Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Операционные системы

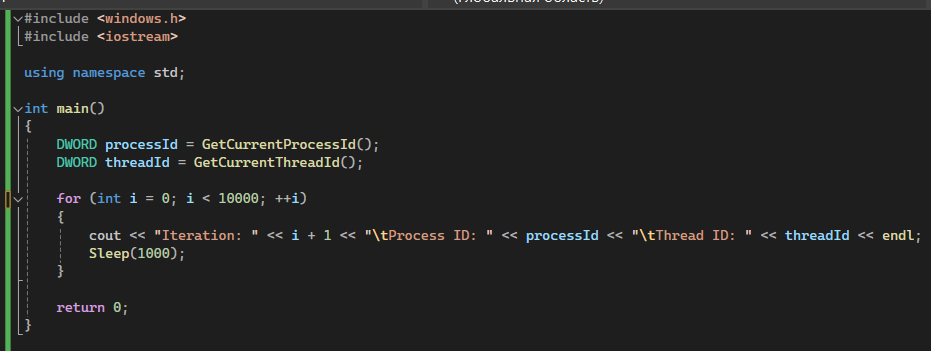
Студент: Авдей А.Ю.

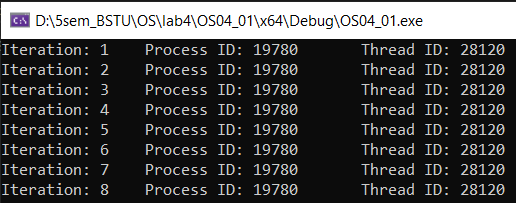
ФИТ 3 курс 2 группа

Преподаватель: Комкова А.В.

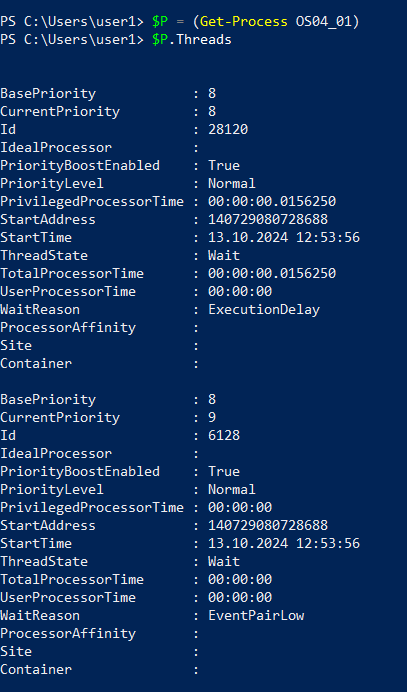
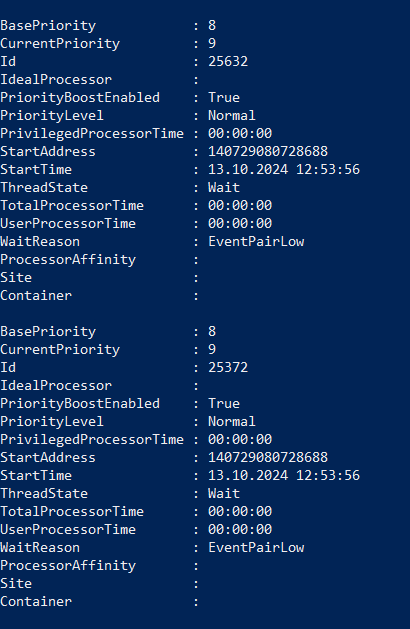
**Задание 01**

1. Разработайте консольное Windows-приложение OS04\_01 на языке С++, выполняющее длинный цикл с временной задержкой и с выводом на консоль идентификаторов текущего процесса и текущего потока.





2. Продемонстрируйте информацию об потоках процесса OS04\_01 с помощью утилит PowerShell ISE и Performance Monitor.

BasePriority: 8 — базовый приоритет потока, который был задан системе.

CurrentPriority: 8 или 9 — текущий приоритет потока. Поток 28120 имеет текущий приоритет 8, а остальные потоки — 9.

Id: уникальный идентификатор потока.

PriorityBoostEnabled: True — система может временно повышать приоритет этого потока для решения некоторых задач.

PriorityLevel: Normal — нормальный уровень приоритета.

PrivilegedProcessorTime: время, проведенное потоком на процессоре для выполнения привилегированных операций. Во всех потоках это значение равно 0.

StartAddress: адрес начала выполнения кода потока.

StartTime: 13.10.2024 12:53:56 — время начала работы потока.

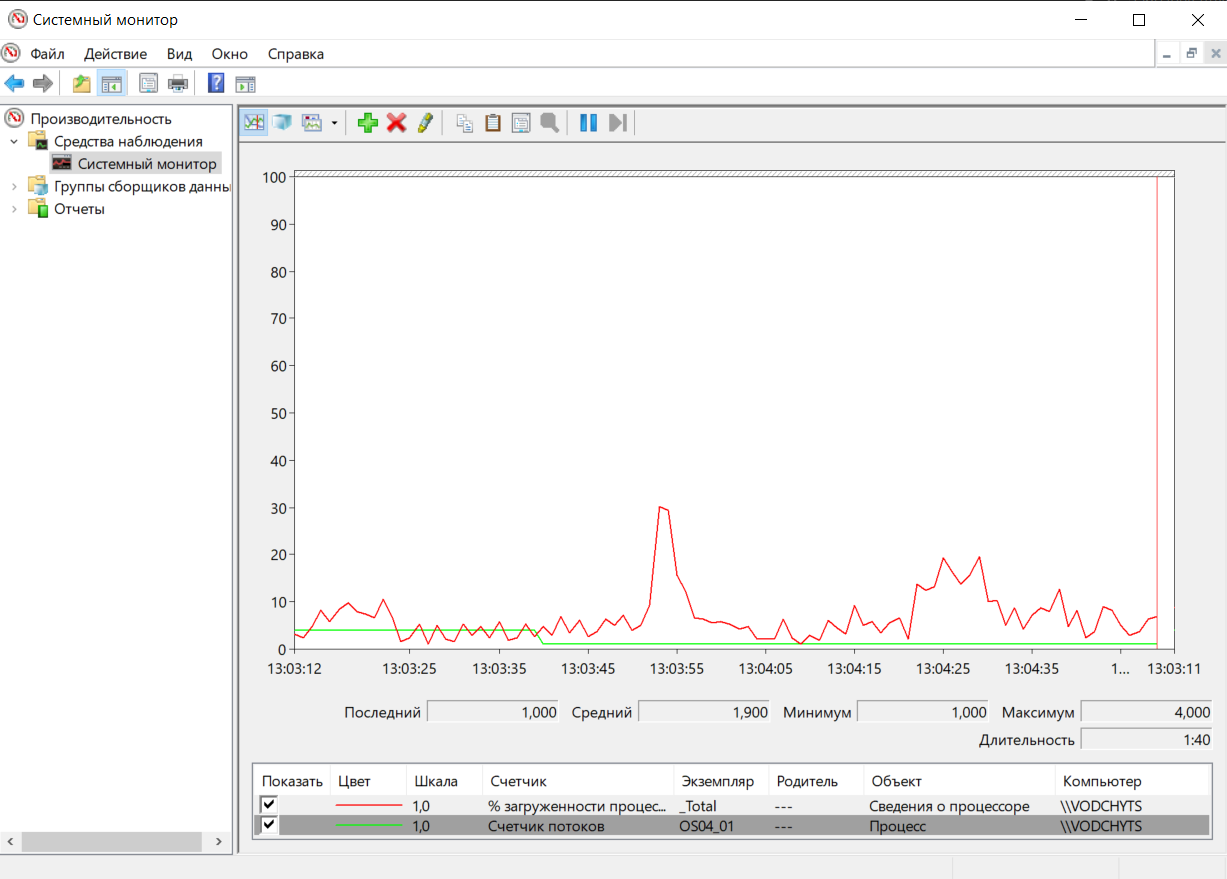
ThreadState: Wait — состояние потока в ожидании.

TotalProcessorTime: время, затраченное процессором на выполнение потока. Все потоки показывают 0, что указывает на то, что они находятся в состоянии ожидания и еще не потребляли процессорное время.

UserProcessorTime: время, проведенное потоком в пользовательском режиме. Опять же, значение 0.

