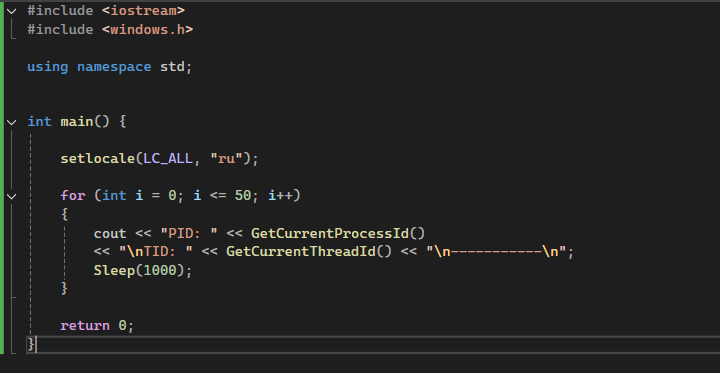
ОТЧЁТ

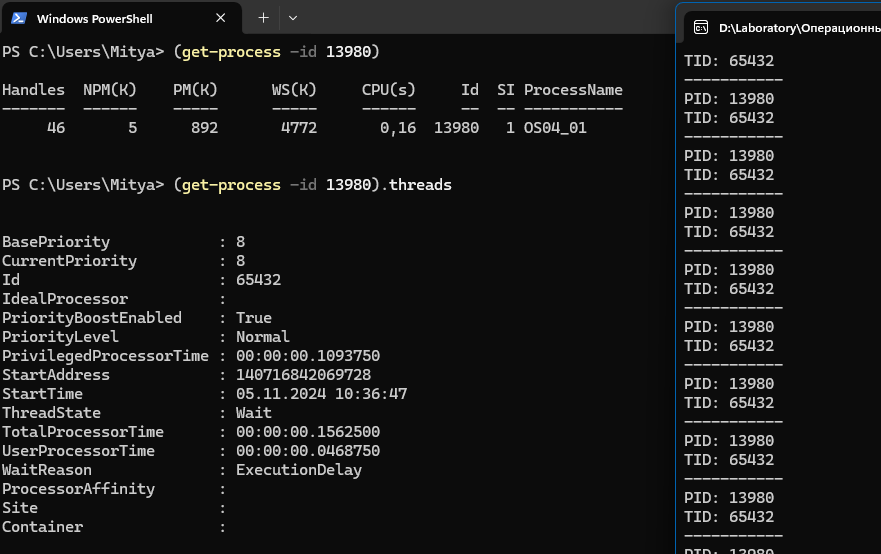
Операционные Системы

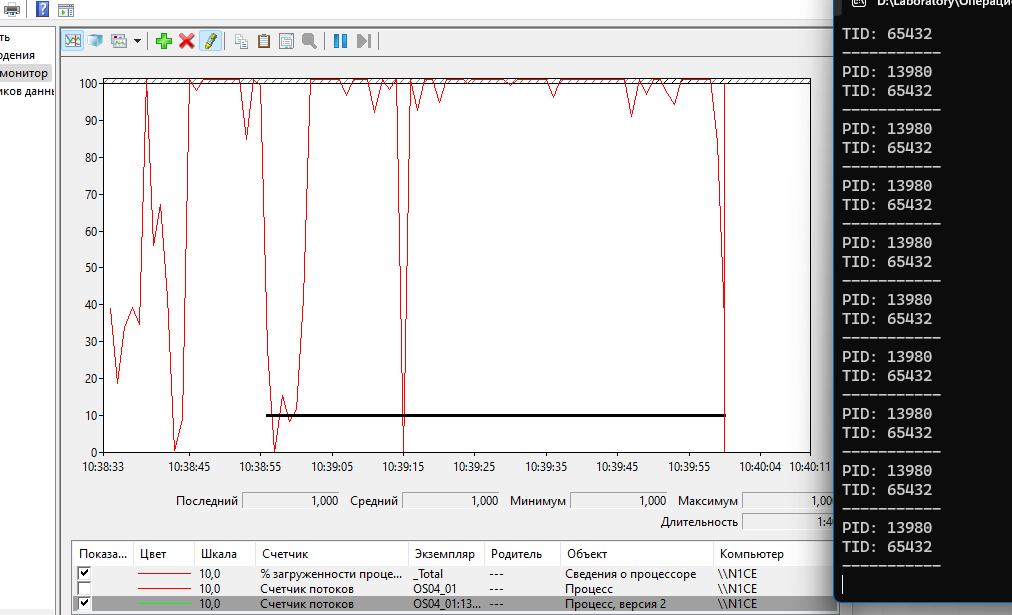
Беласин Д. А.

Лабораторная №4

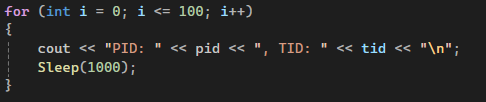
**Задание 01**

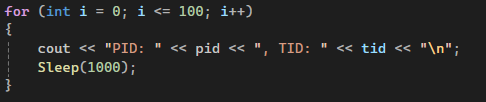
1. Разработайте консольное Windows-приложение OS04\_01 на языке С++, выполняющее длинный цикл с временной задержкой и с выводом на консоль идентификаторов текущего процесса и текущего потока. 

