**Филиал федерального государственного бюджетного образовательного учреждения высшего образования**

**«Национальный исследовательский университет «МЭИ»**

**в г. Смоленске**

Кафедра вычислительной техники

Направление: 09.04.01. «Информатика и вычислительная техника»

Профиль: «Программное обеспечение средств вычислительной техники и

автоматизированных систем»

Практическая работа №2

«Отладка простых программ на вычислительном кластере с использованием OpenMP»

по курсу:

«Вычислительные системы»

Студент: Старостенков А.А.

Группа: ВМ-22(маг)

Вариант: 19

Преподаватель: Федулов А.С.

Смоленск, 2023

**Задание**

1. Отладить, скомпилировать и запустить на ГВК простые тестовые примеры на языке Си с использованием технологии OpenMP. В качестве базовых примеров использовать примеры из лекции № 3 по технологии OpenMP по курсу «Вычислительные системы». При выводе результатов для каждой программы предусмотреть вывод своей фамилии и инициалов (в латинице).
   1. Определить версию OpenMP (пример 1).
   2. Организовать задержку выполнения программы на заданный интервал времени и измерить этот интервал с помощью функций OpenMP (пример 2). Интервал задержки в секундах выбрать равным номеру по журналу. Отлаженную программу запустить несколько раз (2-3). Убедиться в том, что результат отличается от запуска к запуску. Объяснить причину этого.
   3. Определить максимально возможное число нитей (потоков), заданное по умолчанию (пример 3). Изменить это число с помощью команды **export OMP\_NUM\_THREADS=n.** В качестве числа потоков n в параллельной области использовать значение «номер по журналу»+4. Вновь запустить пример 3. Оценить результаты.
   4. В примере 4 проверить определение числа нитей в параллельной области тремя способами: с помощью переменной окружения (присвоить значение «номер по журналу»+2), с помощью функции OpenMP (присвоить значение «номер по журналу»+3), c помощью опции директивы pragma omp parallel (присвоить значение «номер по журналу»+4). Оценить результаты выполнения примера 4.
   5. Пример 5 выполнить с числом потоков в параллельной области, равным номеру по журналу+8. Число потоков в параллельной области задать любым способом из рассмотренных в примере 4.
   6. Пример 6 выполнить с числом потоков в параллельной области, равным номеру по журналу+9.
   7. Программу из примера 7 запустить несколько раз. Убедиться в наличии гонок по данным.
   8. Примеры 9 выполнить с числом потоков в параллельной области, равным номеру по журналу+7. Выполнить программу с опцией reduction и без нее. Объяснить полученные результаты.
   9. Пример 9.1 выполнить без модификаций.
   10. Пример 10 выполнить без модификаций. Определить число итераций цикла, реализованных каждой нитью.

**Ход работы**

1.1. Определить версию OpenMP (пример 1).

Код:



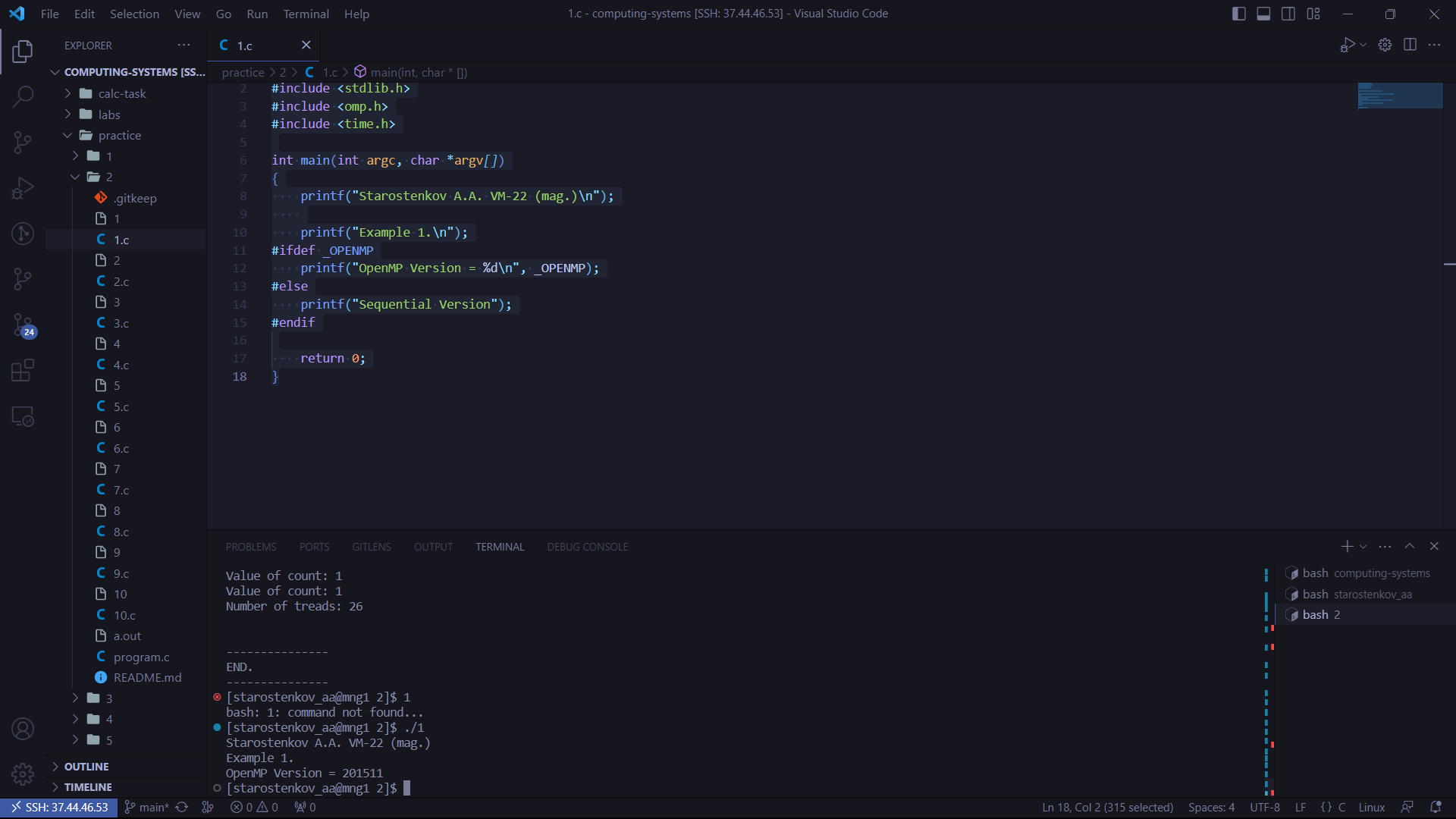