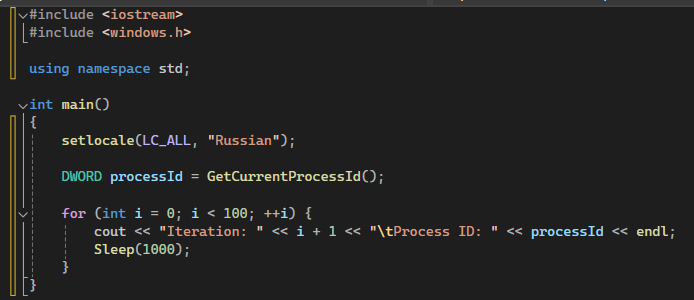
WaitReason: причина ожидания:

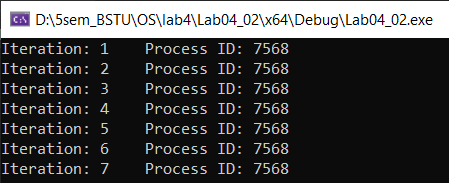
* Поток с ID 28120 имеет причину ожидания ExecutionDelay (Thread execution is delayed).
* Дополнительные потоки (ID 6128, 25632, 25372) находятся в ожидании с причиной EventPairLow (The thread is waiting for event pair low).



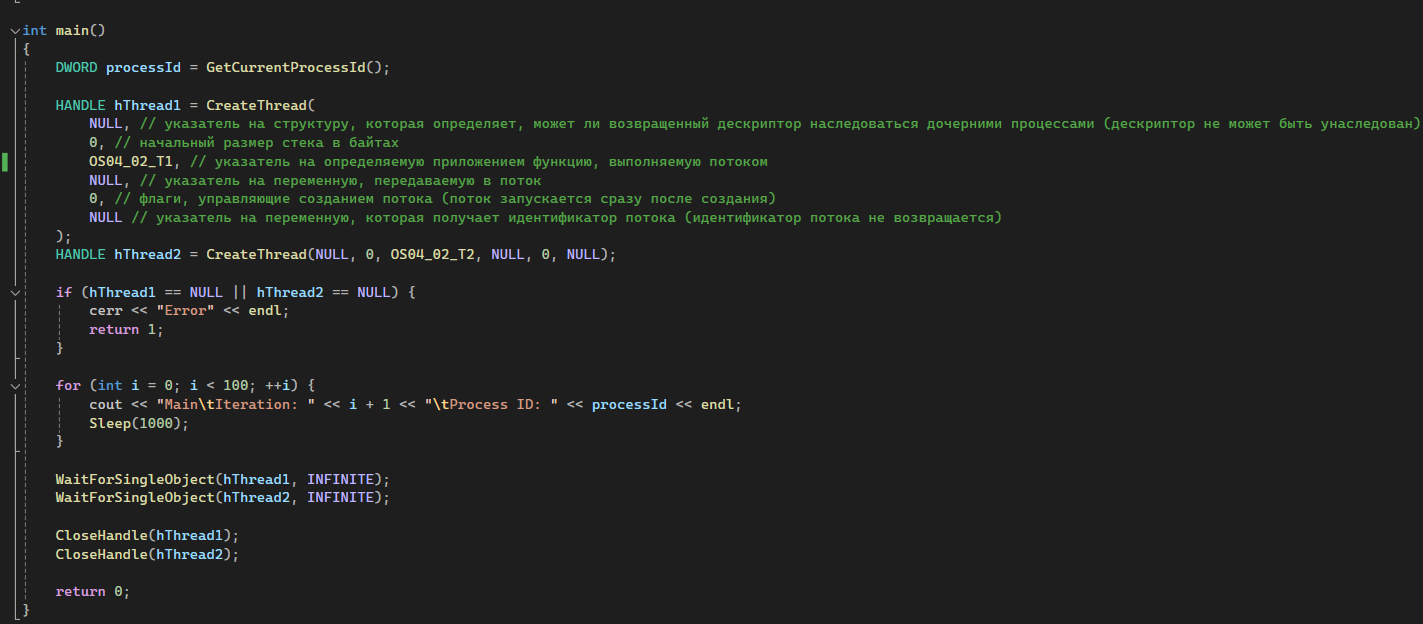
**Задание 02**

3. Разработайте консольное Windows-приложение OS04\_02 на языке С++, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса.

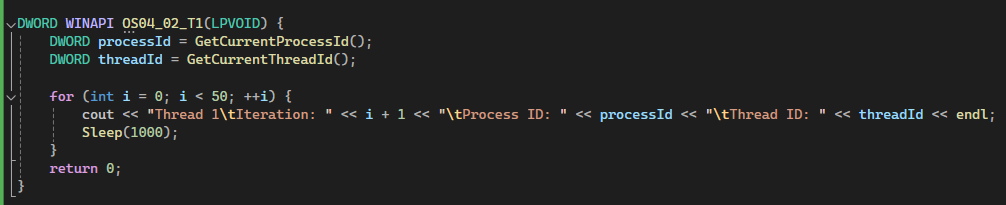




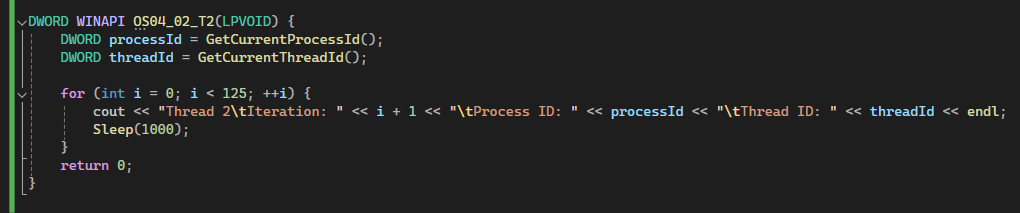
4. Процесс OS04\_02 должен создать два потока: потоковые функции OS04\_02\_T1, OS04\_02\_T2.



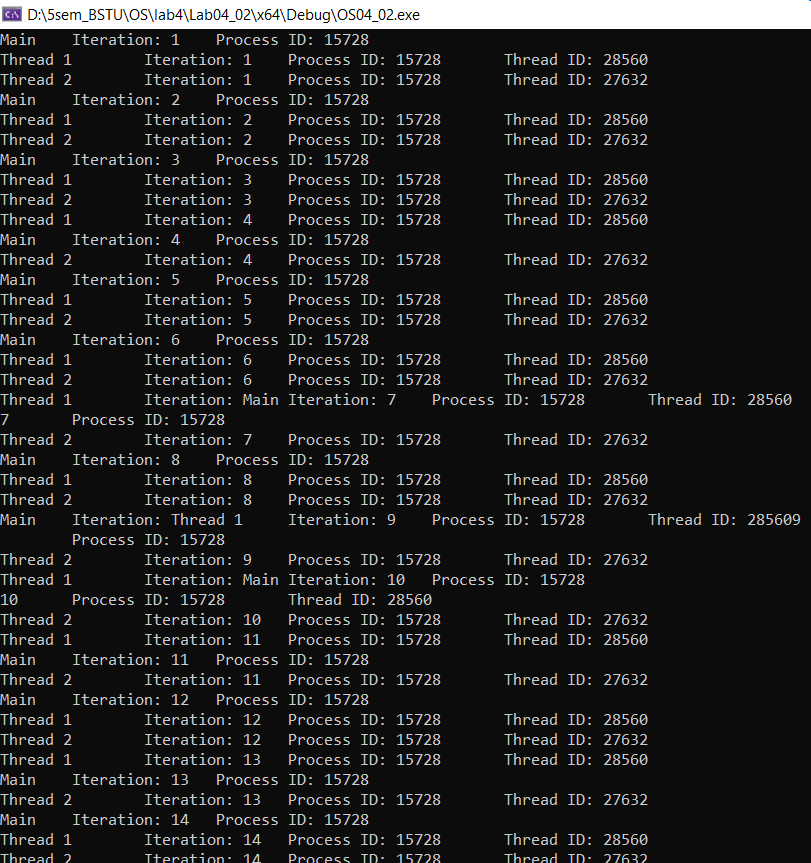
5. Поток OS04\_02\_T1 - выполняет цикл 50 итераций с временной задержкой в 1 сек. с выводом на консоль идентификаторов процесса и потока.

