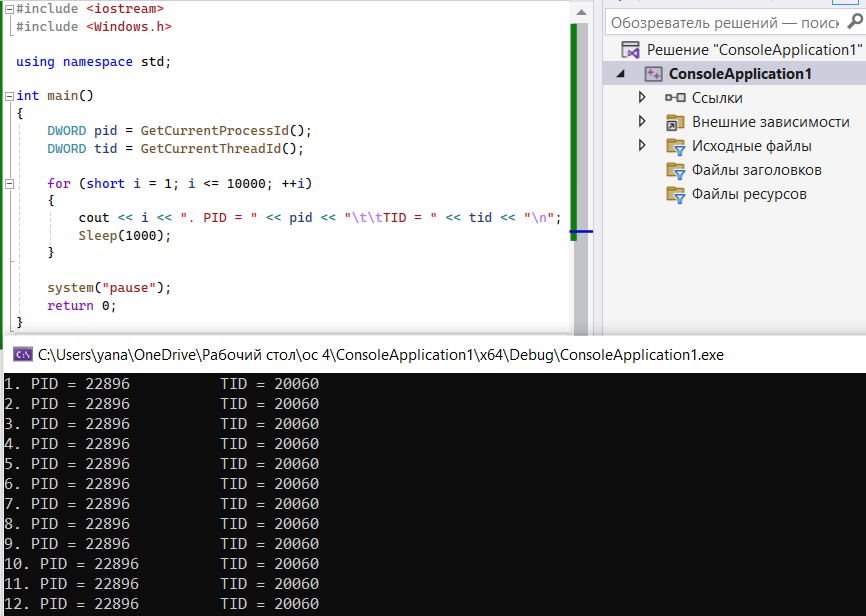
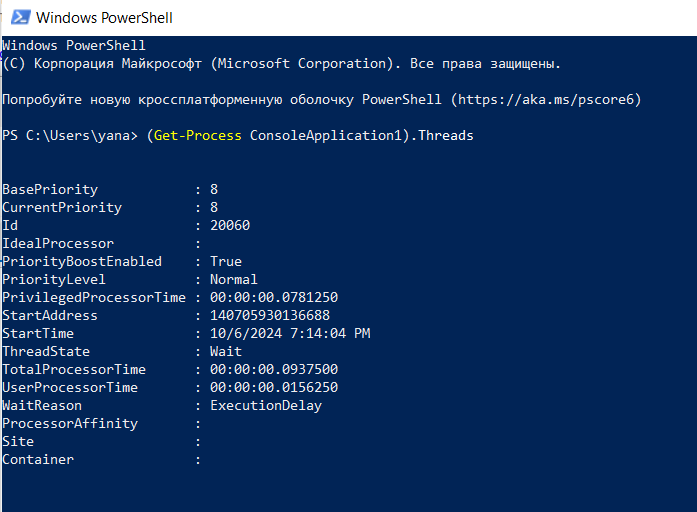
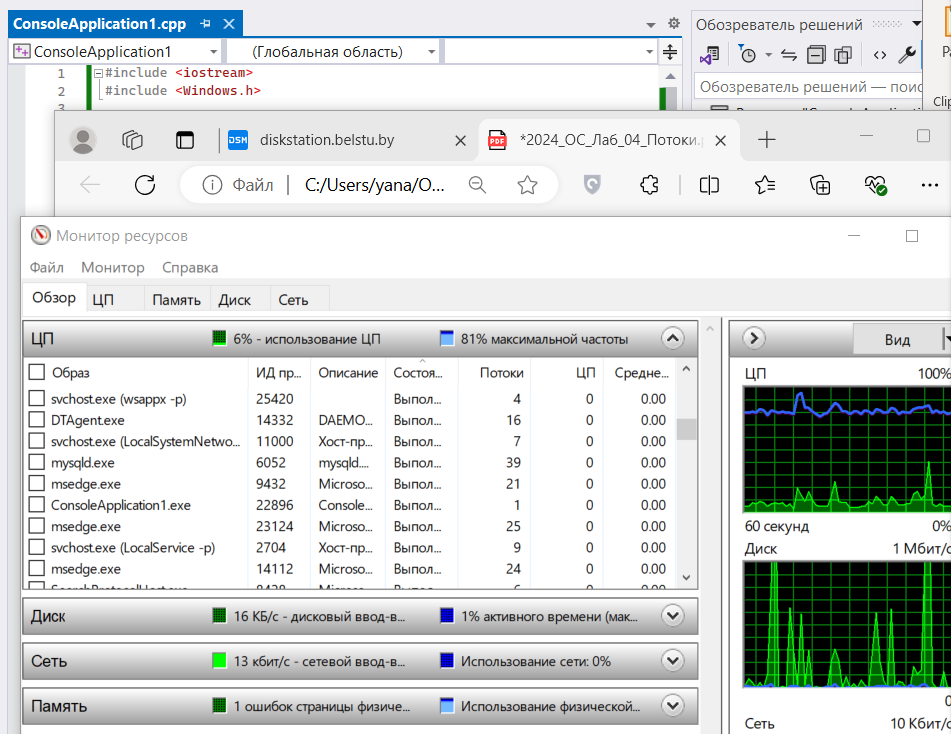
**Задание 01**