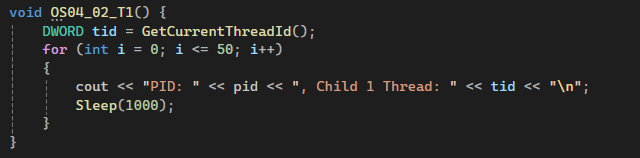
2. Продемонстрируйте информацию об потоках процесса OS04\_01 с помощью утилит PowerShell ISE и Performance Monitor. 

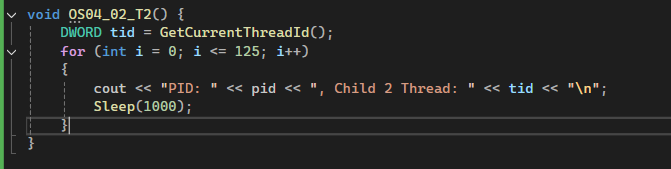


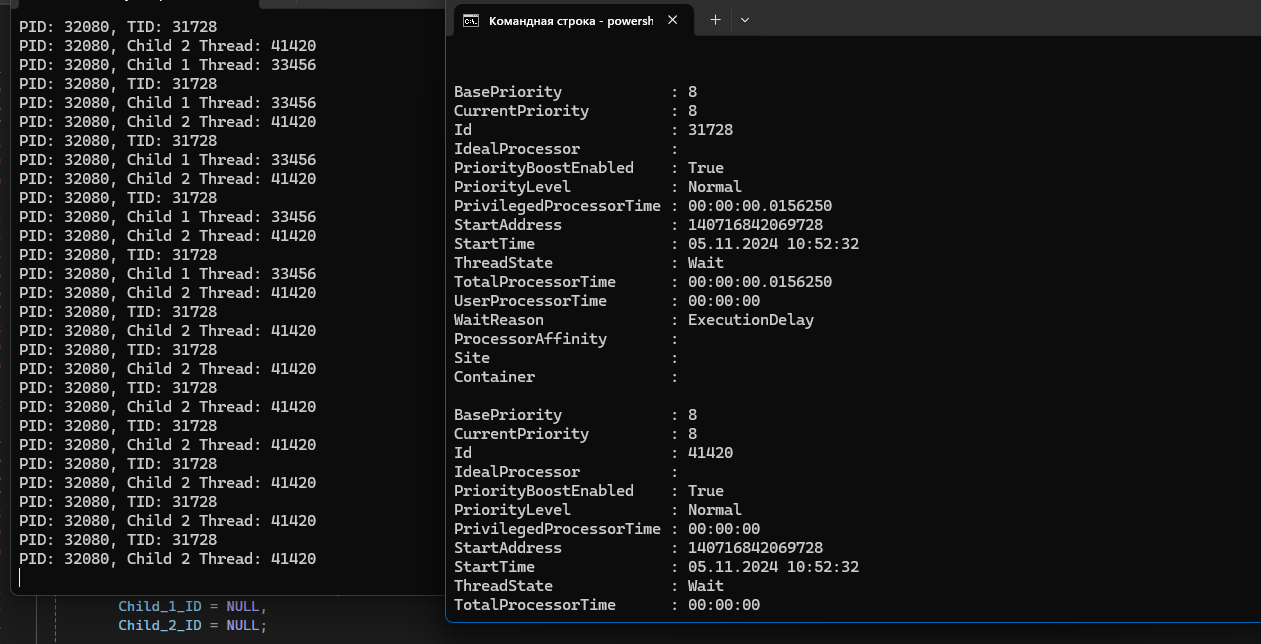
**Задание 02**

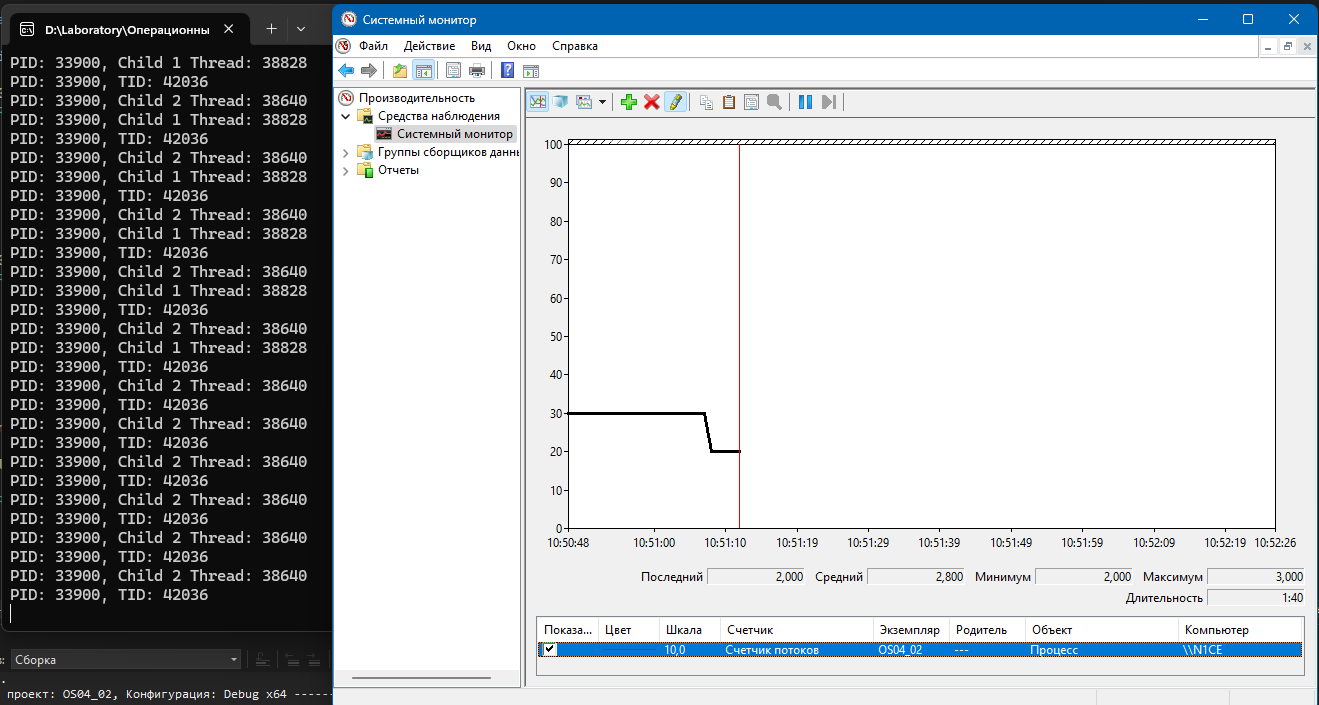
3. Разработайте консольное Windows-приложение OS04\_02 на языке С++, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса. 