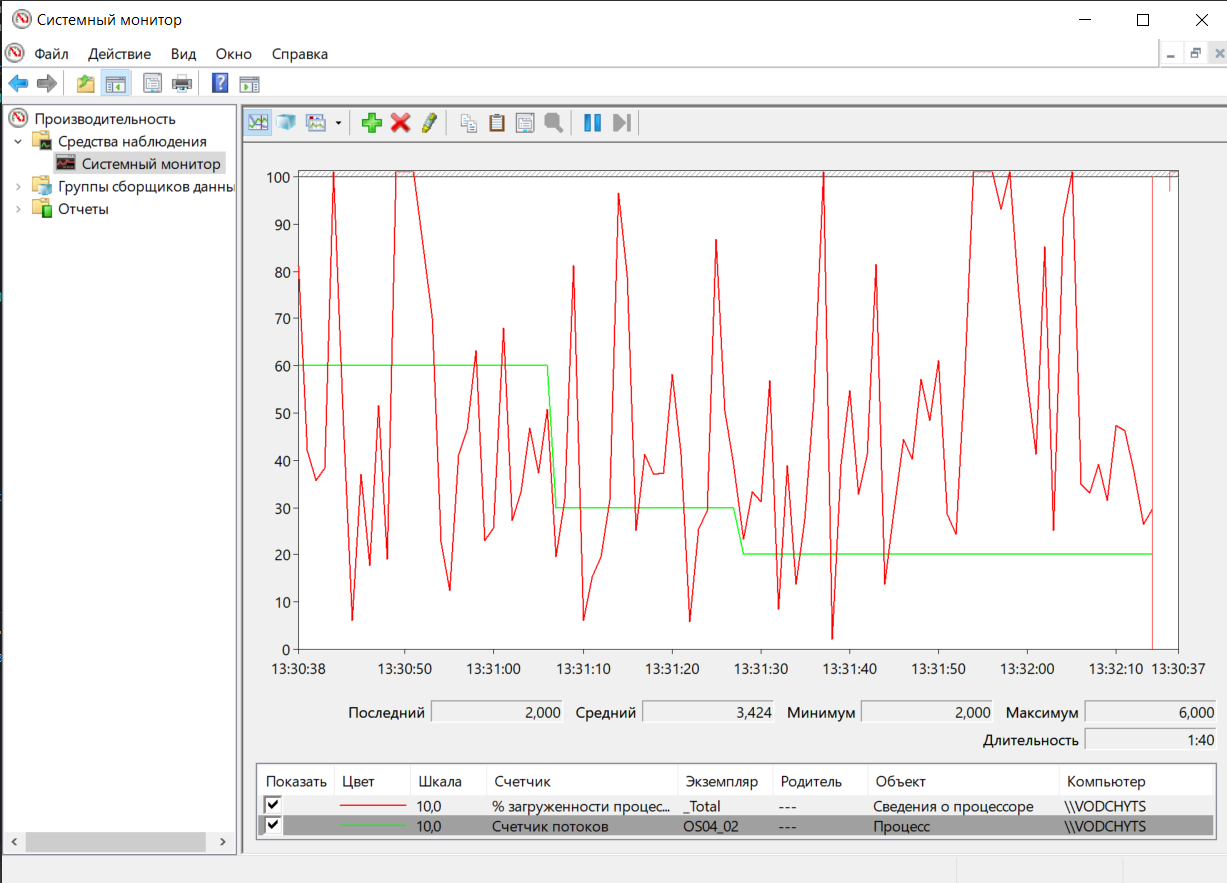


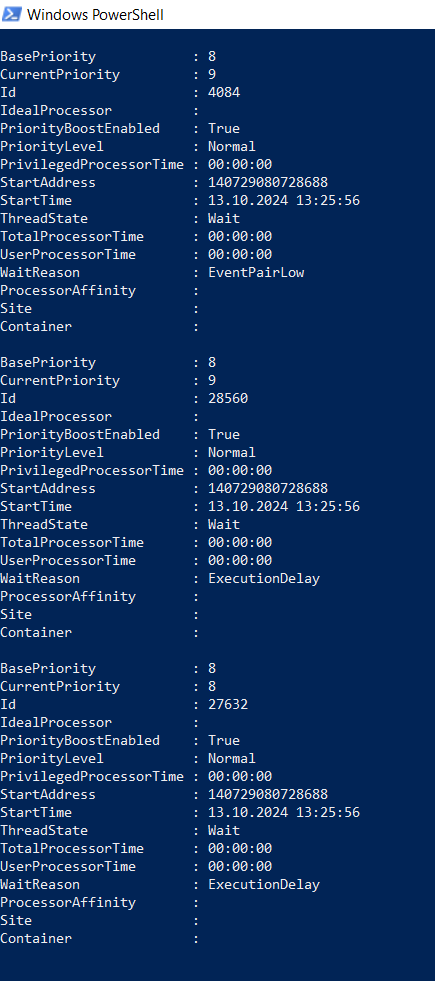
6. Поток OS04\_02\_T2 - выполняет цикл 125 итераций с временной задержкой в 1 сек. с выводом на консоль идентификаторов процесса и потока.



7. Продемонстрируйте информацию об потоках процесса OS04\_02 с помощью утилит PowerShell ISE и Performance Monitor.