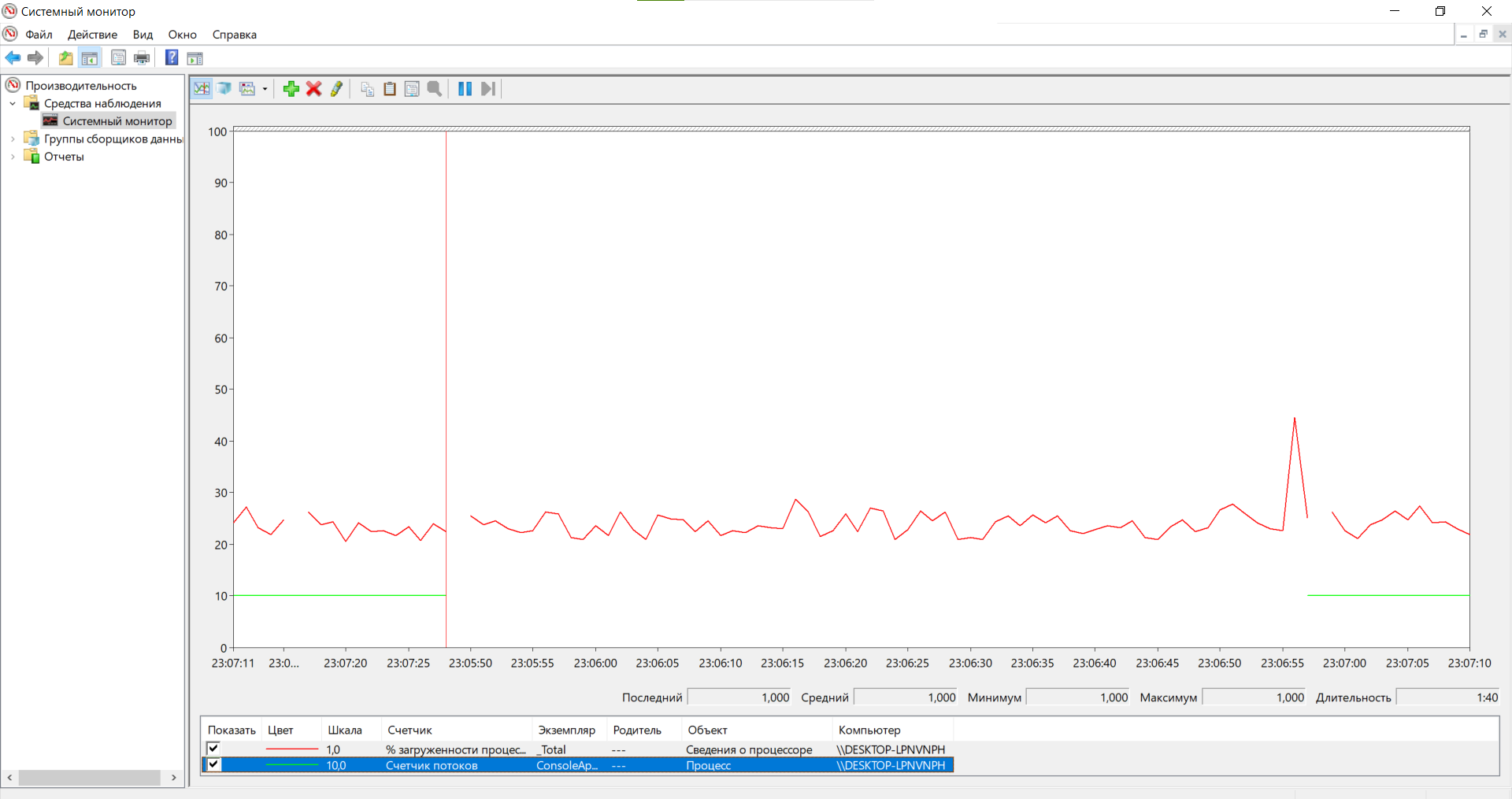
1. Разработайте консольное Windows-приложение OS04\_01 на языке С++, выполняющее длинный цикл с временной задержкой и с выводом на консоль идентификаторов текущего процесса и текущего потока.



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







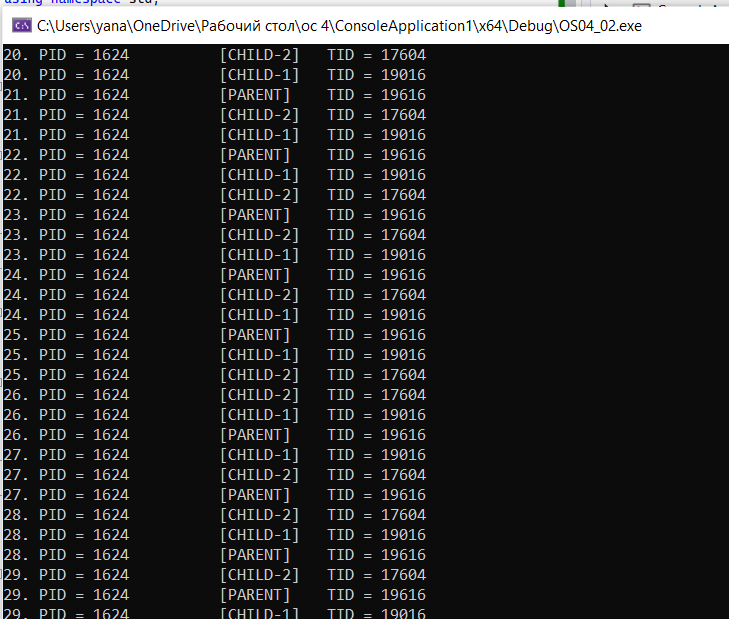
**Задание 02**

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.