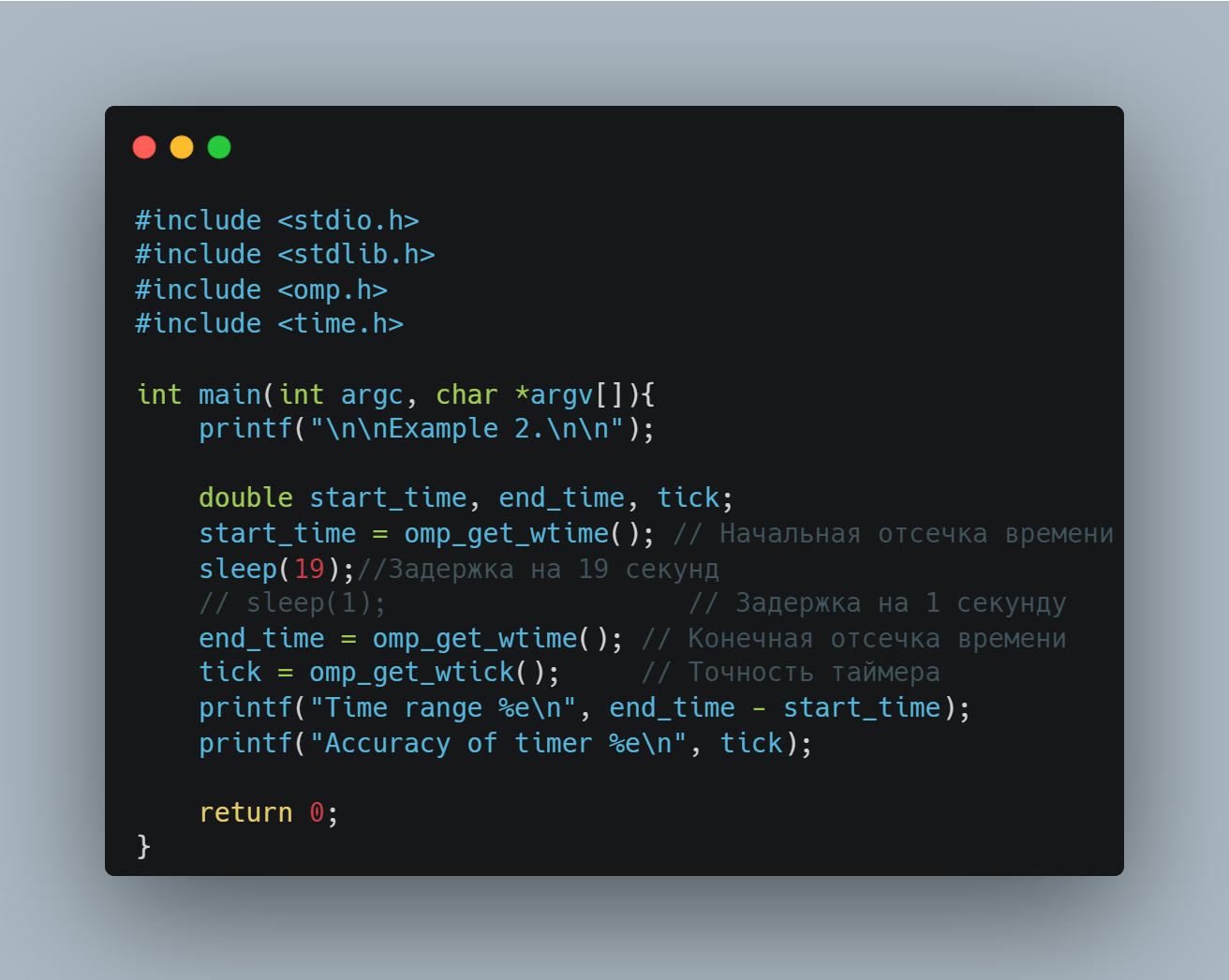