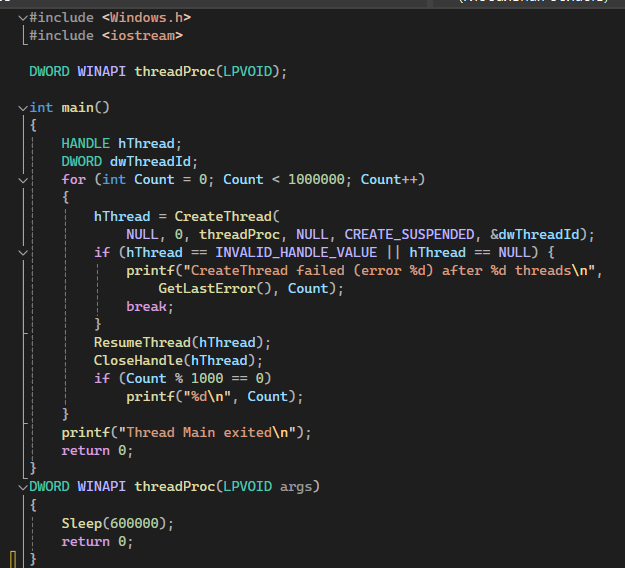


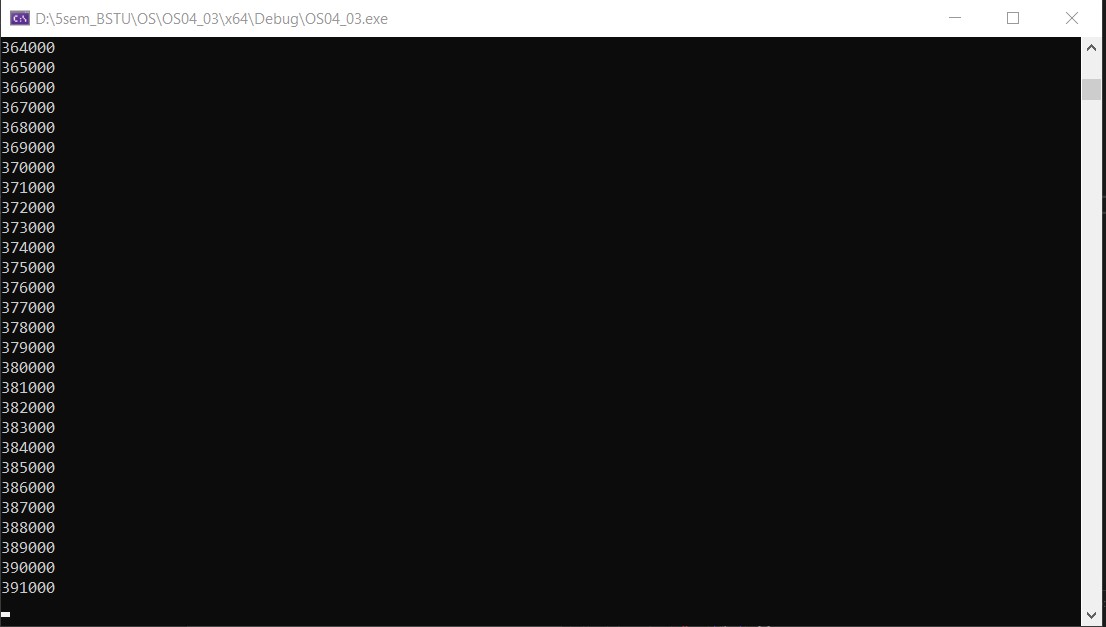




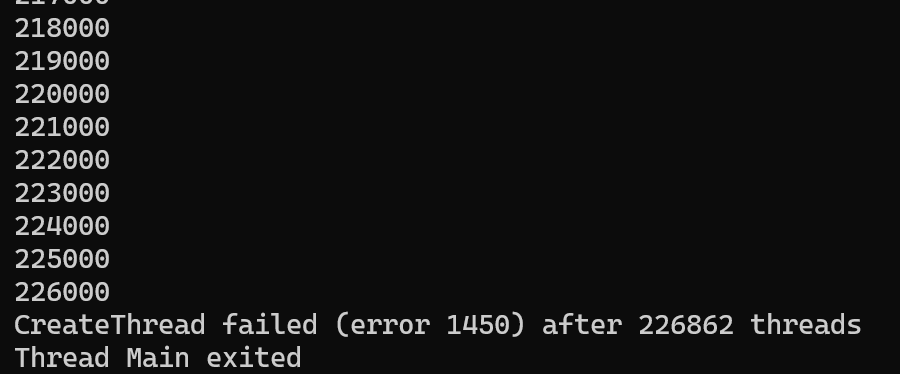
**Задание 03**

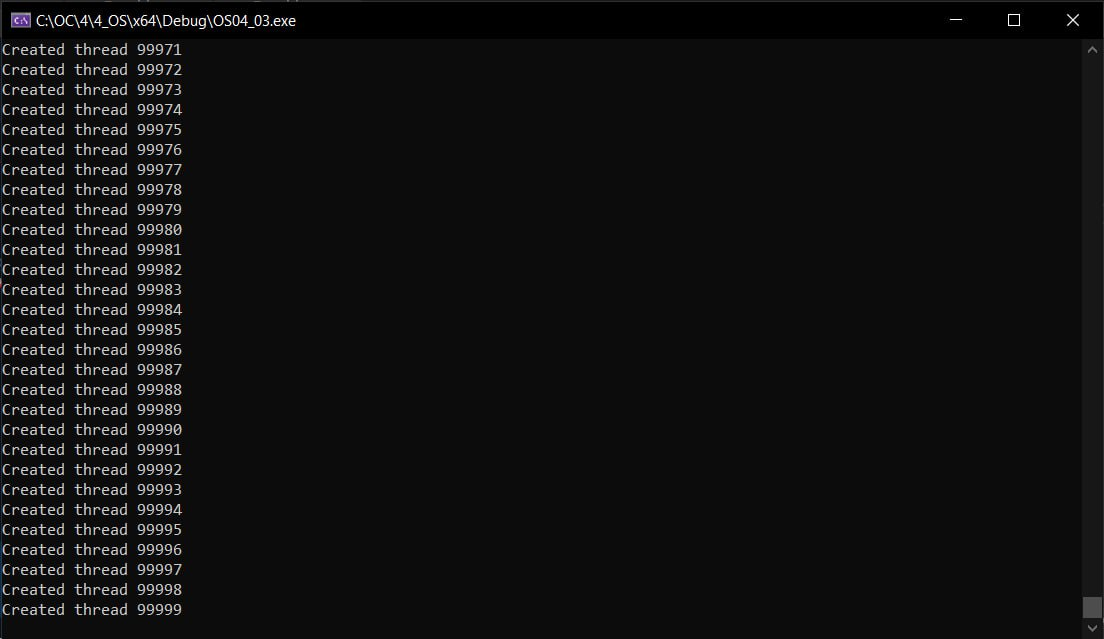
8. Определите, какое максимальное количество потоков можно создать в одном процессе на вашем компьютере, используя нижеприведенный программный код (или аналогичный). Сравните результаты между собой. От чего зависит 3 максимальное количество потоков? Какое ограничение накладывает операционная система.





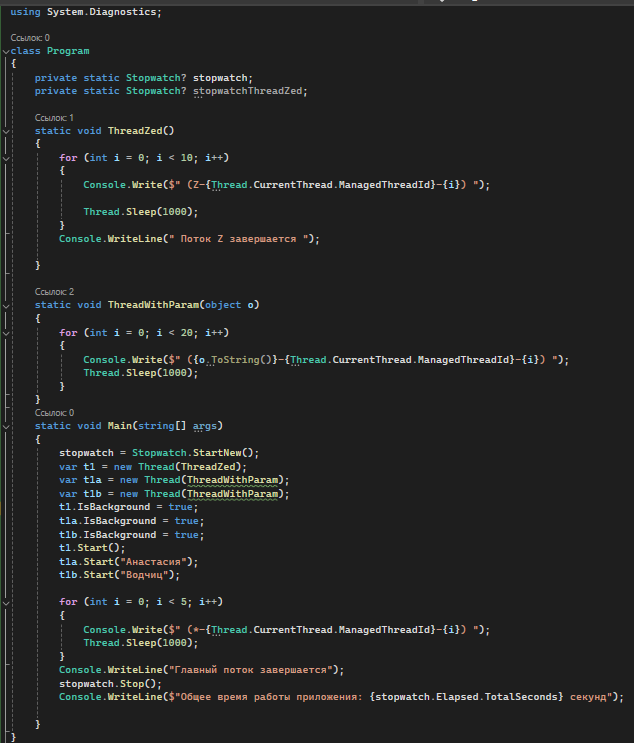
Другие результаты:



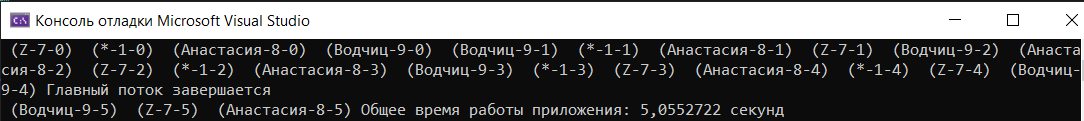


**Задание 04**

9. Создайте консольное приложение на C#, которое запускает три дополнительных потока и завершается через пять секунд. Один дополнительный поток завершается через 10 секунд, остальные два – через двадцать секунд.

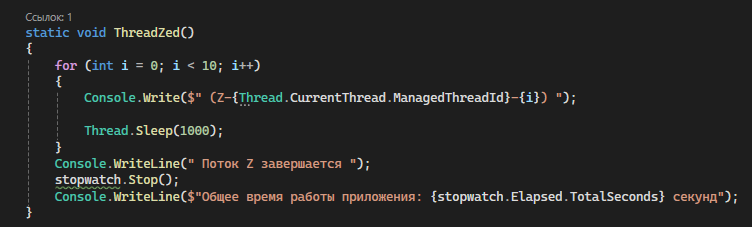


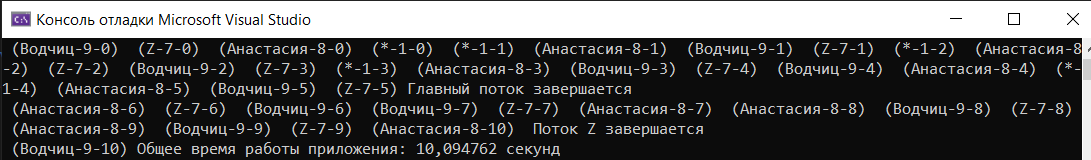
10. Выполните приложение. Обратите внимание на общее время работы приложения.



11. Измените значение свойства IsBackground для первого дополнительного потока на false и снова выполните приложение. Обратите внимание на общее время работы приложения.

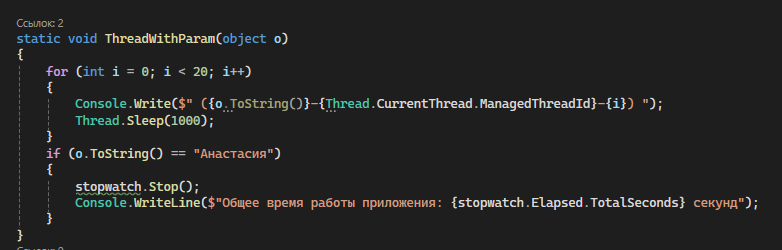


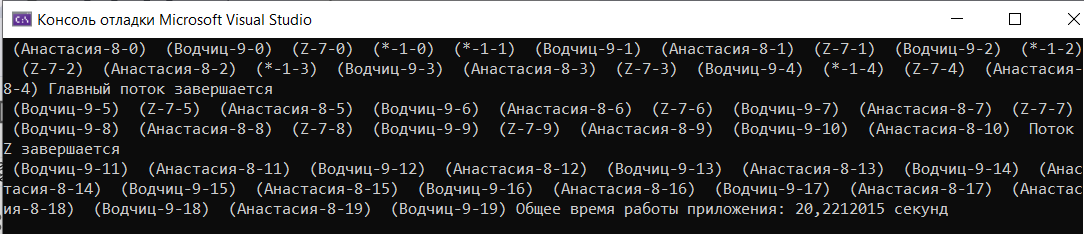




12. Измените значение свойства IsBackground для второго дополнительного потока на false и снова выполните приложение. Обратите внимание на общее время работы приложения.







