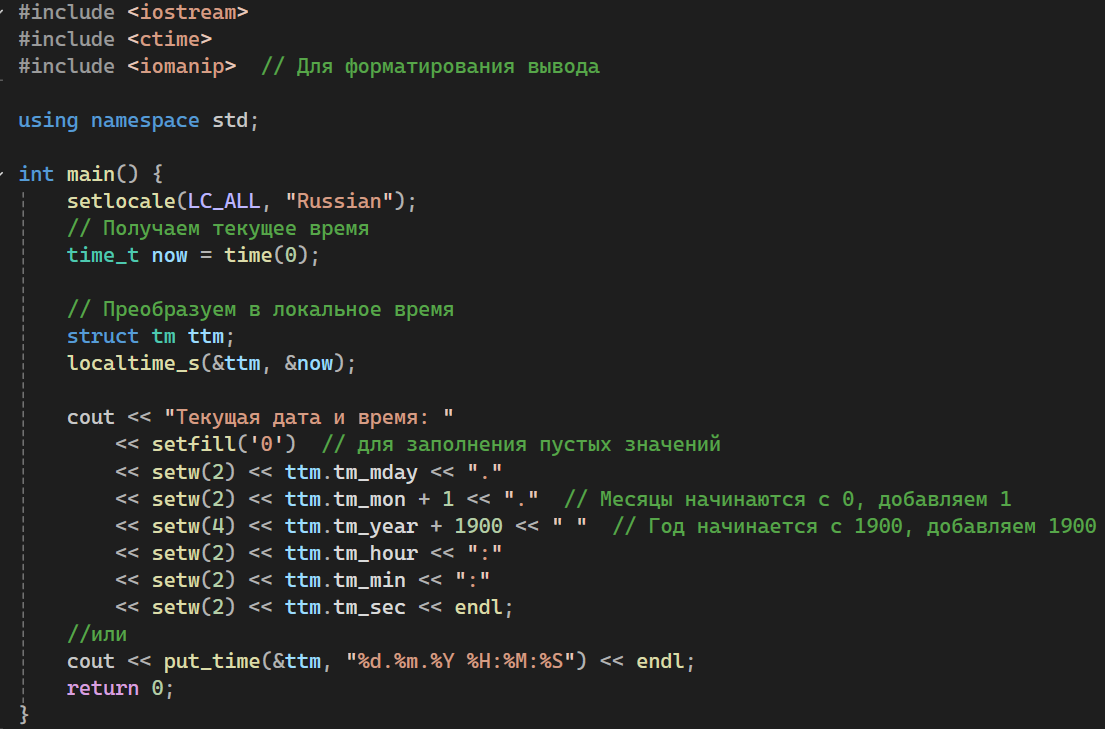
Минск 2024

**Лабораторная работа №7**

**Задание 01. Windows**

Разработайте приложение OS07\_01. Приложение OS07\_01 выводит на консоль текущую локальную дату и время в формате дд.мм.ггг чч:мин:сек.





time возвращает текущие дату и время в виде объекта типа time\_t.

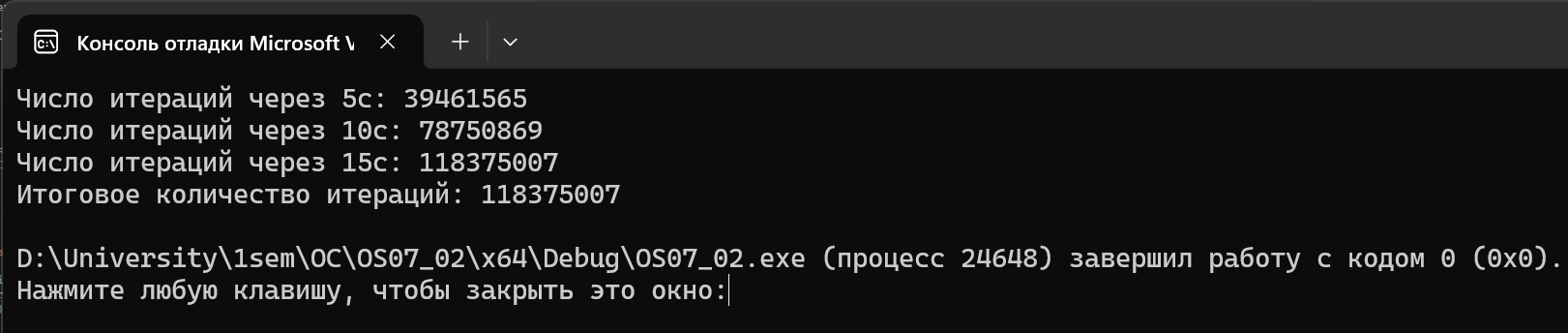
tm имеет особенность: оно не хранит полный год, а считает годы относительно 1900 года. Это сделано для обеспечения обратной совместимости с ранними версиями языка C и стандартных библиотек времени.

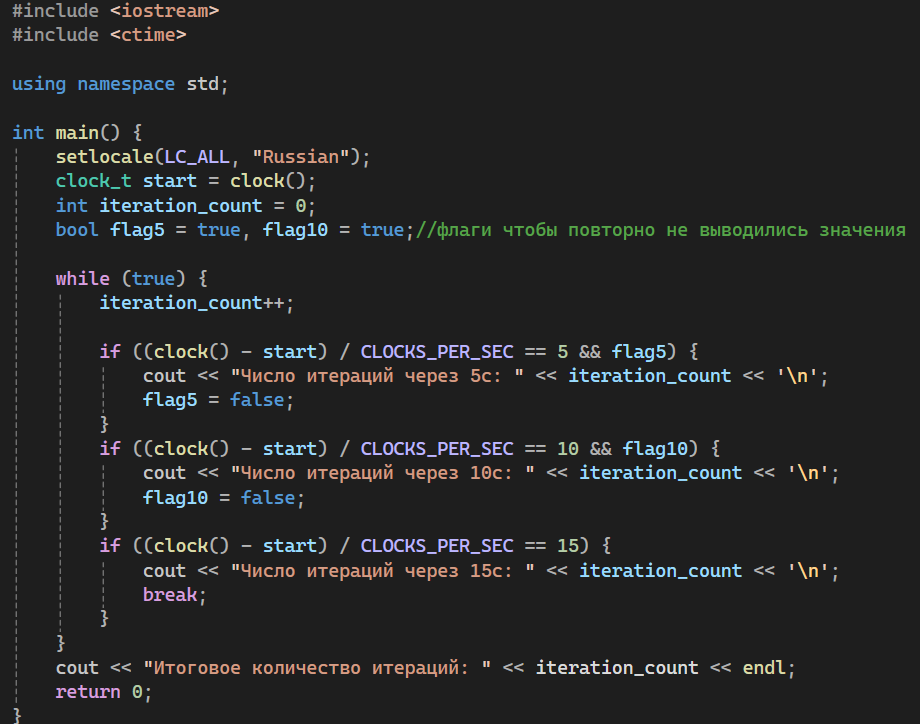
Чтобы собственно получить дату/время и ее компоненты (часы, минуты и т.д.), надо получить из объекта time\_t структуру tm с помощью функции localtime().