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

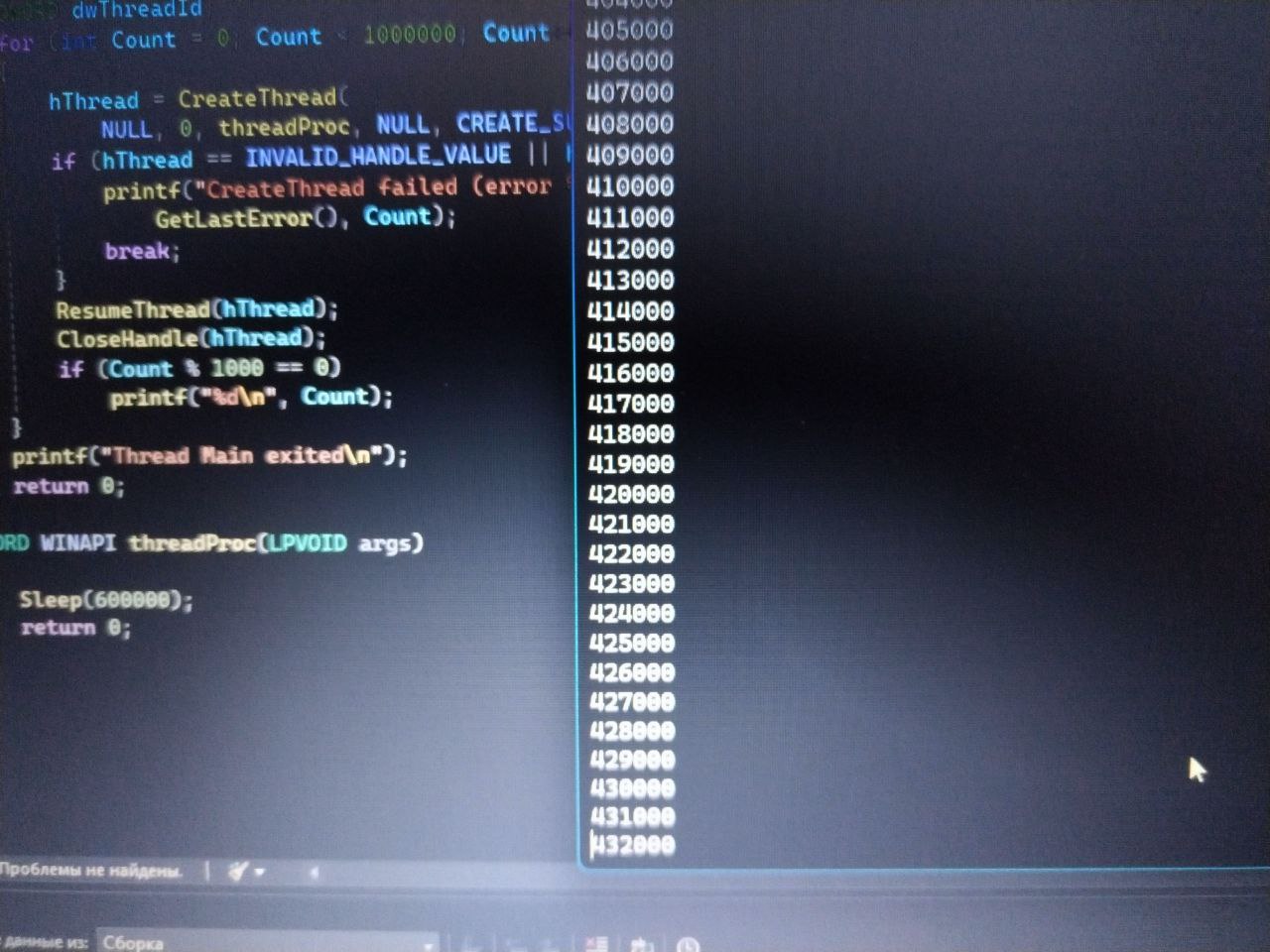
5. Поток OS04\_02\_T1 - выполняет цикл 50 итераций с временной задержкой в 1 сек. с выводом на консоль идентификаторов процесса и потока. 