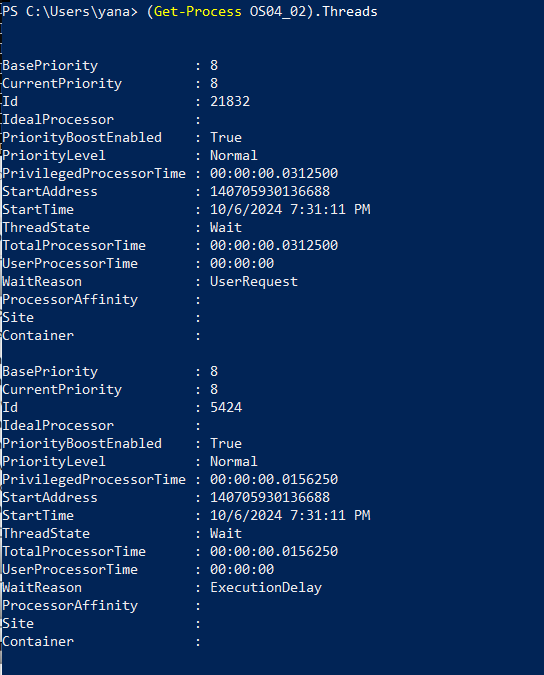
4. Процесс OS04\_02 должен создать два потока: потоковые функции OS04\_02\_T1, OS04\_02\_T2.

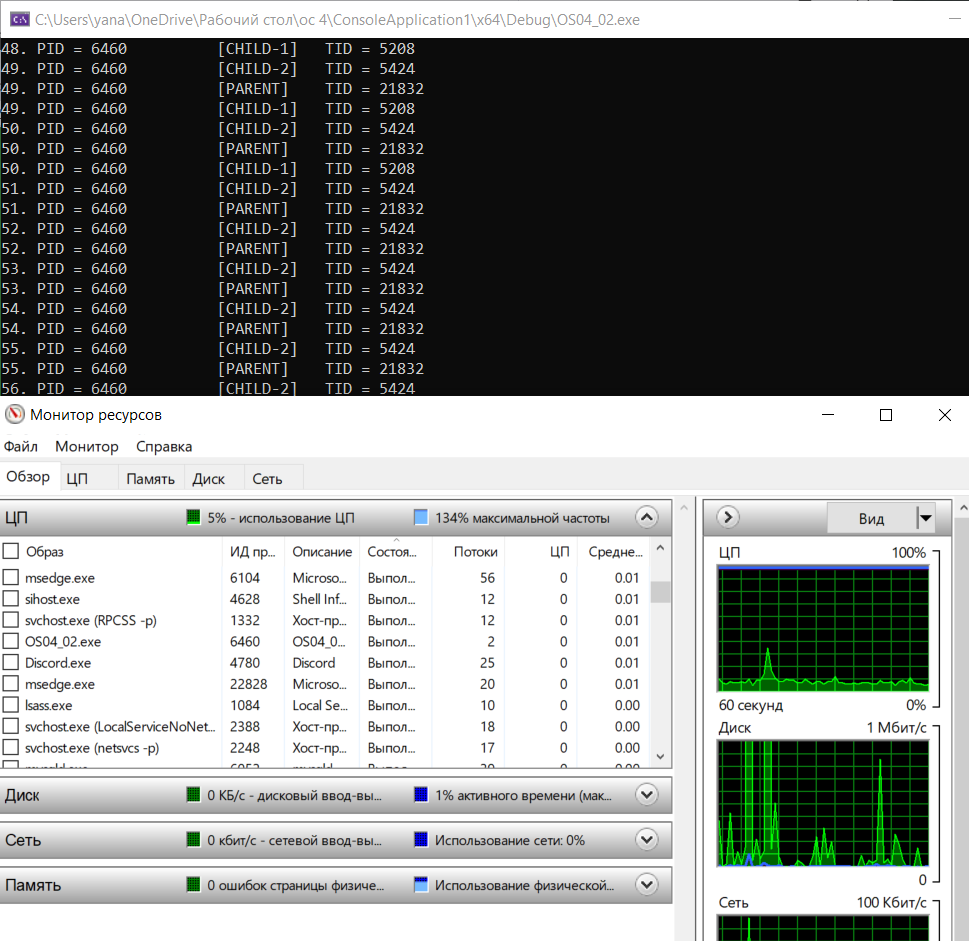
5. Поток OS04\_02\_T1 - выполняет цикл 50 итераций с временной задержкой в 1 сек. с выводом на консоль идентификаторов процесса и потока.