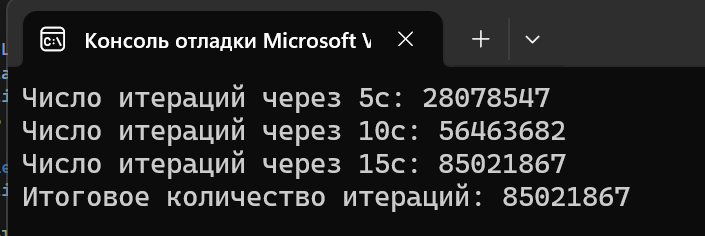
Функция std::put\_time работает как средство для форматирования объектов времени (std::tm) в текстовом представлении. Она принимает объект времени, форматирует его согласно переданной строке формата и выводит результат в поток

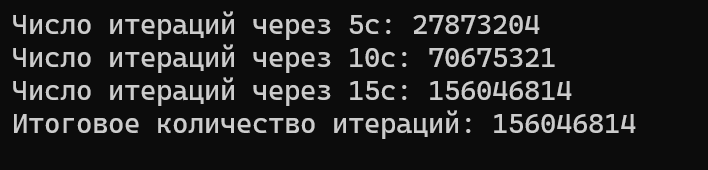
**Задание 02. Windows**

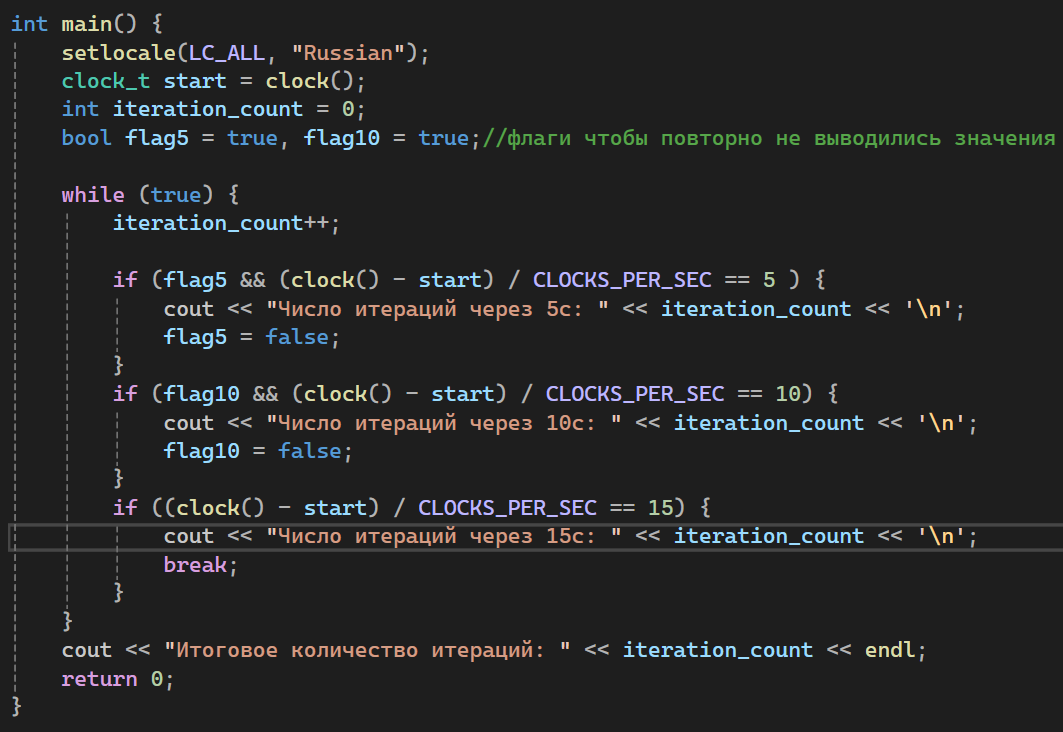
Разработайте приложение OS07\_02, выполняющее бесконечный цикл. В теле цикла подсчитывается количество итераций. Выведите на консоль значения счетчика итераций через 5 сек. и 10 сек. Корректно завершите работу цикла и приложения через 15 сек., выведите итоговое значение счетчика итераций.











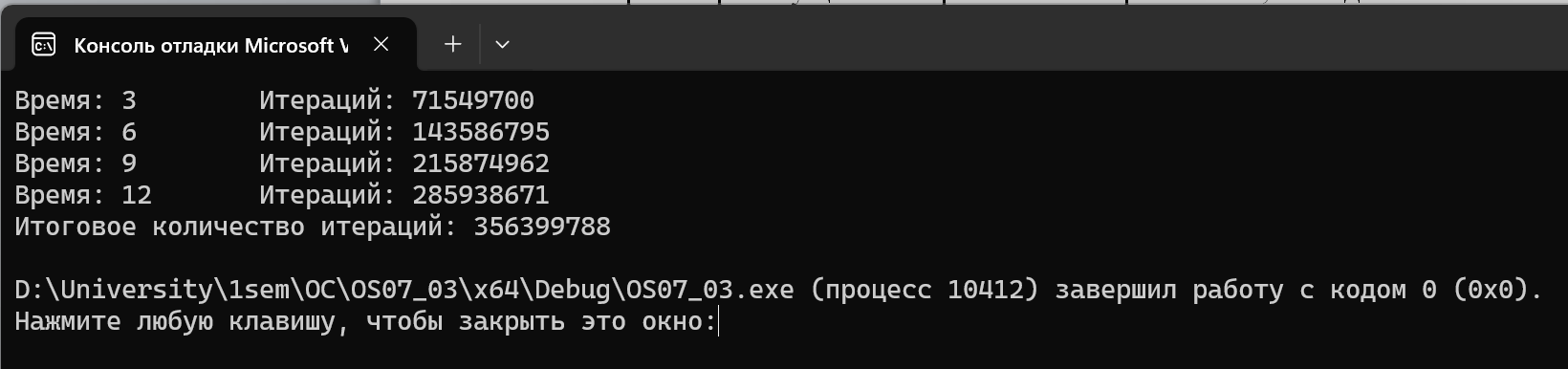
clock() - Функция возвращает количество тактов процессора, прошедших с начала работы программы