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

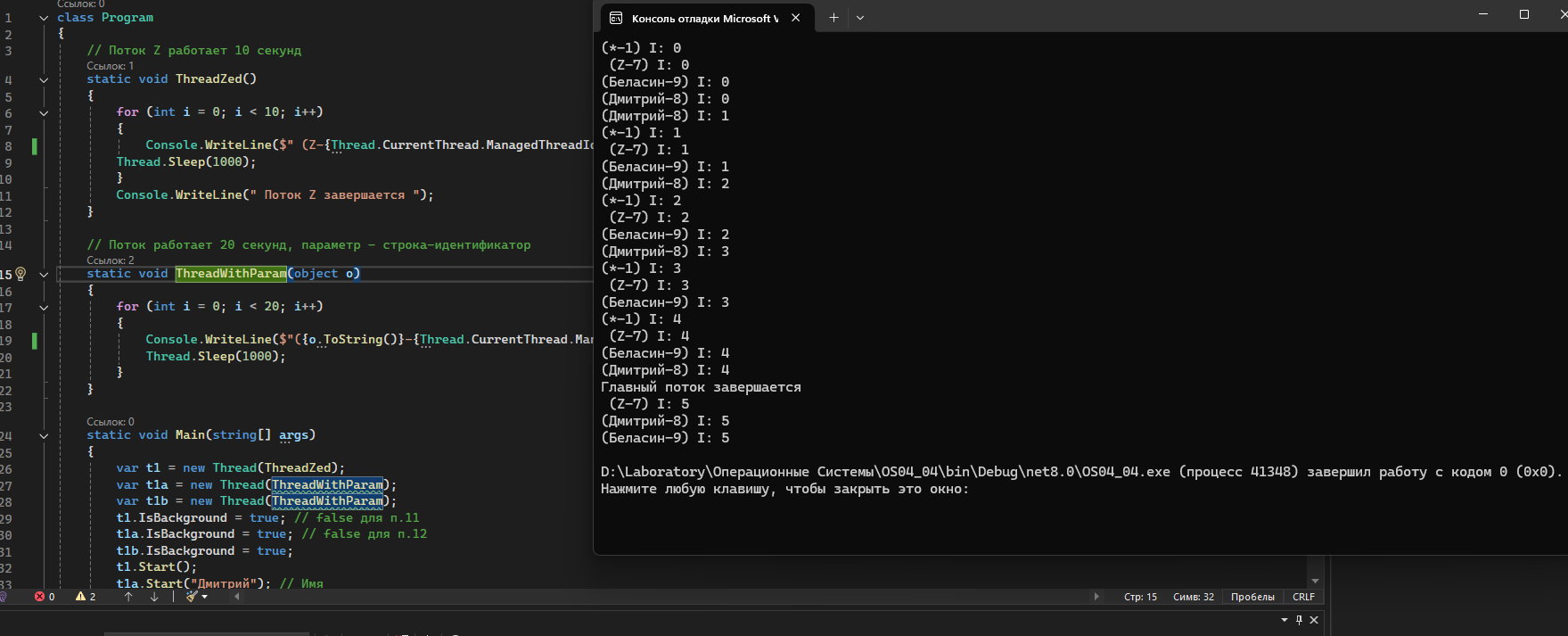
7. Продемонстрируйте информацию об потоках процесса OS04\_02 с помощью утилит PowerShell ISE и Performance Monitor. 



**Задание 03.**

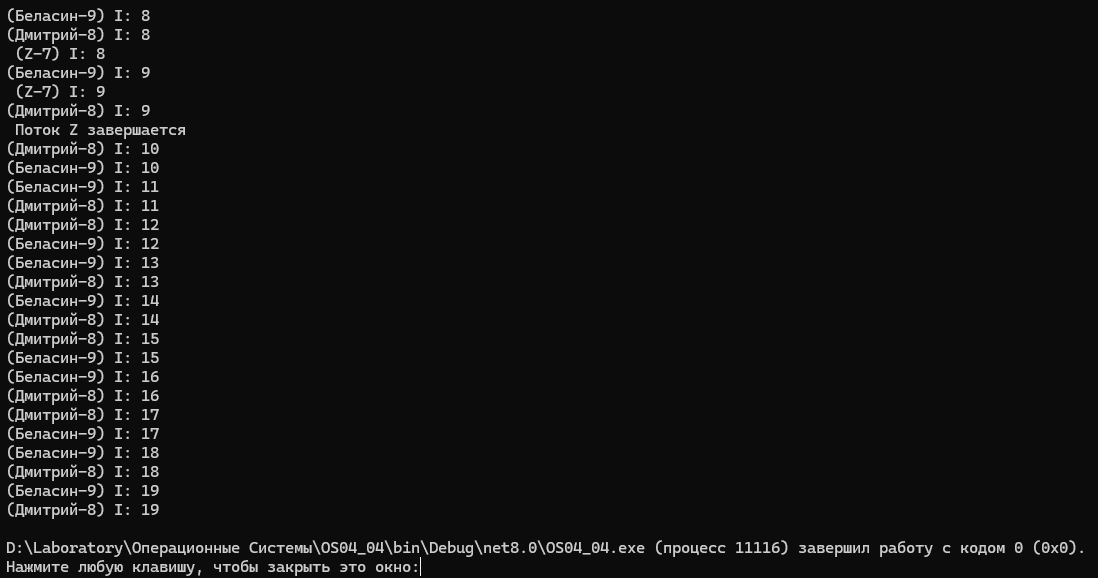
8. Определите, какое максимальное количество потоков можно создать в одном процессе на вашем компьютере, используя нижеприведенный программный код (или аналогичный). 