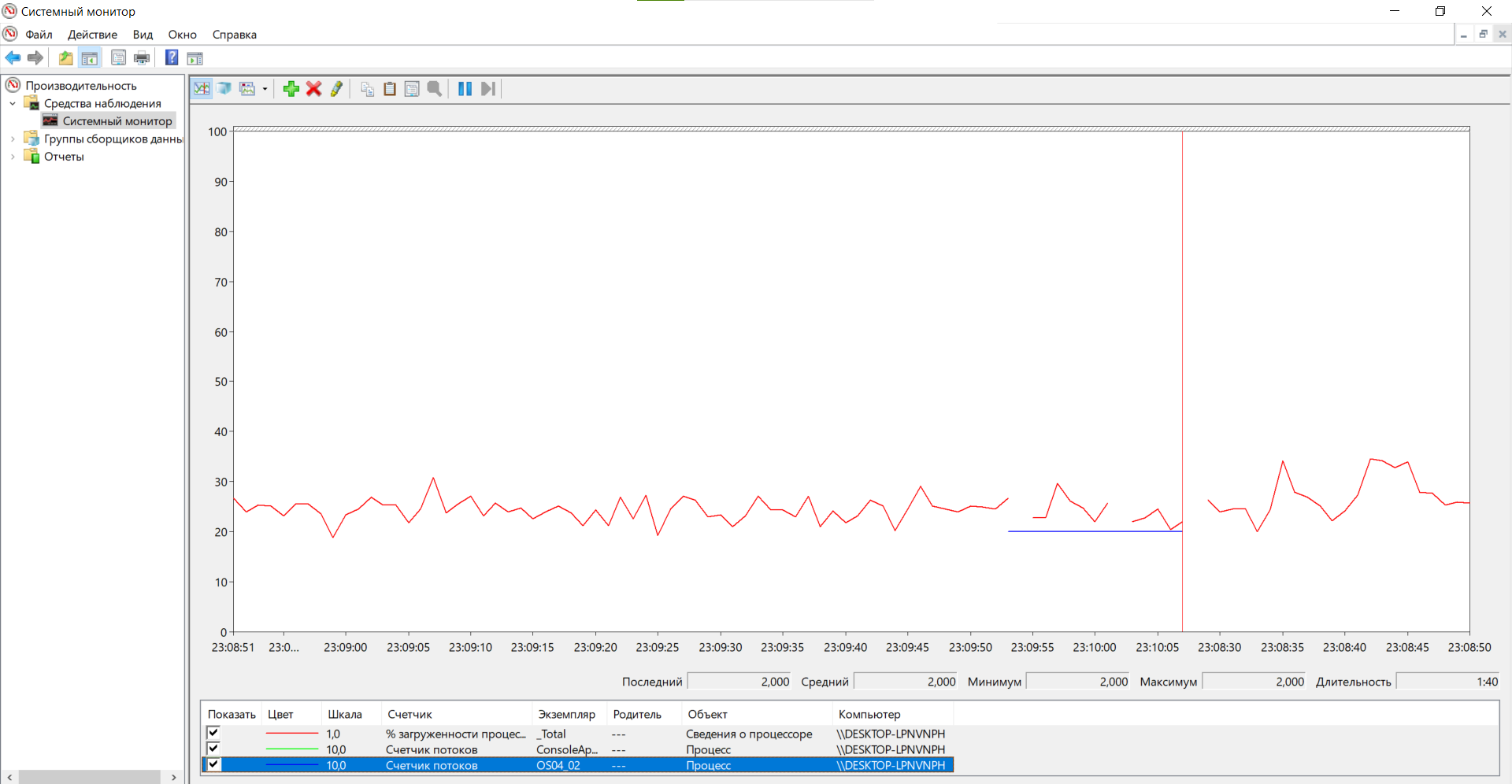
6. Поток OS04\_02\_T2 - выполняет цикл 125 итераций с временной задержкой в 1 сек. с выводом на консоль идентификаторов процесса и потока.



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



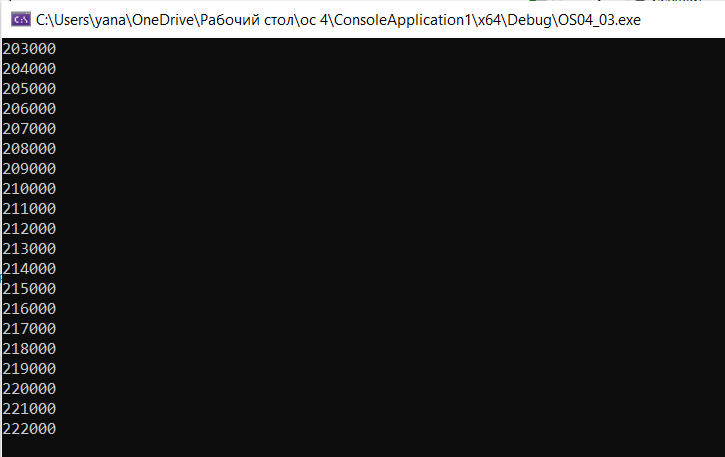




**Задание 03.**

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 максимальное количество потоков? Какое ограничение накладывает операционная система.

Максимальное количество потоков:



Максимальное количество потоков зависит от:

ресурсов системы ( оперативной памяти наприер), параметров операционной системы (некоторые ее настройки могут устанавливать ограничения), архитектура процессора (на 32-битных меньше чем на 64)

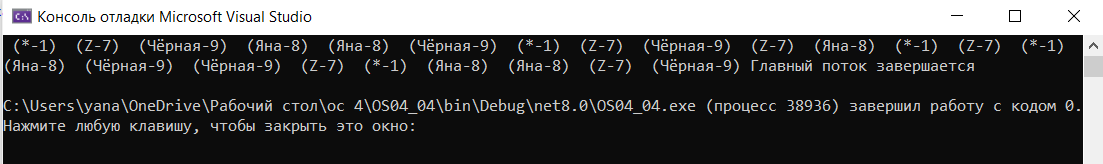
Операционная система накладывает следующие ограничения:

Каждвый поток использует определенное количество системных ресурсов, и есть ограничение на коливество потоков которые могут быть созданы в одном процессе

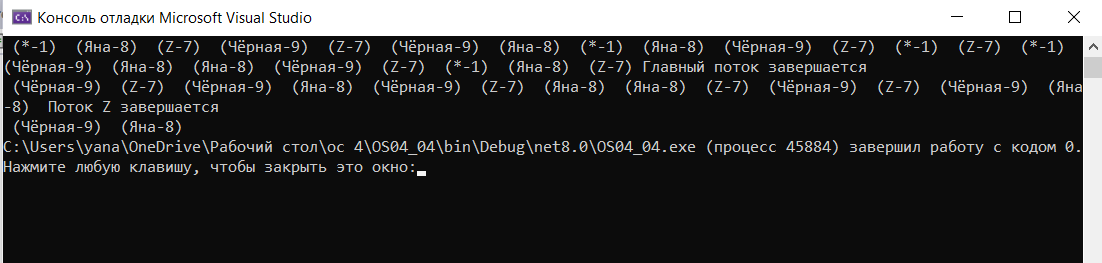
**Задание 04**

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.