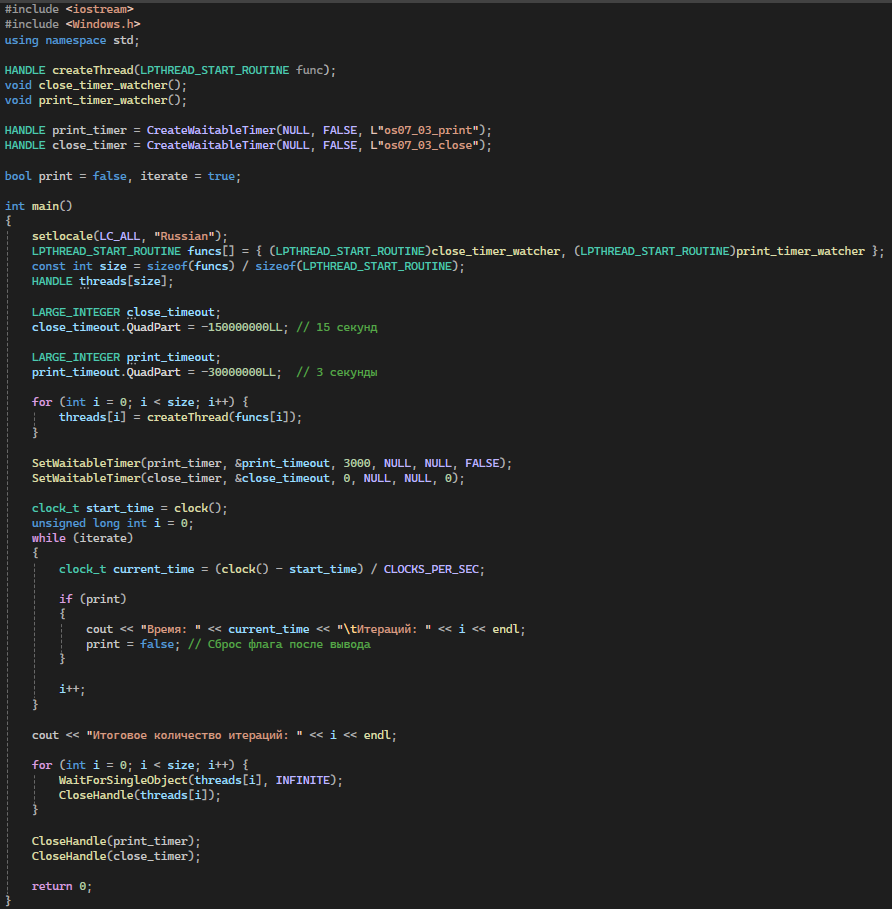
CLOCKS\_PER\_SEC - Константа, определяющая количество тактов в одной секунде. Обычно равна 1 000 000.

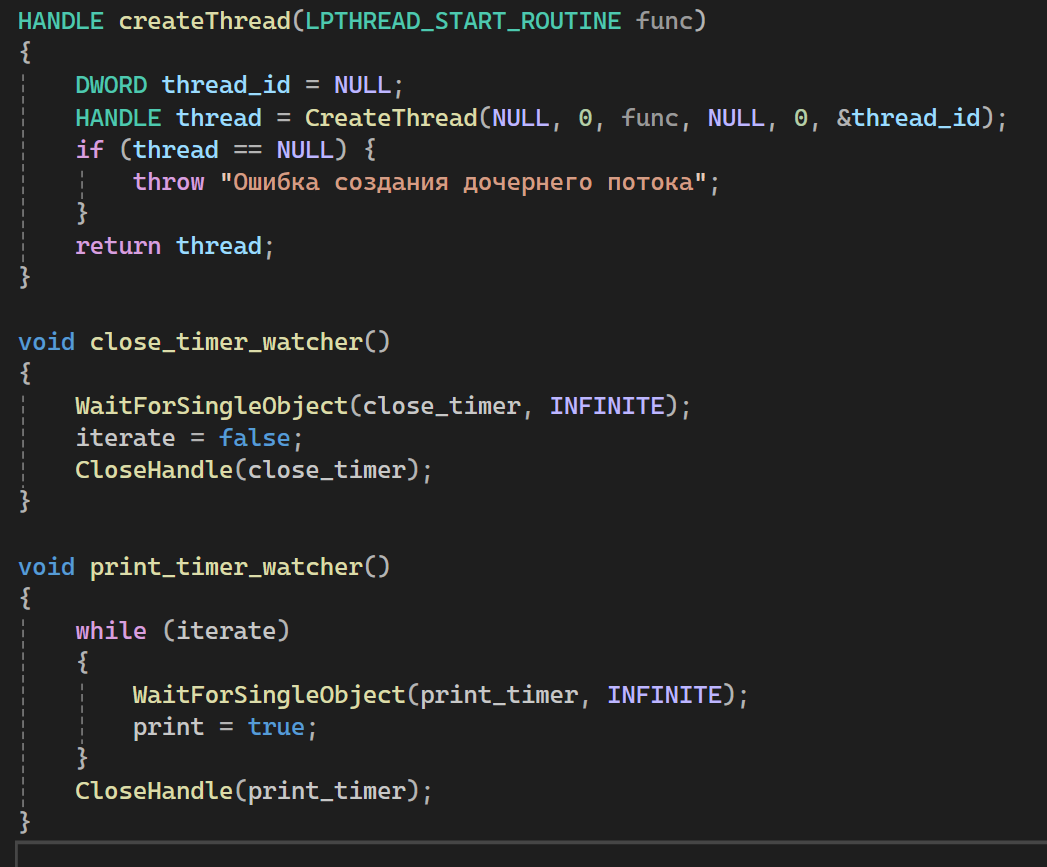
Деление разницы значений clock() на CLOCKS\_PER\_SEC позволяет получить время в секундах.