**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. Фрагмент программного кода приведен ниже. (Вставьте Свои Имя Фамилию). 

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

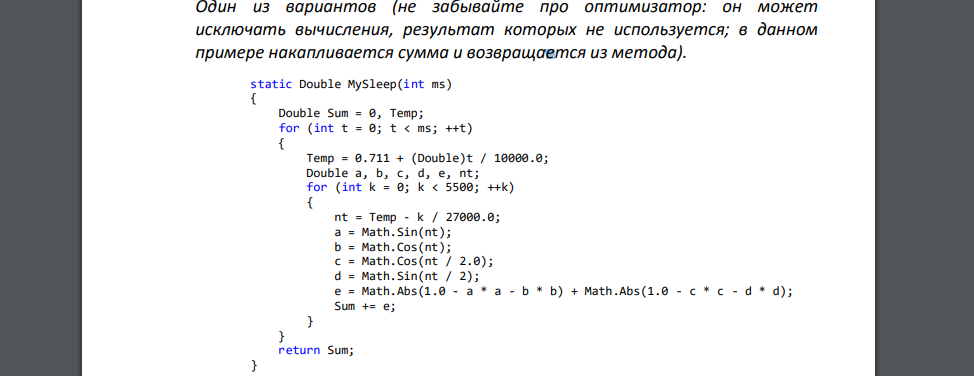
11. Измените значение свойства IsBackground для первого дополнительного потока на false и снова выполните приложение. Обратите внимание на общее время работы приложения. 

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.

Работа приложения завершается после завершения всех нефоновых процессов

**Задание 05 (вспомогательное)**

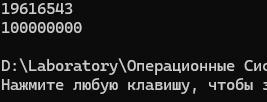
15. Создайте функцию, которая производит ЛЮБЫЕ вычисления длительностью n миллисекунд на вашем компьютере (для последующих заданий метод Thread.Sleep(n) не подходит, так как он освобождает центральный процессор и ничего не делает). Убедитесь, что MySleep(10000) работает ровно 10 секунд. Один из вариантов (не забывайте про оптимизатор: он может исключать вычисления, результат которых не используется; в данном примере накапливается сумма и возвращается из метода). 

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

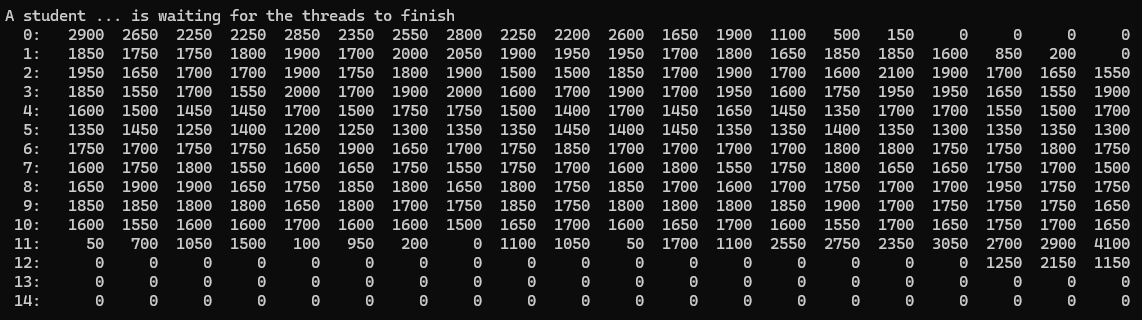
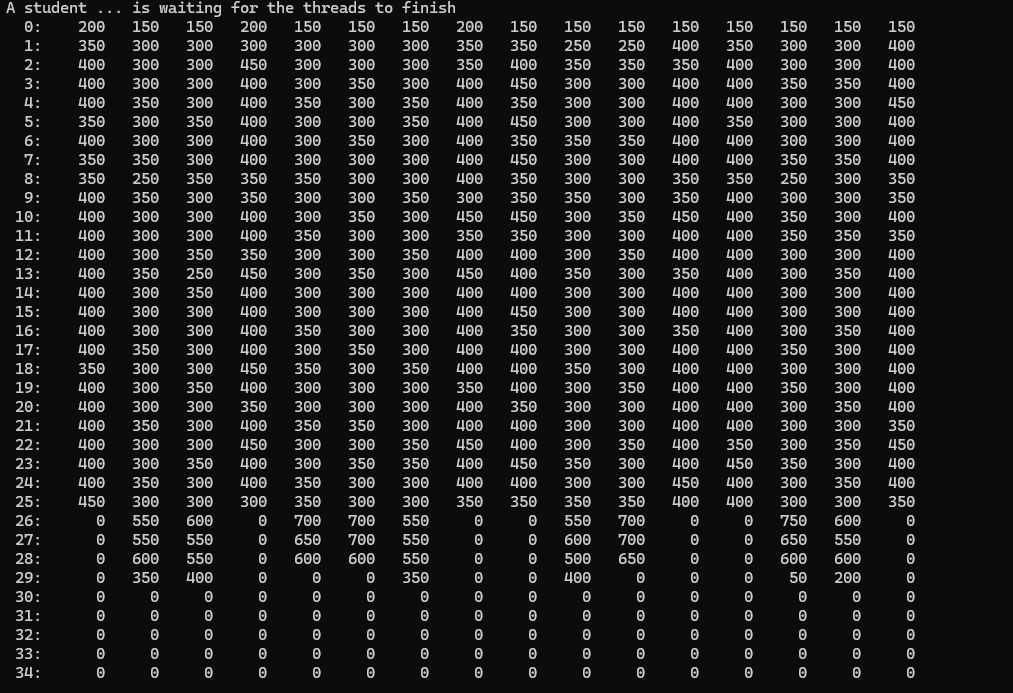
**Задание 06**

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.