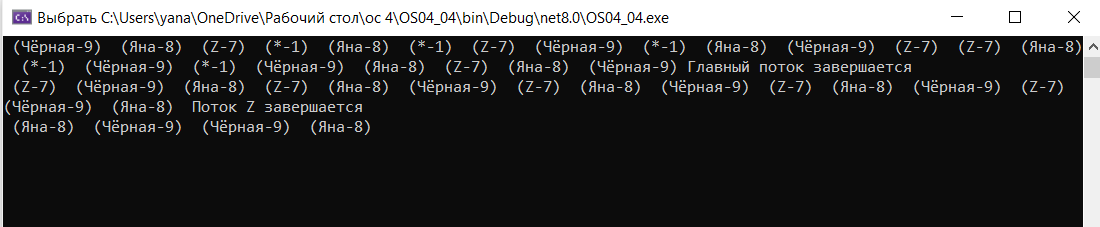
10. Выполните приложение. Обратите внимание на общее время работы приложения.



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.



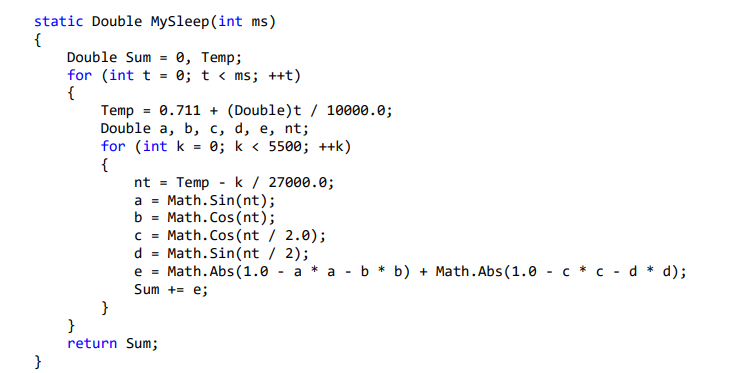
13. Вставьте три скриншота в отчет и объясните результат.

.IsBackground определяет, является поток фоновым или основным:

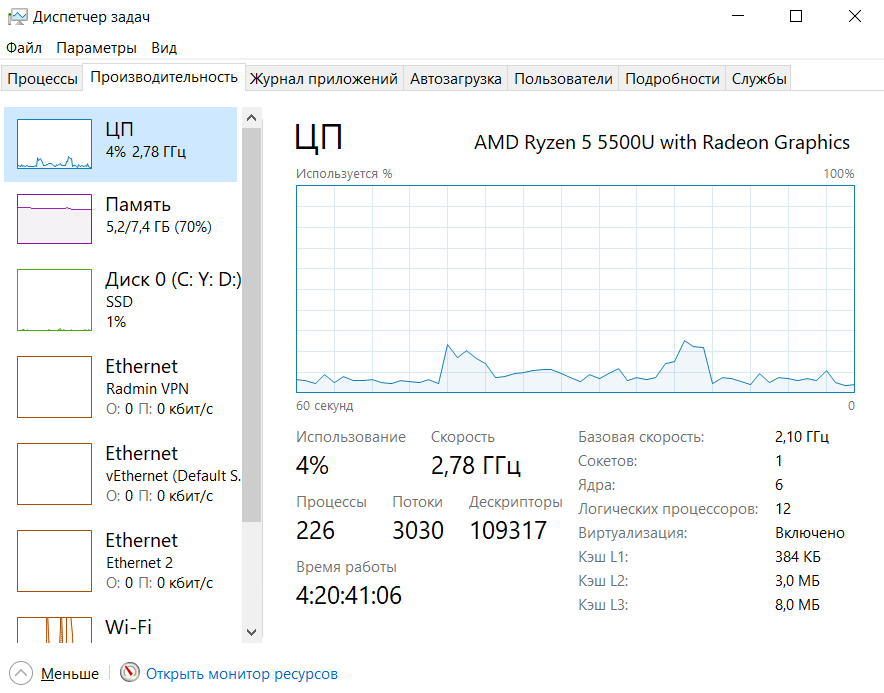
* **Фоновый поток**:
  + Не блокирует завершение приложения.
  + Используется для задач, которые могут продолжаться в фоновом режиме.
* **Основной поток**:
  + Блокирует завершение приложения.
  + Используется для критически важных задач, которые должны завершиться перед выходом.

**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).

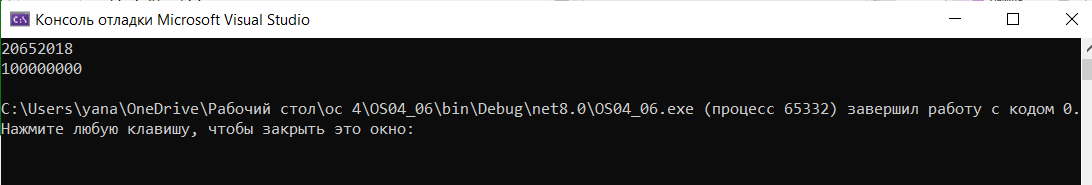


16. Узнайте количество ядер и логических процессоров в вашем компьютере (приложение 1).