**Задание 03. Windows**

Указание: самостоятельно освойте и примените периодический ожидающий таймер. Разработайте приложение OS07\_03, выполняющее бесконечный цикл. В теле цикла с задержкой подсчитывается количество итераций. Выведите на консоль значения счетчика итераций каждые 3 сек. Корректно завершите работу цикла и приложения через 15 сек., выведите итоговое значение счетчика итераций.







Ожидающие таймеры - объекты ядра, которые предназначены для отсчета промежутков времени.

Таймеры создаются с помощью функции CreateWaitableTimer

Параметры: атрибуты безопасности, тип сброса FALSE-таймер синхронизации, имя таймера

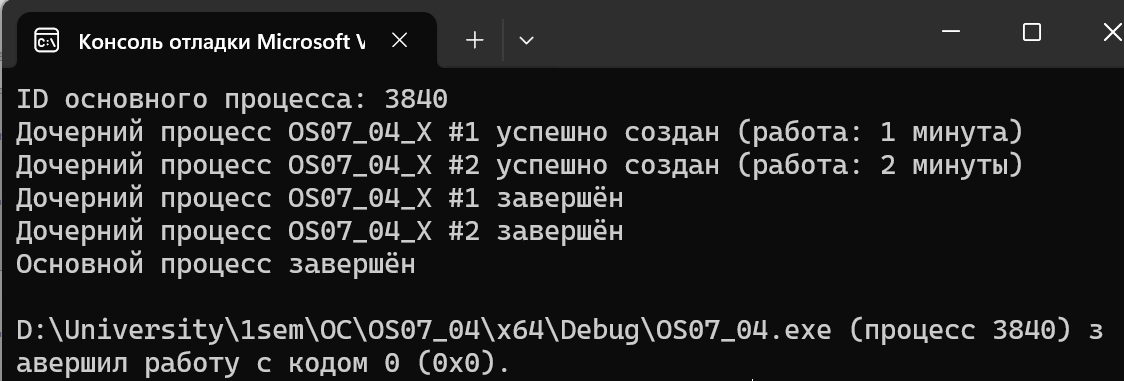
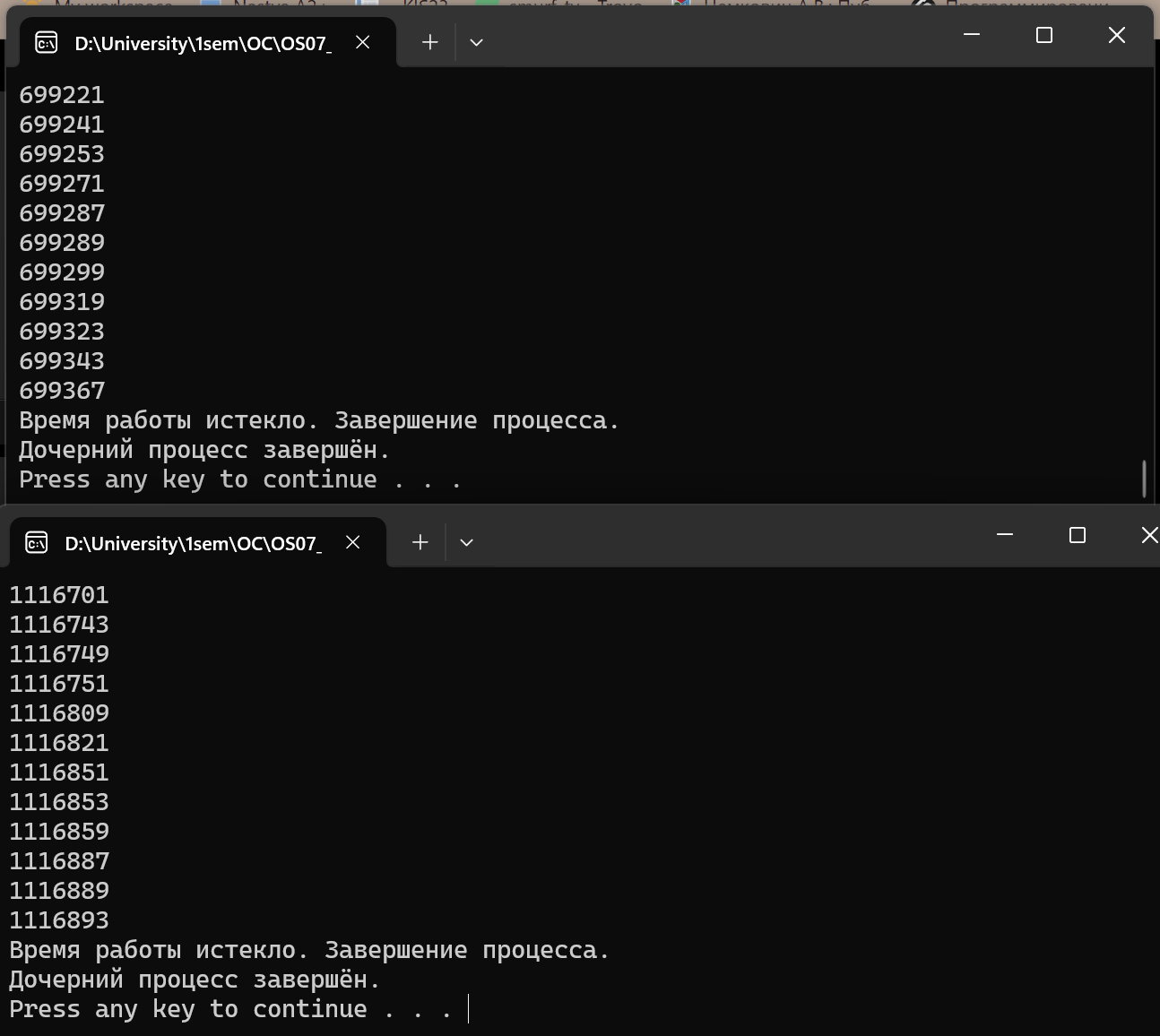
SetWaitableTimer – устанавливаем ожидающий таймер