18. Сравните результаты в группе. Есть ли какая закономерность? Примечание. В лабораторной работе №6 это приложение нужно будет исправить.



**Задание 07**

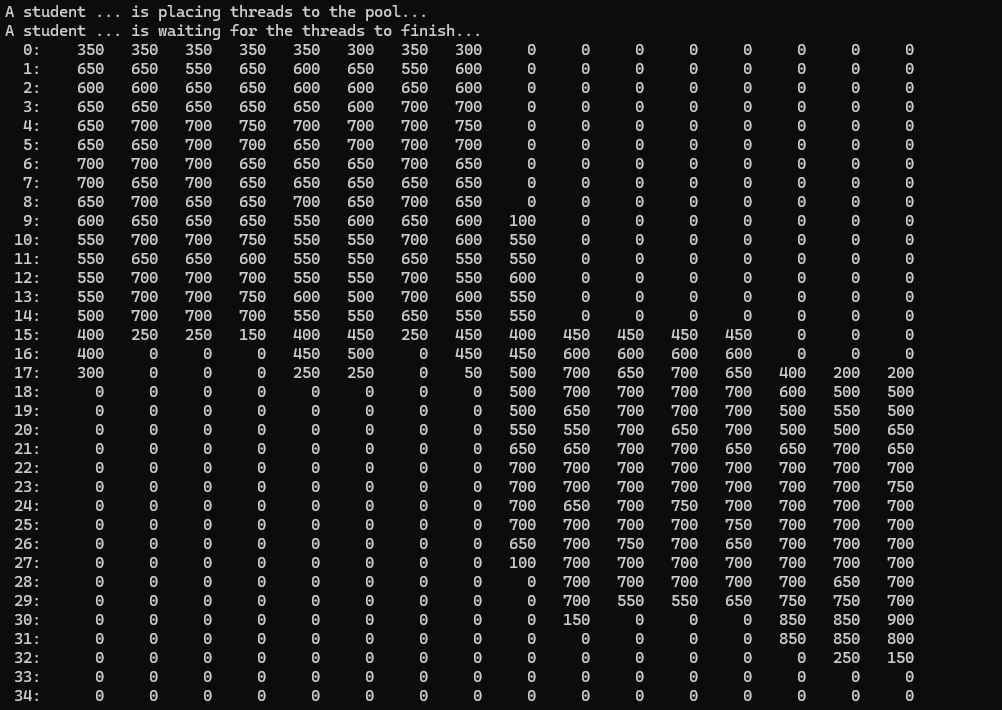
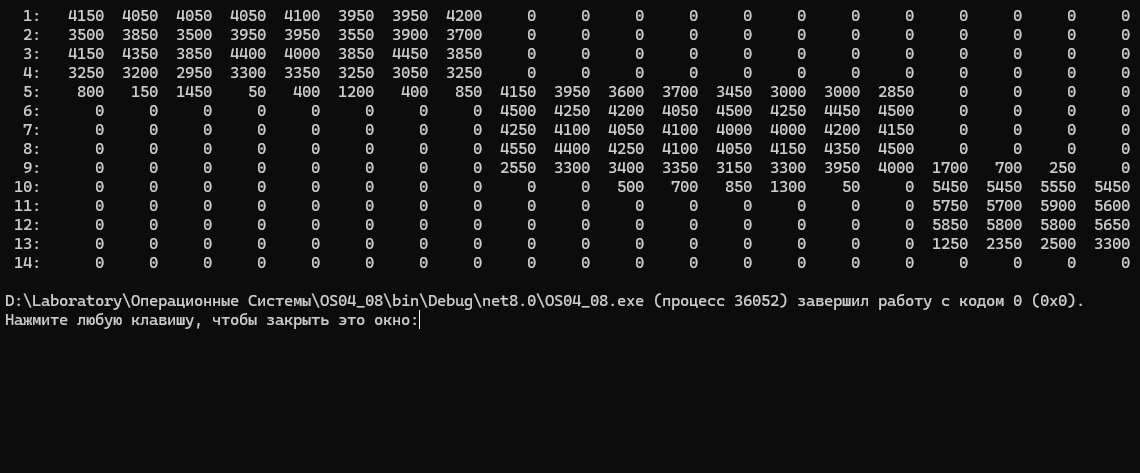
19. Разработайте консольное приложение OS04\_07 на языке С#, запускающее N потоков, каждый из которых будет производить вычисления t секунд (использовать разработанный в задании 5 метод), используя класс System.Threading.Thread. Сохраните информацию о работе потоков в течение T секунд и выведите на экран 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).  

const int ThreadCount = 16;