13. Вставьте три скриншота в отчет и объясните результат.

Управляемый поток может быть основным или фоновым. Фоновые потоки отличаются от основных только в одном аспекте: фоновый поток не поддерживает работу управляемой среды выполнения. После того, как остановятся все основные потоки, система принудительно останавливает все фоновые потоки и завершает работу процесса.

Основной поток выполняет свои действия и завершает работу после 5 итераций цикла, не дожидаясь завершения фоновых потоков t1, t1a и t1b. Поэтому, когда основной поток завершает свою работу, приложение останавливается, и фоновые потоки не успевают завершить свои действия. Время выполнения: 5,055.

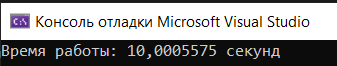
IsBackground = false означает, что поток теперь является основным. Приложение не завершится, пока все основные потоки не завершат свою работу. Если t1 не является фоновым, он будет продолжать выполняться даже после того, как основной поток завершит свои операции. Время выполнения: 10,09.

Если также сделать t1a основным потоком, общее время увеличится до 20 секунд, потому что оба потока будут работать до своего завершения.

**Задание 05 (вспомогательное)**

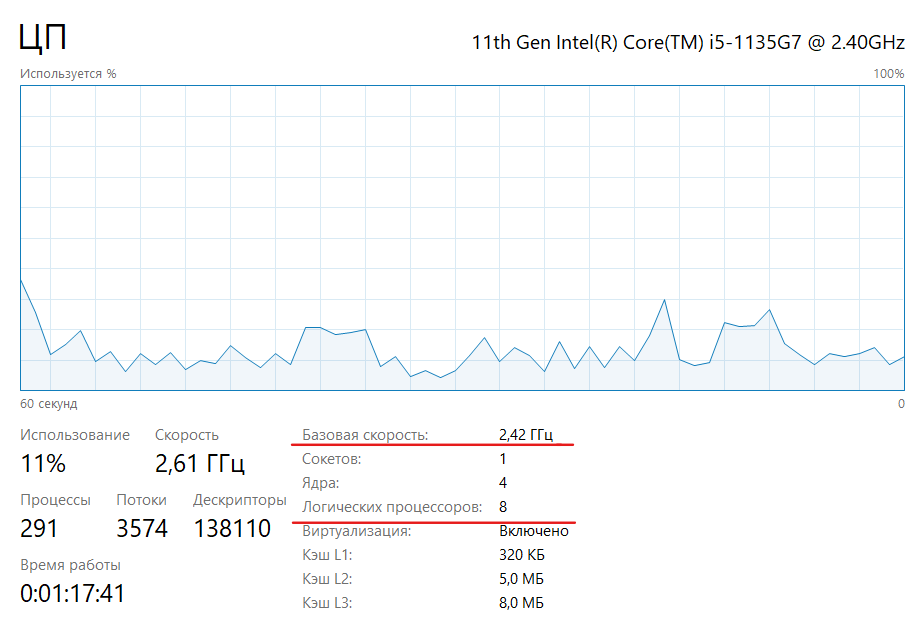
15. Создайте функцию, которая производит ЛЮБЫЕ вычисления длительностью n миллисекунд на вашем компьютере (для последующих заданий метод Thread.Sleep(n) не подходит, так как он освобождает центральный процессор и ничего не делает). Убедитесь, что MySleep(10000) работает ровно 10 секунд.





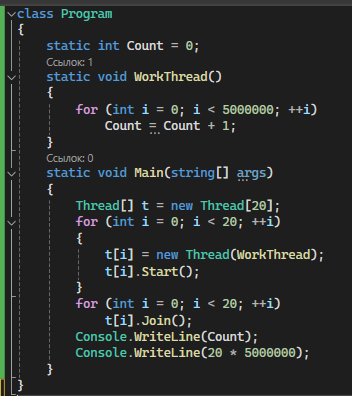
16. Узнайте количество ядер и логических процессоров в вашем компьютере.

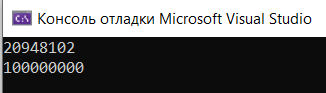
Запустите Диспетчер задач Ctrl+Shift+Esc. Перейдите на закладку Производительность и посмотрите, сколько в вашем компьютере имеется процессорных ядер и логических процессоров.



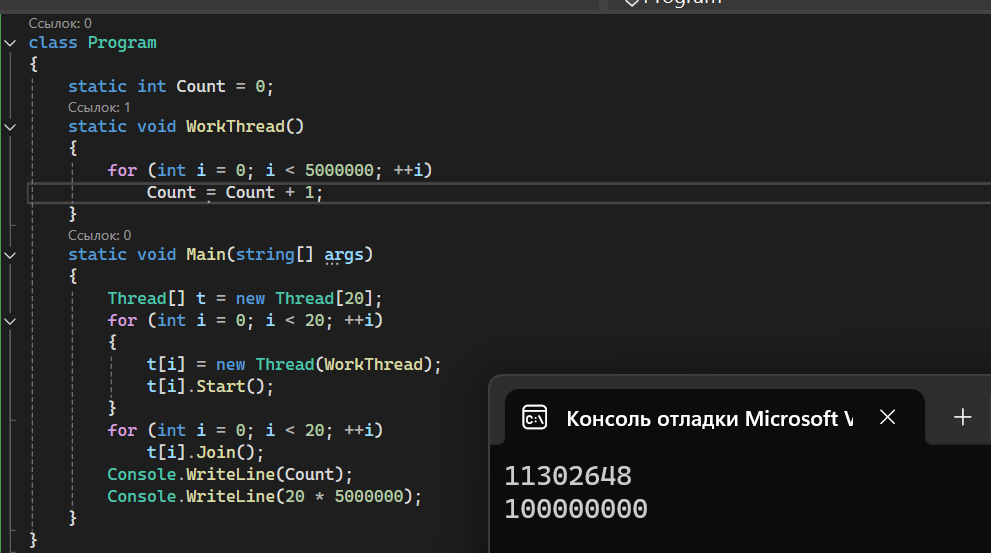
**Задание 06**

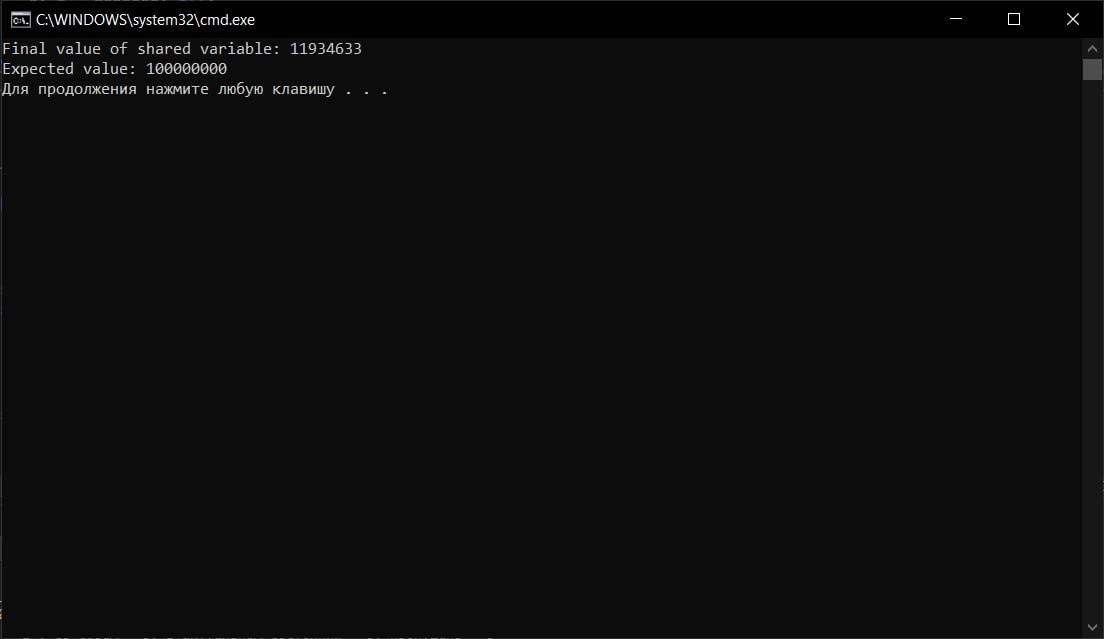
17. Разработайте консольное приложение OS04\_06 на языке С#, запускающее 20 потоков, каждый из которых в цикле 5000000 раз увеличивает на единицу значение общей для всех потоков переменной. Исходное значение переменной — ноль. Выведите результат и сравните с произведением 20х5000000.