**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.

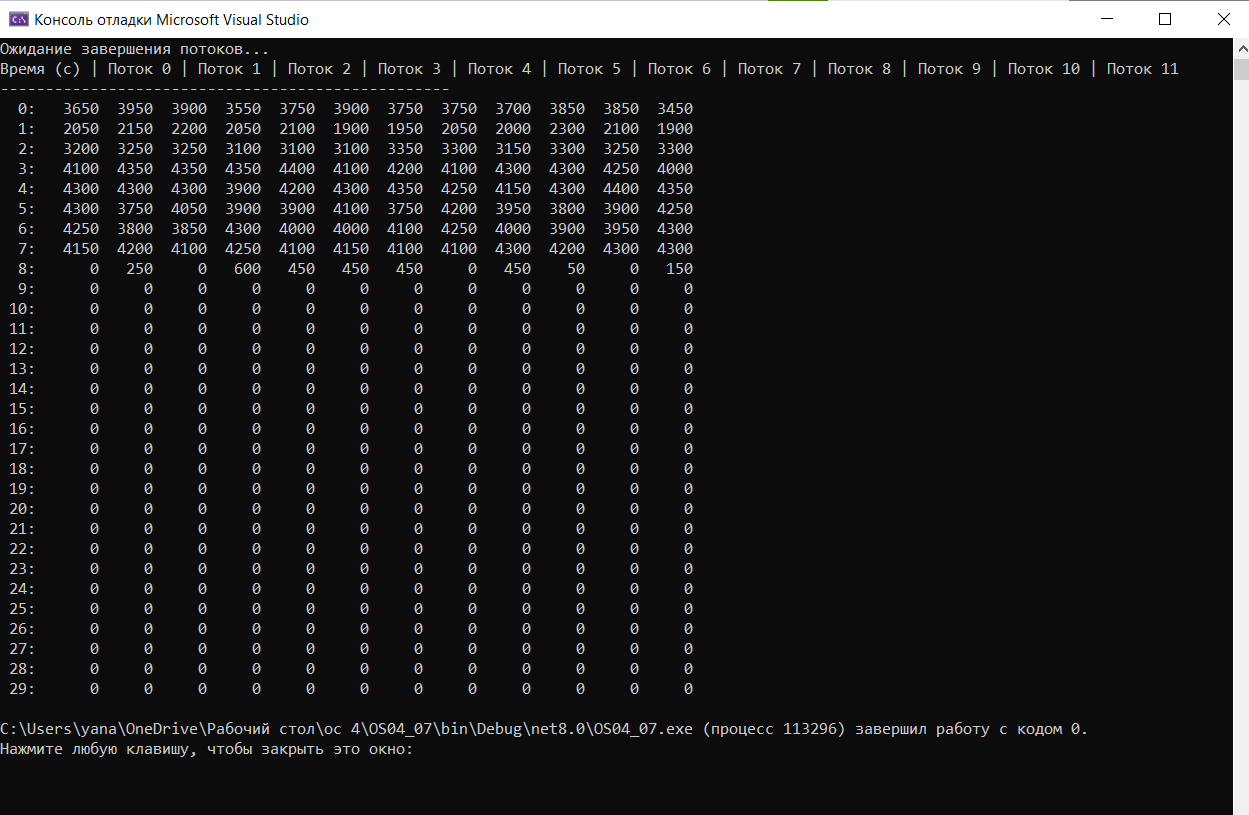


**Задание 07**

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. (позволяет выполнять потоки параллельно).

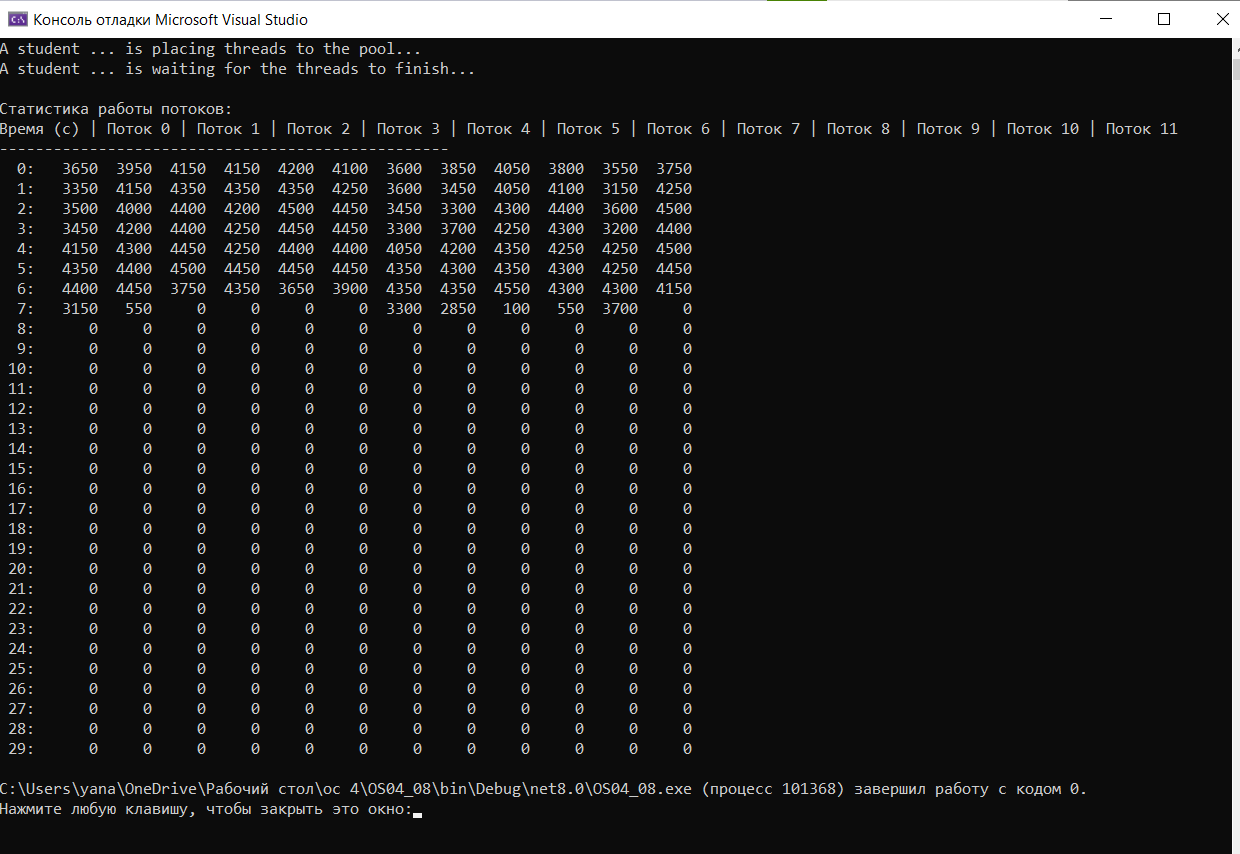
Сохраните информацию о работе потоков в течение T секунд и выведите на экран в виде таблицы < >.