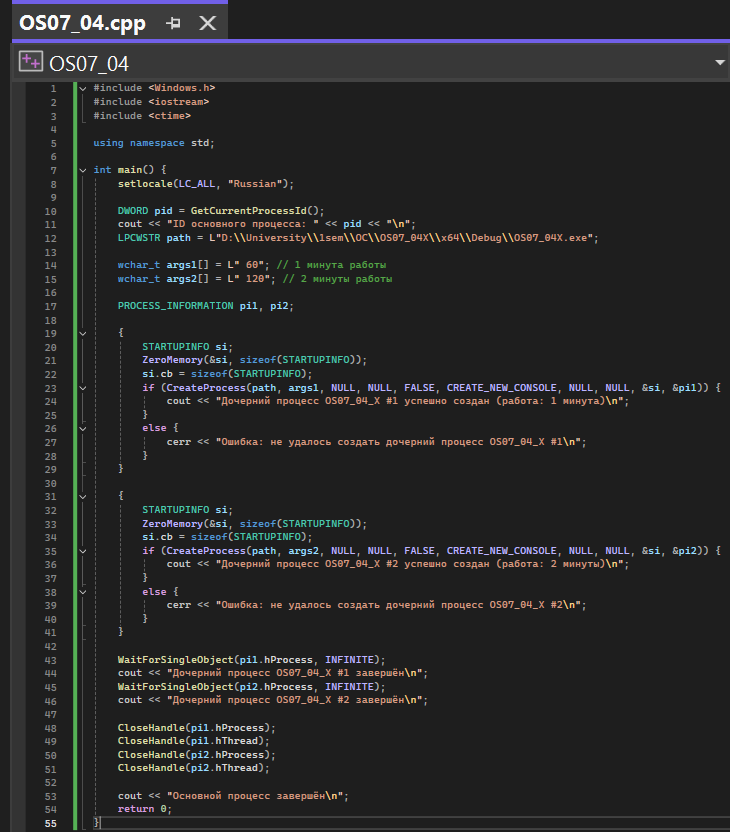
Параметры: дескриптор, время срабатывания, период времени, процедура завершения, параметр процедуры, управление питанием(TRUE: Таймер продолжает отсчет. FALSE: Таймер сбрасывается. Таймер сбрасывается при выходе из спящего режима.)

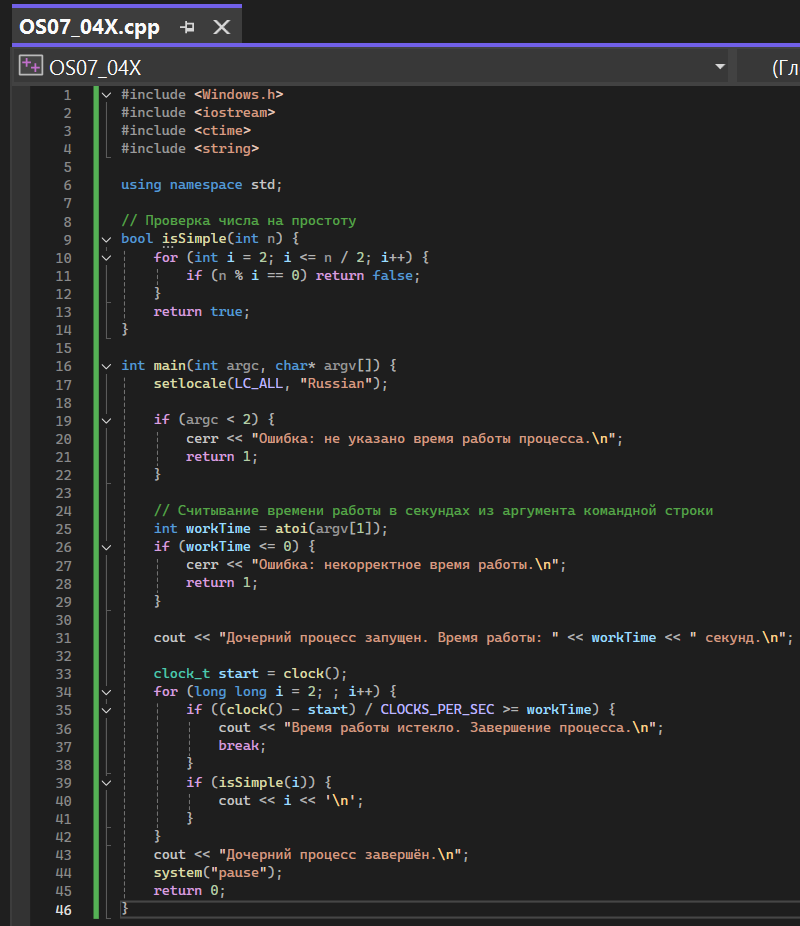
LPTHREAD\_START\_ROUTINE - Это тип, определенный в Windows API, используемый для указания функций, которые будут выполняться в потоках.

**Задание 04. Windows**

Разработайте приложение OS07\_04, запускающее два одинаковых дочерних процесса OS07\_04\_X. Процессы OS07\_04\_X вычисляют и выводят на консоль (каждый в свою) пронумерованный ряд простых положительных чисел (простое число делится нацело только на себя и 1). Первый дочерний процесс должен выполняться 1 минуту и корректно завершаться. Первый дочерний процесс должен выполняться 2 минуты и корректно завершаться. Приложение OS07\_04 завершается после завершения дочерних процессов.