18. Сравните результаты в группе. Есть ли какая закономерность?



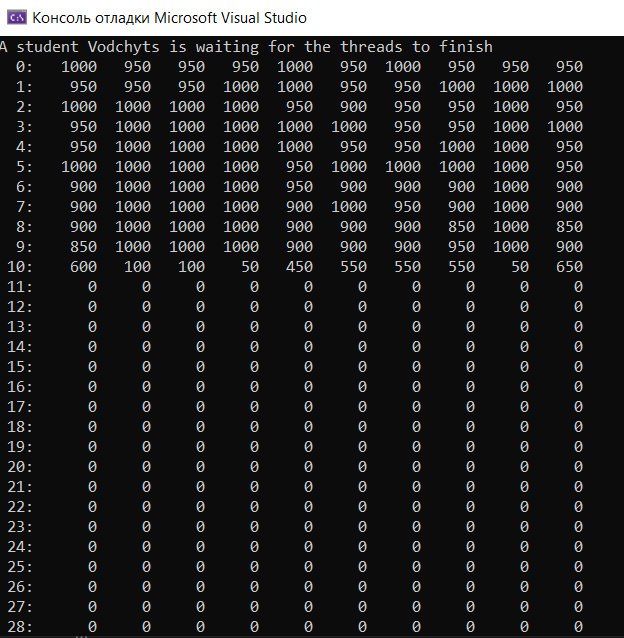


В коде проблема заключается в том, что переменная Count используется несколькими потоками одновременно без синхронизации. Это приводит к состоянию гонки, где несколько потоков могут одновременно читать и записывать значение Count, что может привести к неправильным результатам.

Когда один поток увеличивает Count, другой поток может прочитать его значение до того, как первый поток завершит операцию. В результате некоторые инкременты могут быть потеряны.

**Задание 07**

19. Разработайте консольное приложение OS04\_07 на языке С#, запускающее N потоков, каждый из которых будет производить вычисления t секунд (использовать разработанный в задании 5 метод), используя класс System.Threading.Thread. Сохраните информацию о работе потоков в течение T секунд и выведите на экран в виде таблицы < >. Подберите подходящие параметры в зависимости от количества логических процессоров в вашем компьютере (например, для четырех логических процессоров N = 10, t = 10, T=30).



1. Первые 10 секунд (строки 0-9):

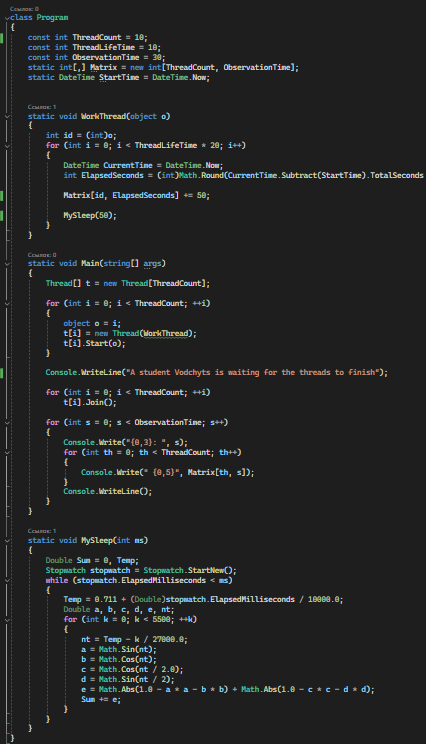
* В первые 10 секунд потоки активно выполняют свою работу.
* Все потоки работают относительно равномерно, каждый выполняет примерно 950–1000 единиц работы за каждую секунду.

1. 11-я секунда (строка 10):

* В этой строке видно резкое снижение работы потоков.
* Это может свидетельствовать о том, что потоки начинают завершать работу.

1. После 11-й секунды (строки 11-29):

* Все потоки завершили свою работу, так как значения для всех потоков равны нулю. К этому моменту (примерно на 11-й секунде) все потоки закончили выполнение, и программа больше не фиксирует их активность.

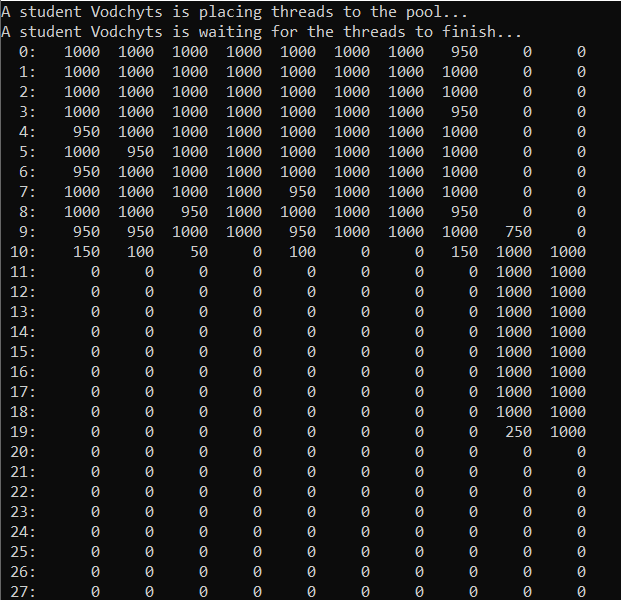


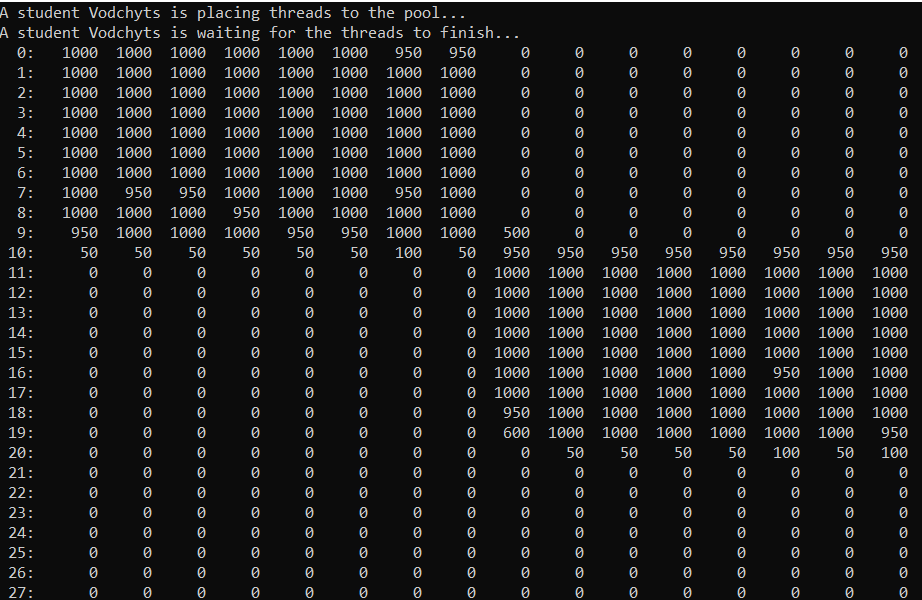
**Задание 08**

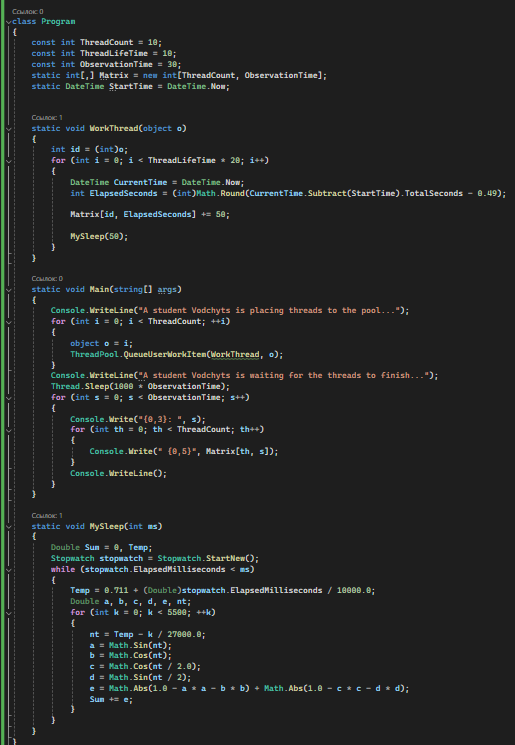
20. Скопируйте консольное приложение OS04\_07 как OS04\_08. Теперь используйте пул потоков. Выведите статистику работы потоков на экран в виде таблицы

Пул потоков — это механизм, который управляет коллекцией потоков, позволяя повторно использовать их для выполнения задач. Вместо создания отдельных потоков, теперь используется ThreadPool.QueueUserWorkItem. ThreadPool позволяет повторно использовать потоки для выполнения задач, что помогает избежать накладных расходов на создание и уничтожение потоков, особенно в сценариях с высокой нагрузкой.