Подберите подходящие параметры в зависимости от количества логических процессоров в вашем компьютере (например, для четырех логических процессоров N = 12, t = 9, T=30).



**Задание 08**

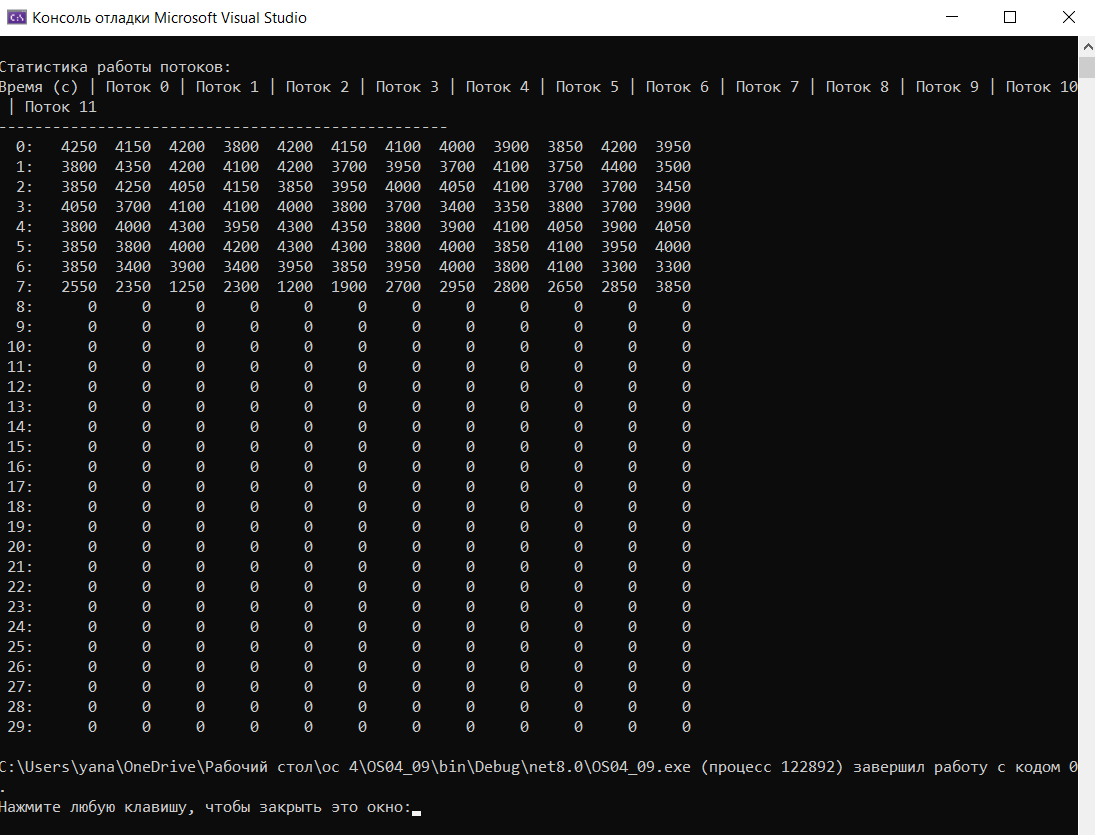
20. Скопируйте консольное приложение OS04\_07 как OS04\_08. Теперь используйте пул потоков. Выведите статистику работы потоков на экран в виде таблицы.