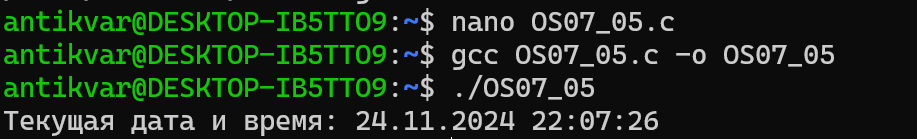


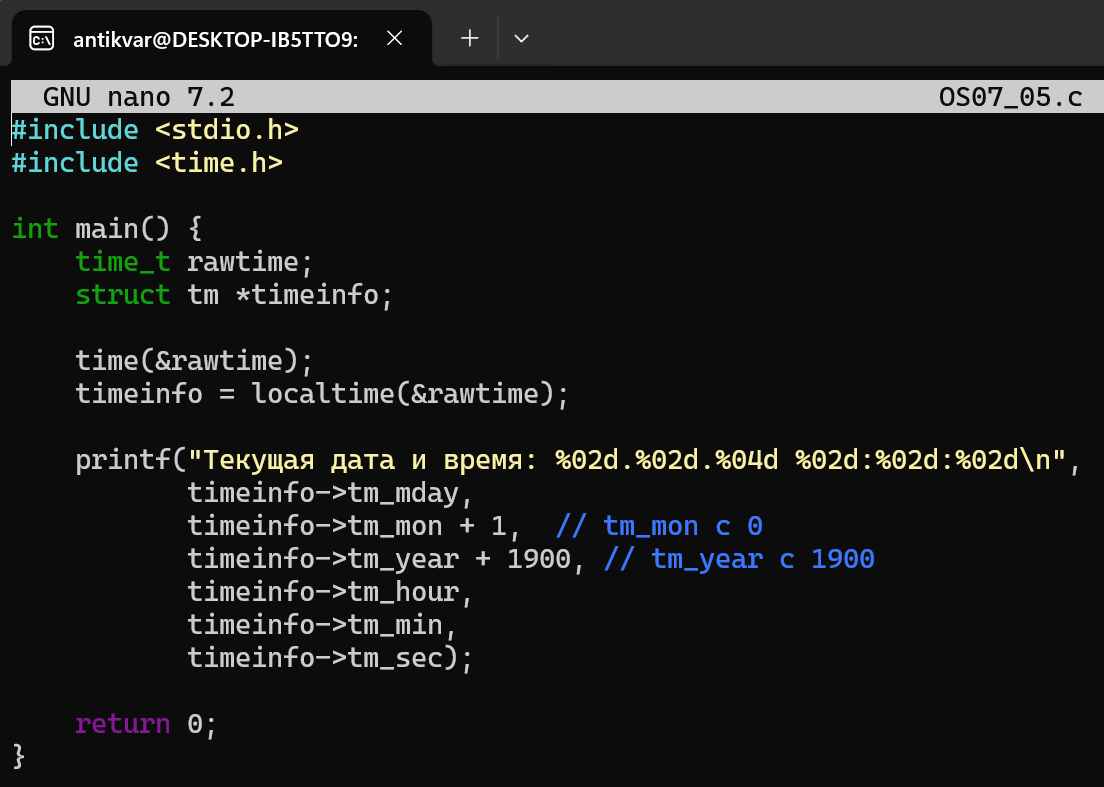


ZeroMemory — это макрос, предоставляемый в Windows API, который используется для заполнения памяти нулями. Применяется для инициализации структуры STARTUPINFO

**Задание 05. Linux**

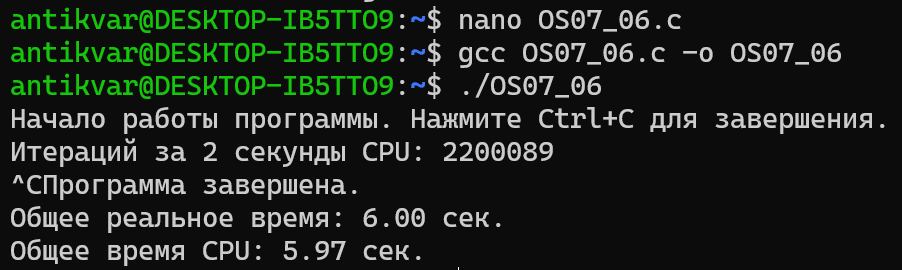
Разработайте приложение OS07\_05. Приложение OS07\_05 выводит на консоль текущую локальную дату и время в формате дд.мм.ггг чч:мин:сек.