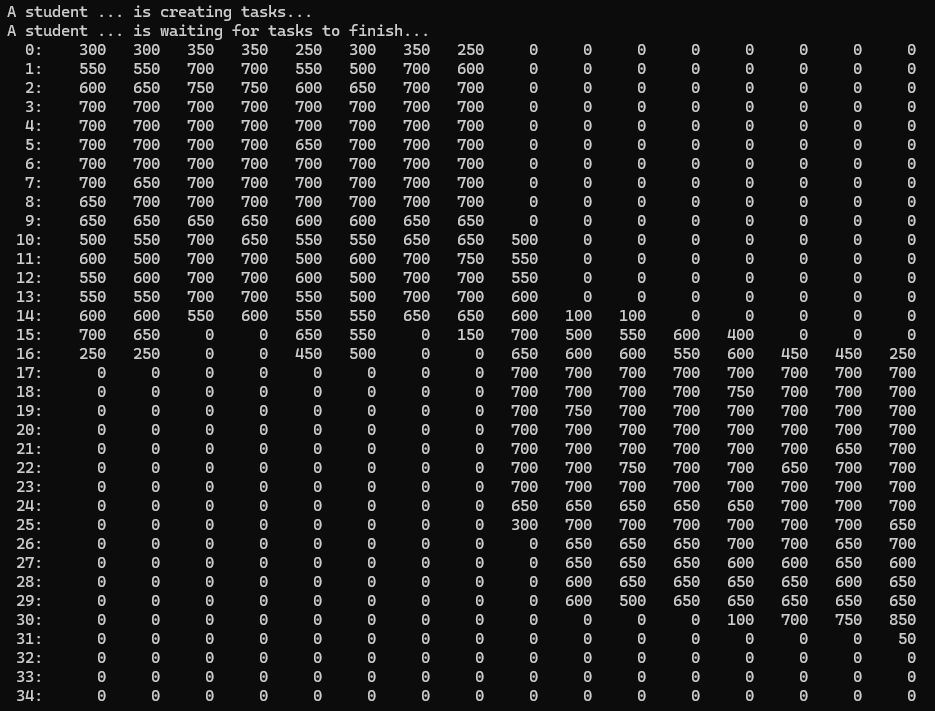
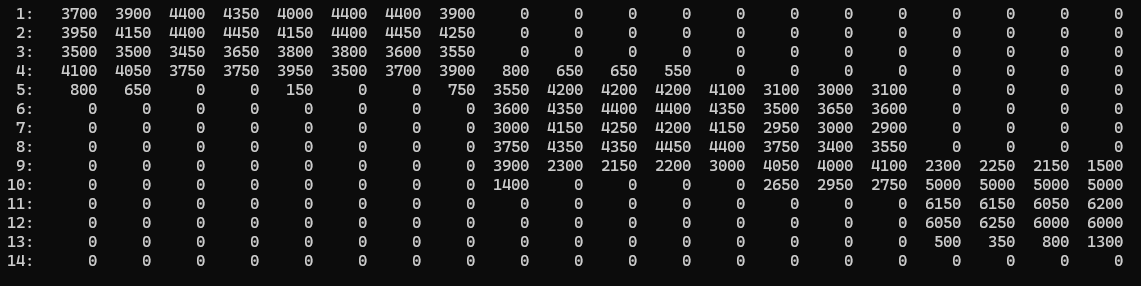
const int ThreadLifeTime = 10;

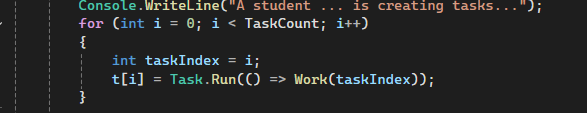
const int ObservationTime = 35;

**Задание 08**

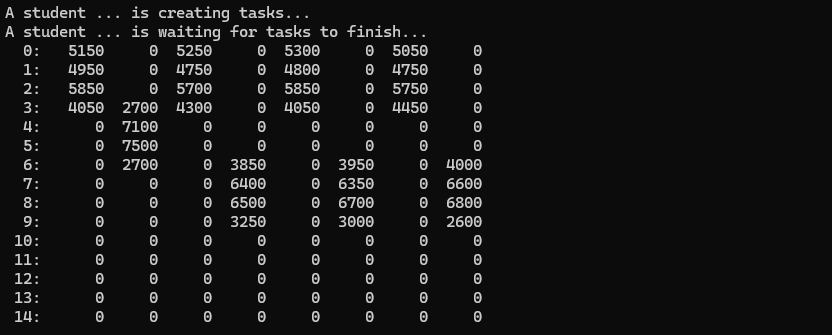
20. Скопируйте консольное приложение OS04\_07 как OS04\_08. Теперь используйте пул потоков. Выведите статистику работы потоков на экран в виде таблицы. 