**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.

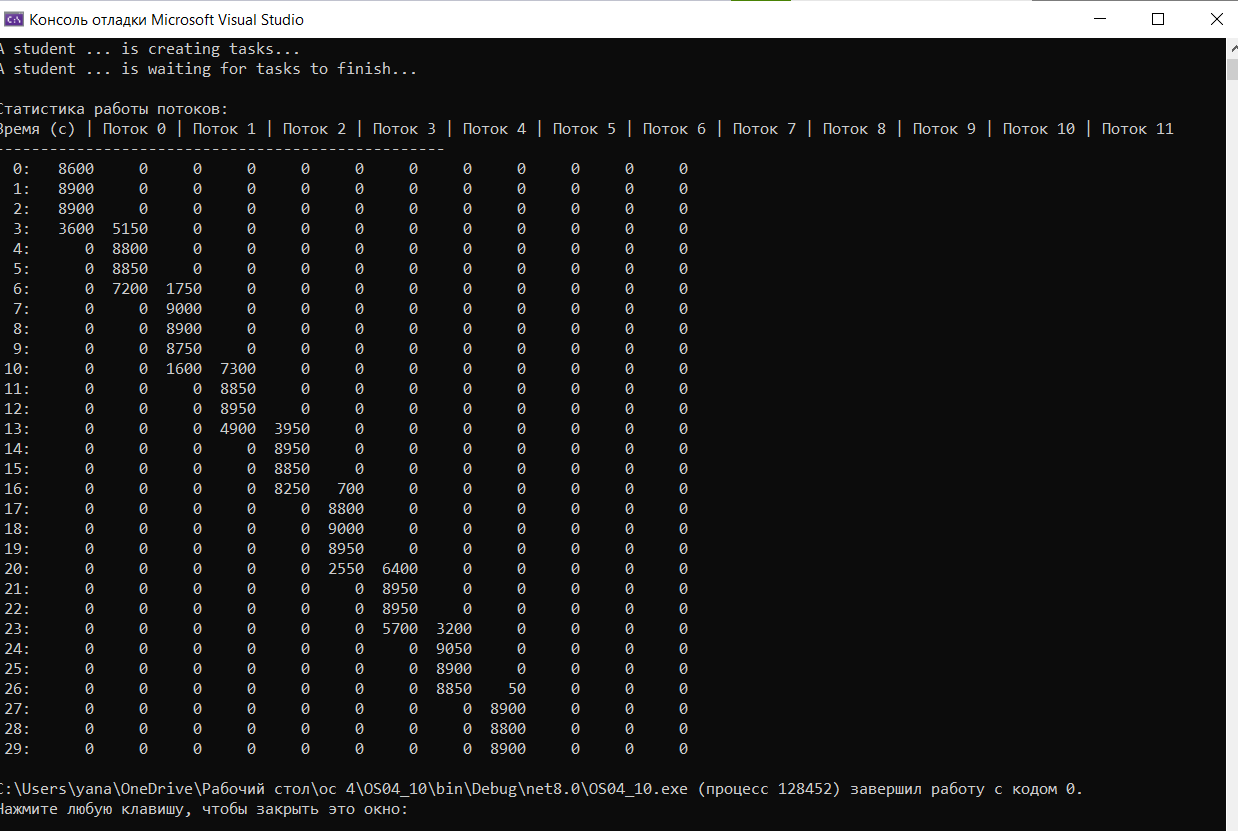
**7:** большое количество накладных расходов на создание и уничтожение потоков, особенно если потокам нужно часто запускаться и завершаться.

**8:** В коде с пулом потоков потоки создаются и управляются автоматически. Мы просто ставим задачи в очередь, и пул сам решает, как и когда их выполнять.

**9:** мы явно инициализируем массив задач и запускаем каждую задачу отдельно. Это может создать дополнительные накладные расходы на управление задачами, особенно если в системе много короткоживущих задач.

**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).

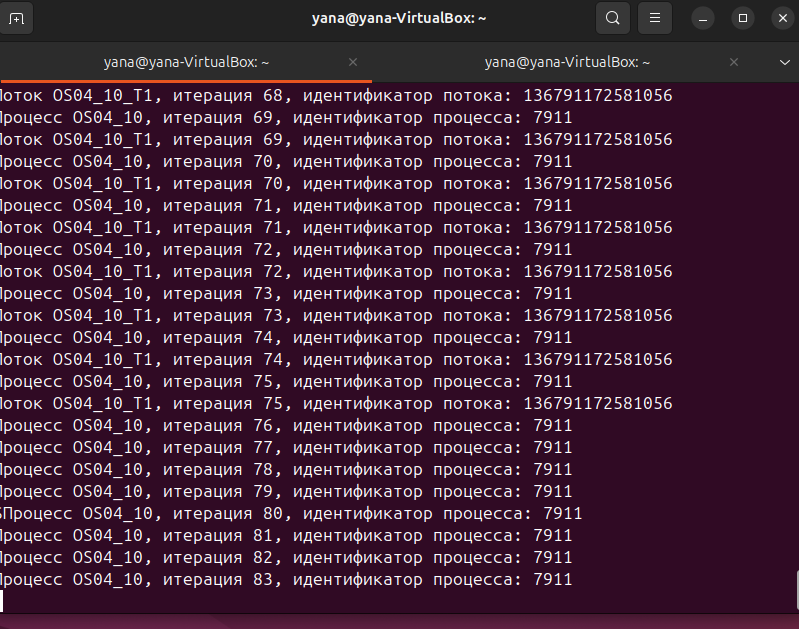


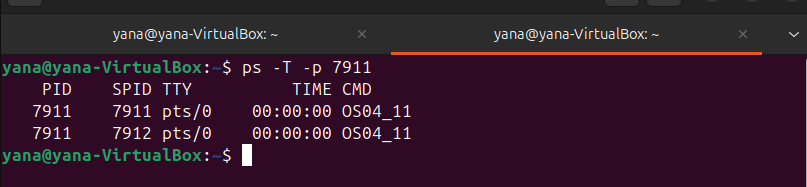


25. Процесс OS04\_10 должен создать поток: потоковая функция OS04\_10\_T1.

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.

-T: Показывает информацию о потоках для указанного процесса.

-p \_\_\_\_: Указывает PID процесса, для которого нужно отобразить информацию о потоках.

**Задание 12.**

28. Что такое поток управления OS?

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

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

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?

Свойство программного кода (программы) корректно работать в нескольких потоках одновременно. Гарантирует, что при исполнении нескольких потоков код будет правильно себя вести.

Имеет **два** основных принципа *(из лекции Смелова)*:

1. Код не должен сам себя менять

2. Не должно быть статической области памяти (общей для двух потоков)

36. Что такое реентерабельность кода?

**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.