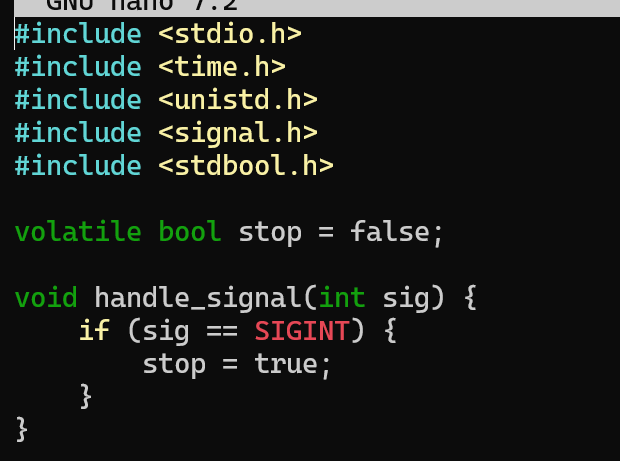


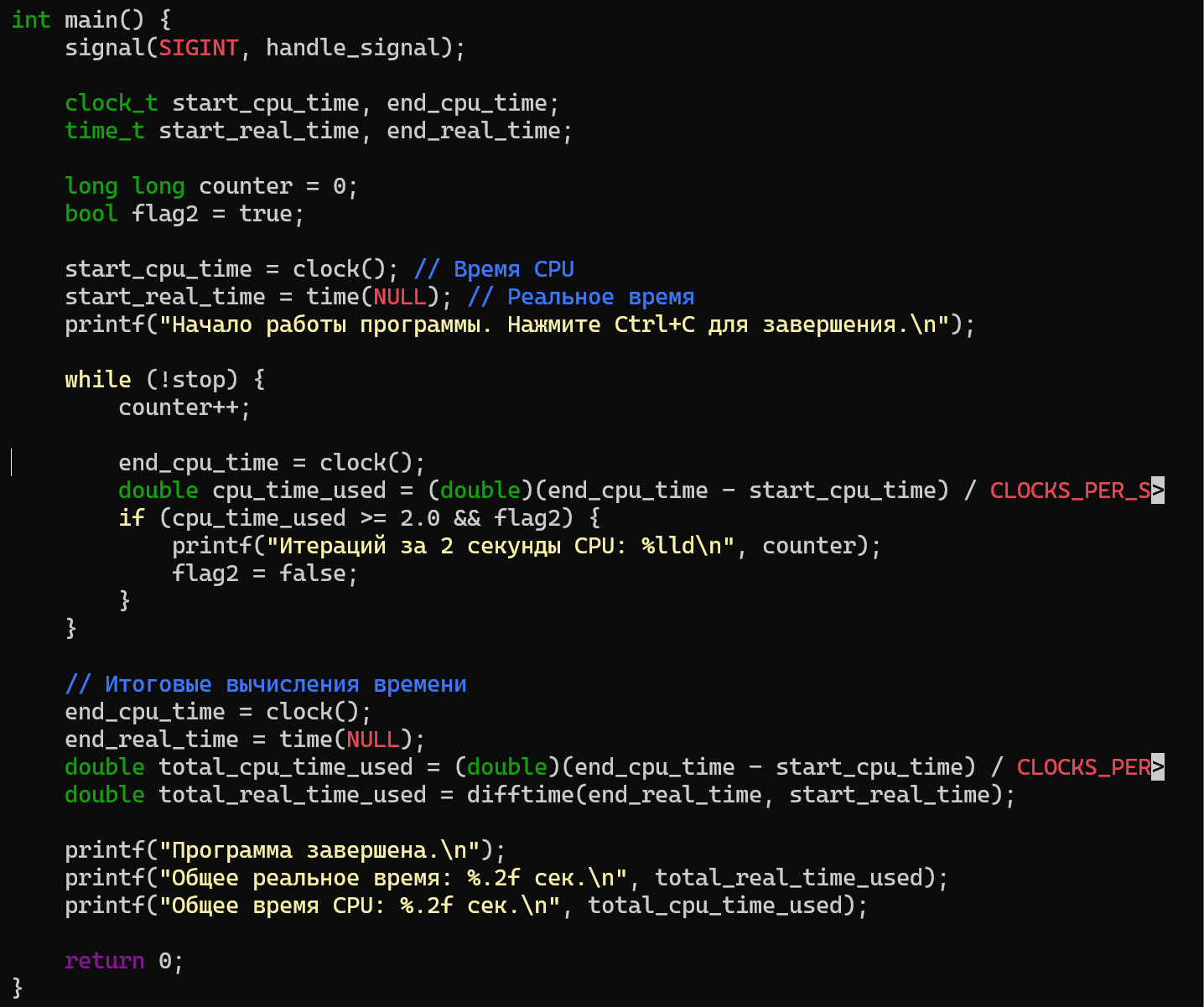


**Задание 06. Linux**

Разработайте приложение OS07\_06, выполняющее бесконечный цикл. В теле цикла подсчитывается количество итераций. Выведите на консоль значения счетчика итераций через 2 сек. процессорного времени и корректно завершите цикл. Приложение OS07\_06 должно выполнять замер реального затраченного на работу цикла времени и выводить его значения на консоль.







signal() — функция для установки обработчика сигналов.

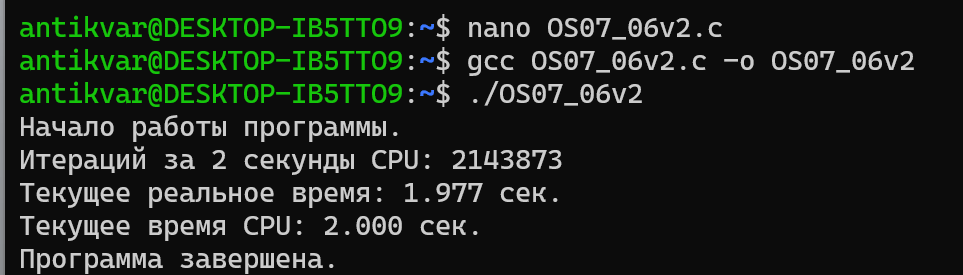
SIGINT — это сигнал, который отправляется при нажатии Ctrl+C в терминале.

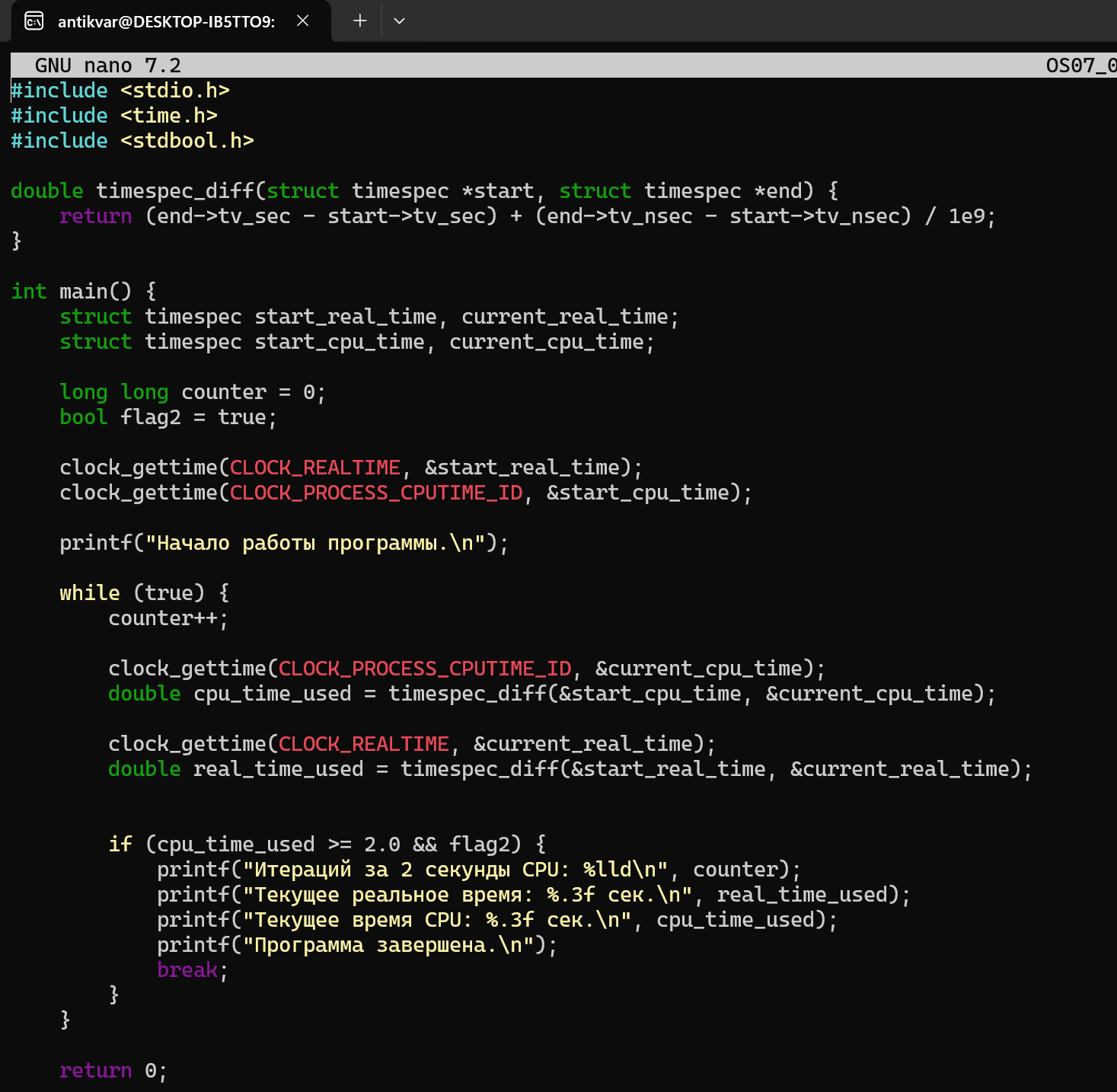
handle\_signal — это функция-обработчик, которая будет вызвана, когда программа получит сигнал SIGINT