Эффективное количество потоков в пуле часто зависит от количества логических процессоров в системе. Как правило, рекомендуется использовать количество потоков, равное количеству логических процессоров, умноженному на некий коэффициент, чтобы обеспечить максимальную эффективность.

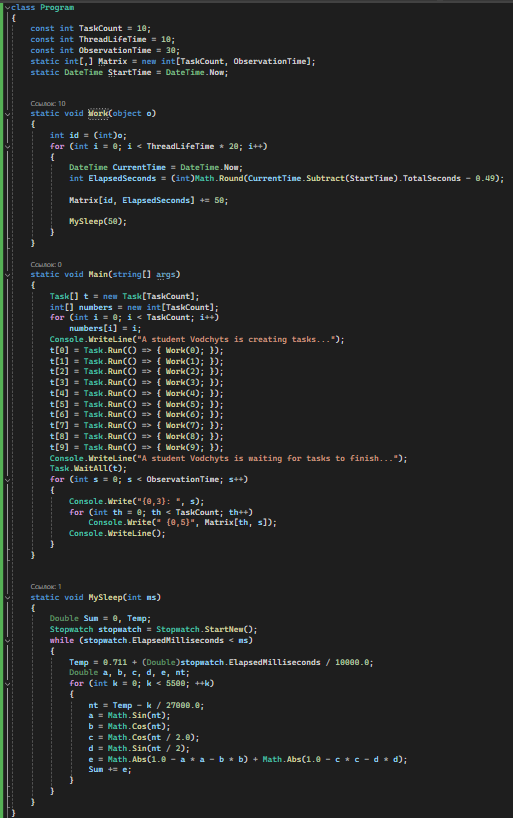


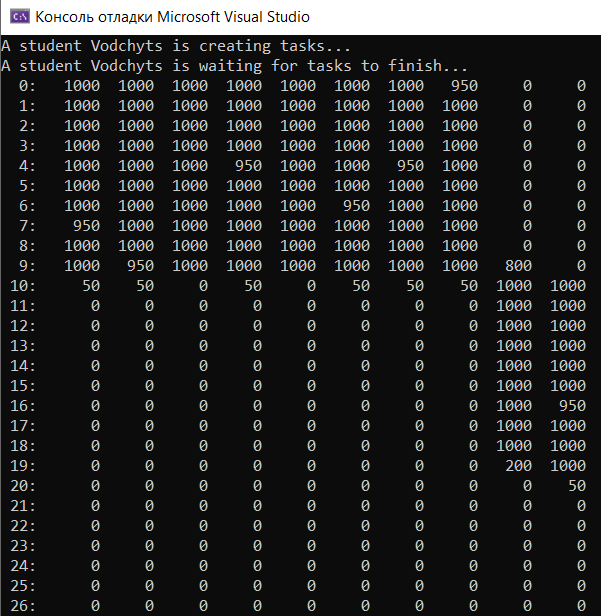


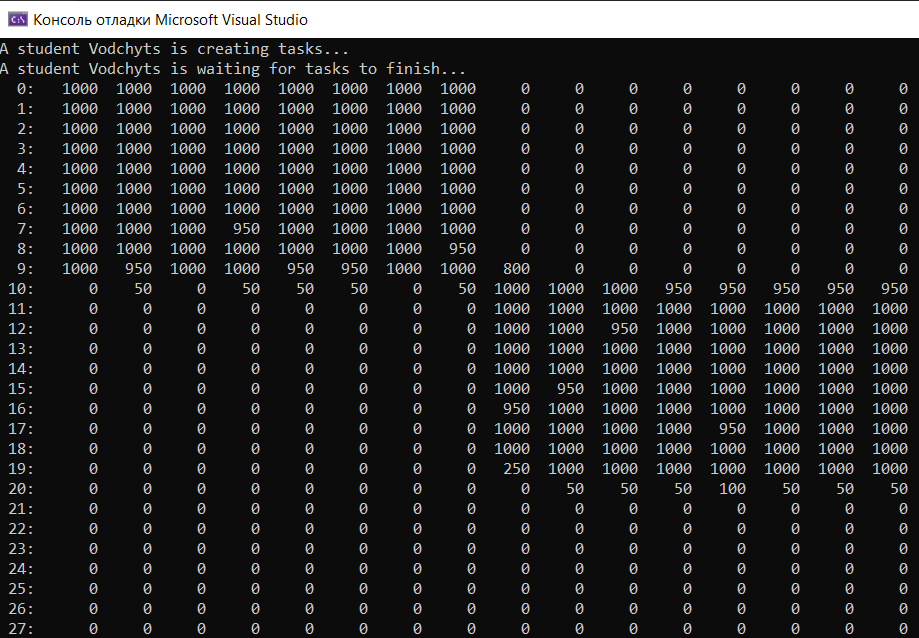


**Задание 09**

21. Скопируйте консольное приложение OS04\_07 как OS04\_09. На этот раз используйте System.Threading.Tasks.Task. Выведите статистику работы потоков на экран в виде таблицы. Сравните результаты заданий 7-9 и запишите вывод в отчет.



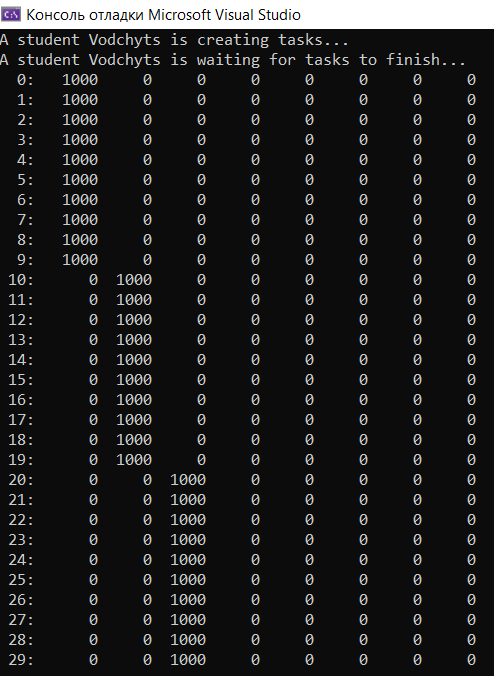


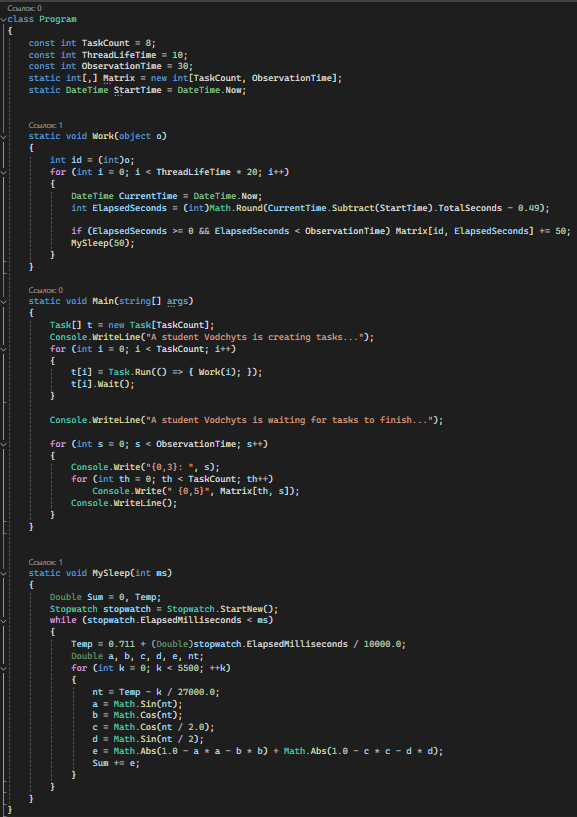


Задачи — это способ выполнения работы в фоновом режиме, не блокируя основной поток программы. Когда мы создаем задачи, нам не нужно беспокоиться о том, как управлять потоками вручную. Задачи делают это за нас. Задачи используют пул потоков для управления выполнением. Это означает, что когда задача завершает свою работу, поток может быть повторно использован для выполнения другой задачи.