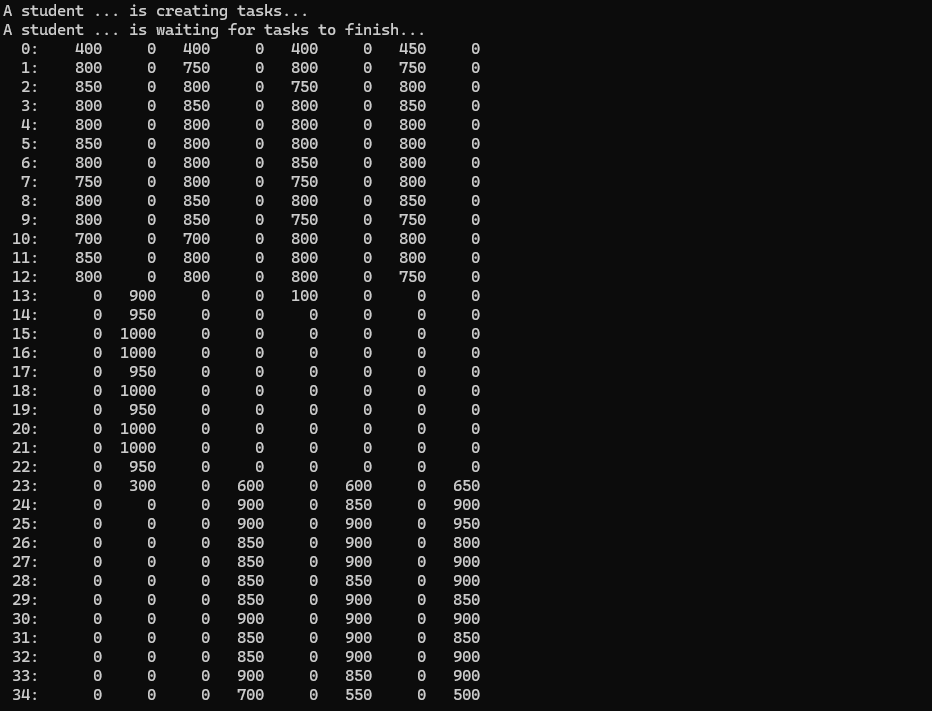
**Задание 09**

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. 

22. [For nerds only] Реализуйте запуск задач в цикле. 

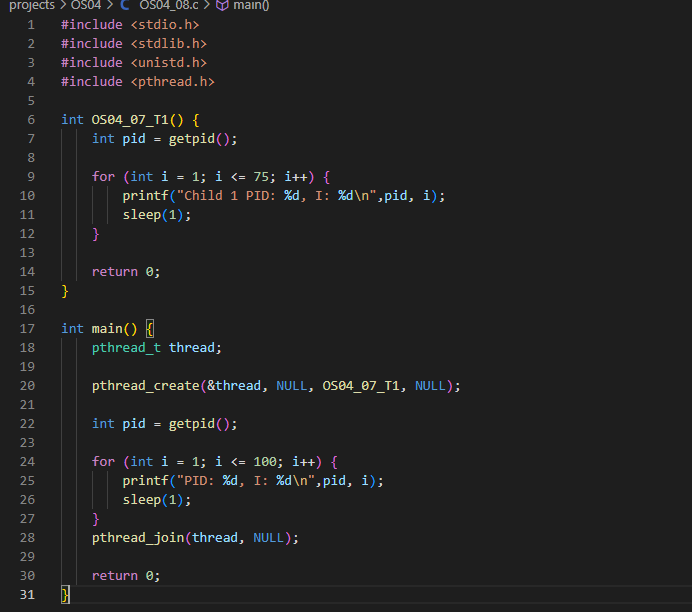
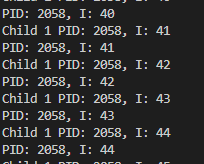
**Задание 10**

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. 

**Задание 12. Ответьте на следующие вопросы**

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

Поток управления — это механизм, который определяет порядок выполнения задач в операционной системе (ОС). Он управляет переключением между задачами, обеспечивая эффективное использование ресурсов.

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

* **Windows**: В Windows потоки создаются с помощью функции CreateThread.
* **Linux**: В Linux потоки создаются с помощью системного вызова pthread\_create из библиотеки POSIX threads (pthreads).

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

* **Системные потоки**: управляются операционной системой. Они могут быть более эффективными, так как ОС управляет их планированием.
* **Пользовательские потоки**: управляются приложением, а не ОС. Они требуют больше времени на переключение, но могут быть полезны в специфичных задачах.

**31. Что такое многопоточность?**

Многопоточность — это способность программы выполнять несколько потоков одновременно. Это позволяет улучшить производительность и отзывчивость приложения.

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

Контекст потока — это информация, необходимая для выполнения потока, включая его состояние, регистры процессора и стек. Он нужен для того, чтобы ОС могла правильно переключать потоки, сохраняя их состояние.

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

* **Готов (Ready)**: Поток готов к выполнению, но ждет своей очереди.
* **Запущен (Running)**: Поток активно выполняется.
* **Ожидание (Waiting)**: Поток ожидает завершения другого потока или ресурса.
* **Завершен (Terminated)**: Поток завершил выполнение и не может быть запущен снова.

**34. Что такое LWP?**

LWP (Light Weight Process) — это легкий процесс, который может иметь собственный стек и контекст, но разделяет ресурсы с другими потоками в пределах одного процесса. Это облегчает переключение между потоками.

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

Потокобезопасность — это свойство кода, которое гарантирует правильное функционирование при одновременном доступе из нескольких потоков. Это предотвращает ошибки и некорректное поведение программы.

**36. Что такое реентерабельность кода?**

Реентерабельность — это способность быть вызванной несколькими потоками одновременно без нарушений, при этом не полагаясь на глобальные или статические данные. Это обеспечивает безопасность при параллельном выполнении.

**37. Что такое Fiber?**

Fiber — это легковесный поток, который управляется приложением, а не операционной системой. Они позволяют выполнять несколько потоков в одном потоке ОС, что может быть полезно для управления большим количеством задач с минимальными накладными расходами.