**clock()** — функция, которая возвращает количество тиков процессора, использованных программой с момента ее запуска.

**time()** — функция, которая возвращает количество секунд, прошедших с 1 января 1970 года (эпоха Unix). Это значение представлено как тип **time\_t**

CLOCKS\_PER\_SEC — это константа, которая определяет количество тиков в секунду для системной функции clock()





CLOCK\_REALTIME — реальное время с начала эпохи (обычно с 1 января 1970 года).

CLOCK\_PROCESS\_CPUTIME\_ID — время работы процесса на CPU.

timespec\_diff\_sec() - функция для вычисления разницы в секундах между двумя временем, представленными в структуре timespec

**Задание 07. Ответьте на следующие вопросы**

**1. Поясните понятие «социальное время» и почему оно не монотонное?**

Социальное время — это время, которое используется в повседневной жизни для организации человеческой деятельности. Оно отличается от физического времени, которое измеряется с помощью физических процессов (например, с помощью атомных часов). Социальное время регулируется обществом и включает в себя такие понятия, как часы дня, недельный график, календарь и другие повседневные элементы.

OS: Социальное время не монотонное: каждый год солнечный год увеличивается на 3 мс, люди измеряют время не точно, поэтому время от времени делались коррекции, например: в 1582 Папа Григорий XIII пропустил 10 дней календаря. Вычисление даты от Рождества по секундам не получится. Поэтому эпоха Unix (POSIX-время) c 01.01.1970 0:00:00 в секундах. Используется 32 бита для представления числа. В 2038 г. счетчик перейдет в область отрицательных чисел. Секунда координации (по сообщению Международной службы вращения Земли, серверы точного времени): последняя секунда 30.06 или 31.12.

**2. Поясните понятие «эпоха Linux», назовите стартовую дату «эпохи Linux» и в каких единицах изменяется время?**

Эпоха Linux — это стартовая дата отсчета времени в системе Linux, которая определяется как 00:00:00 UTC 1 января 1970 года. Это время называют Unix Epoch или POSIX Epoch.

Время в Linux и Unix-системах измеряется как количество секунд, прошедших с этой даты. Это число называется Unix-временем или POSIX-временем и изменяется в единицах секунд. Для более точных временных измерений могут использоваться и миллисекунды или наносекунды, но в основе всегда лежит отсчет секунд от эпохи Unix.

**3. Поясните понятие «Universal Coordinated Time (UCT)».**

Universal Coordinated Time (UTC) — это международно признанное время, которое используется для синхронизации временных стандартов по всему миру. Оно основано на атомных часах и может корректироваться с учетом вращения Земли, чтобы поддерживать точность времени. В отличие от времени по Гринвичу (GMT), UTC не зависит от часовых поясов и корректируется с использованием високосных секунд.

**4. Поясните понятия «относительное время» и «абсолютное время».**

Относительное время — это время, измеренное относительно некоторой исходной точки или события. Например, если время начала события было установлено как 0, то любое последующее время будет измеряться как прошедшее с момента этого события. В Unix-системах относительное время обычно измеряется как количество секунд, прошедших с момента эпохи Unix.

Абсолютное время — это время, которое указывает на точку во времени независимо от других событий. Это, как правило, универсальное время, которое не зависит от того, когда начинается отсчет. Примером абсолютного времени является UTC или время, заданное в ISO 8601.

**5. Поясните понятие «тик».**

Тик — это минимальная единица измерения времени в контексте работы операционной системы или компьютера. В операционных системах, таких как Linux, тик представляет собой период времени, который проходит между двумя последовательными прерываниями системного таймера. Обычно тик длится несколько миллисекунд и является базовой единицей в ядре ОС для управления временем и синхронизацией задач.

**6. Поясните понятие «ожидающий таймер», перечислите типы таймеров, перечислите состояния, в которых может находится таймер.**

Ожидающие таймеры – объекты ядра, которые предназначены для отсчета промежутков времени. В операционных системах ожидающий таймер часто используется для создания задержек или для планирования задач на будущее.

Переходит в сигнальное состояние при наступлении заданного момента времени. Если ожидающий таймер ждет момента перехода в сигнальное состояние, то говорят, что он находится в активном состоянии. Другое состояние ожидающего таймера пассивное – из этого состояния он не может перейти в сигнальное состояние.

По способу перехода из сигнального стояния в несигнальное, ожидающие таймеры разделяются на таймеры с ручным сбросом и таймеры с автоматическим сбросом, иначе называемые таймерами синхронизации.

По способу перехода из несигнального состояния в сигнальное, ожидающие таймеры бывают периодические и непериодические. Периодические таймеры работают по циклу: активное состояние – сигнальное состояние – активное состояние. Непериодические таймеры могут только один раз перейти из активного состояния в сигнальное.

**7. Перечислите типы часов, используемых в Linux, поясните их назначение.**

Четыре типа часов в Linux:

1. REALTIME (настенное время) - Это системное время, отсчитываемое с эпохи Linux (00:00:00 UTC 1 января 1970 года).
2. MONOTONIC (монотонное время) - Это время, которое начинает отсчитываться с момента загрузки операционной системы.
3. PROCESS (процессорное время процесса) - Измеряет суммарное время, затраченное процессором на выполнение определенного процесса.
4. THREAD (процессорное время потока) - Аналогичен процессорному времени, но отслеживает время, затраченное на выполнение конкретного потока.

**8. Поясните назначение констант HZ, CLOCKS\_PER\_SEC.**

HZ — это константа, которая определяет частоту тиков в операционной системе. Она указывает, сколько тиков происходит за одну секунду.

CLOCKS\_PER\_SEC — это константа, которая указывает количество тиков, которое эквивалентно одной секунде. В большинстве систем она равна 1000, что означает, что процессор отсчитывает 1000 тиков за одну секунду. Эта константа используется для преобразования значений времени в тиках в секунды.