**Задание 10**

23. Скопируйте консольное приложение OS04\_09 как OS04\_10. Уменьшите количество задач до количества логических процессоров. Организуйте выполнение задач по очереди. Выведите статистику работы потоков на экран в виде таблицы.

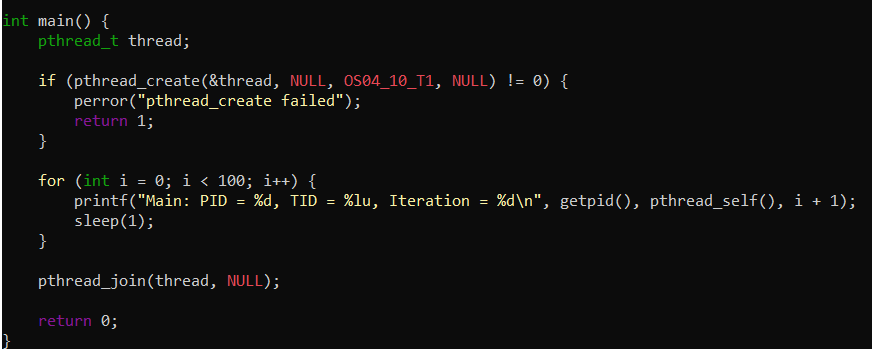




**Задание 11**

24. Разработайте на языке консольное Linux-приложение OS04\_10 на языке С, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса (использовать функции из pthread.h).

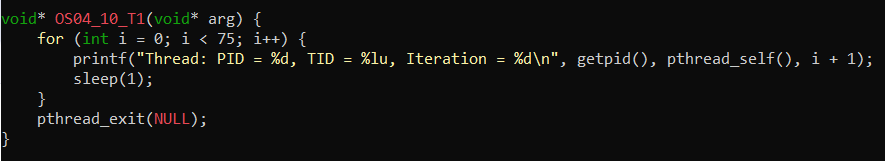
25. Процесс OS04\_10 должен создать поток: потоковая функция OS04\_10\_T1.



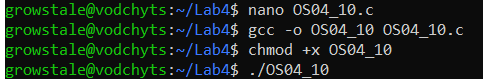
pthread\_create(&thread, NULL, OS04\_10\_T1, NULL). Создаёт новый поток. Аргументы:

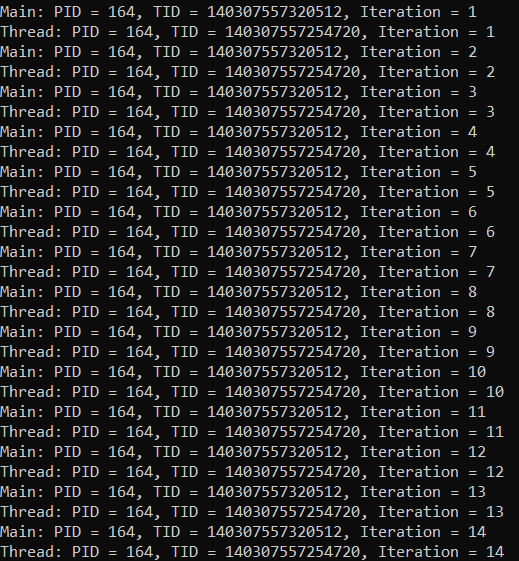
* &thread — указатель на переменную, куда будет записан идентификатор потока.
* NULL — атрибуты потока. Можно передать настройки для потоков, но здесь используются стандартные.
* OS04\_10\_T1 — имя функции, которая будет выполняться в новом потоке.
* NULL — аргумент для потоковой функции. В данном случае поток не получает аргументов.

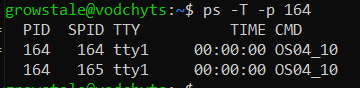
26. Поток OS04\_10\_T1 - выполняет цикл 75 итераций с временной задержкой в 1 сек. с выводом на консоль идентификаторов процесса.



27. Продемонстрируйте информацию о потоках процесса OS04\_10 с помощью утилиты ps.







ps — команда для отображения информации о процессах, выполняющихся в системе.

-T — опция, которая показывает все потоки для каждого процесса.

-p 164 — опция, которая фильтрует результат и показывает информацию только для процесса с PID 164.

**Задание 12. Ответьте на следующие вопросы**

**28. Что такое поток управления OS?**

Объект ядра операционной системы, которому ОС выделяет процессорное время, наименьшая единица работы ядра ОС.

Последовательность инструкций, выполняемых процессором в выделенные ОС интервалы времени. При создании процесса в нём есть как минимум один основной поток.

**29. С помощью каких системных вызовов создаются потоки в Windows и Linux?**

CreateThread (Windows)

pthread\_create (Linux)

**30. Что такое системные и пользовательские потоки?**

**Системные потоки** (потоки ядра ОС) – выполняют различные сервисы ОС и запускаются ядром ОС, используются для реализации пользовательских потоков.

**Пользовательские потоки** – потоки, служащие для решения задач пользователя, и запускаемые приложением.

**31. Что такое многопоточность?**

Это способность ЦП, позволяющая выполнять 2 или более процесса/потока с инструкциями независимо друг от друга, используя ресурсы одного процессора и разделяя между ними процессорное время.

**32. Что такое контекст потока и для чего он нужен?**

Контекст потока – данные, необходимые для возобновления работы потока при его приостановке, а именно:

Программный код, набор регистров, стек памяти, оперативная память, стек ядра и маркер доступа.

**33. Перечислите состояния, в которых может быть поток и поясните их назначение.**

* [5] New – поток создан
* [5] Ready – готов к исполнению
* [5] Running – исполняется
* [12] Sleeping – заснул на некоторое время
* [12] Sleeping suspended – спит и приостановлен до события
* [7] Suspended ready – готов и приостановлен до события
* [5] Blocked – заблокирован извне
* [7] Suspended blocked – заблокирован извне и приостановлен до события
* [5] Finish – поток завершил исполнение
* **Модель 5 состояний:**
* 
* **Модель 7 состояний:**
* 
* **Модель 12 состояний:**
* 

**34. Что такое LWP?**

LWP (light-weight process) – процесс, поддерживающий работу потока пользовательского пространства. Средство достижения многозадачности.

Несколько пользовательских потоков могут быть размещены в одном или нескольких легковесных процессах, что даёт многозадачность на уровне пользователя, которая может иметь некоторые преимущества в производительности.

**35. Что такое потокобезопасность программного кода?**

Потокобезопасность — это способность программы правильно работать в условиях многопоточности, когда несколько потоков (или легковесных процессов) выполняются одновременно. Потокобезопасный код гарантирует, что при параллельном доступе к общим данным или ресурсам не произойдут ошибки или некорректные результаты.

**36. Что такое реентерабельность кода?**

Компьютерная программа в целом или её отдельная процедура называется **реентерабельной**, если она разработана таким образом, что одна и та же копия инструкций программы в памяти может быть совместно использована несколькими пользователями или процессами. При этом второй пользователь может вызвать реентерабельный код до того, как с ним завершит работу первый пользователь и это не должно привести к ошибке или потере данных.

**Реентерабельность** — свойство одной копии программного кода работать в нескольких потоках одновременно. Реентерабельный код всегда потокобезопасен. Реентерабельный код не использует статическую память и не изменяет сам себя, все данные сохраняются в динамической памяти.

**37. Что такое Fiber?**

Fiber (волокно) – механизм для ручного планирования выполнения кода в рамках потока. Находится внутри потоков (процессы –> потоки –> волокна) и является особенно легковесным потоком.