Рисунок 1 – Запрос версии

Успешно, на сервере установлен OpenMP

1.2. Организовать задержку выполнения программы на заданный интервал времени и измерить этот интервал с помощью функций OpenMP (пример 2). Интервал задержки в секундах выбрать равным номеру по журналу. Отлаженную программу запустить несколько раз (2-3). Убедиться в том, что результат отличается от запуска к запуску. Объяснить причину этого.

Код:

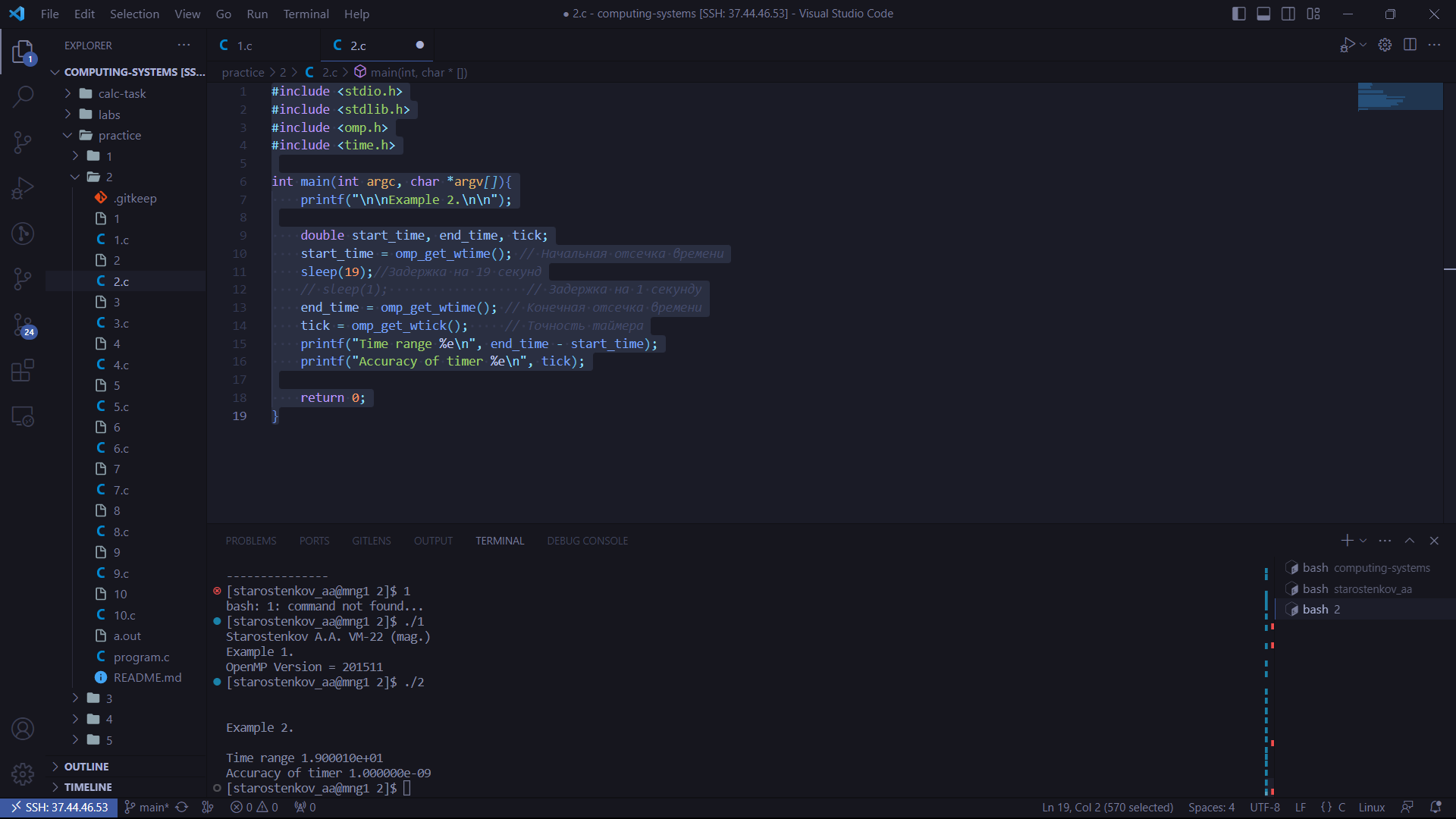


Рисунок 2 – Задержки

Вывод: измеренный временной интервал слегка отличается от запуска к запуску. Это может происходить по целому ряду причин:

1. Linux является многопользовательской системой с вытесняющей многозадачностью – часть процессорного времени уходит на другие задачи.

2. Различное время доступа к памяти, в зависимости от загрузки шин и состояния кэшей процессора

3. Состояние конвейера процессора – время выполнения одних и тех же машинных команд может быть различным.

1.3. Определить максимально возможное число нитей (потоков), заданное по умолчанию (пример 3). Изменить это число с помощью команды export OMP\_NUM\_THREADS=n. В качестве числа потоков n в параллельной области использовать значение «номер по журналу»+4. Вновь запустить пример 3. Оценить результаты.

Код:



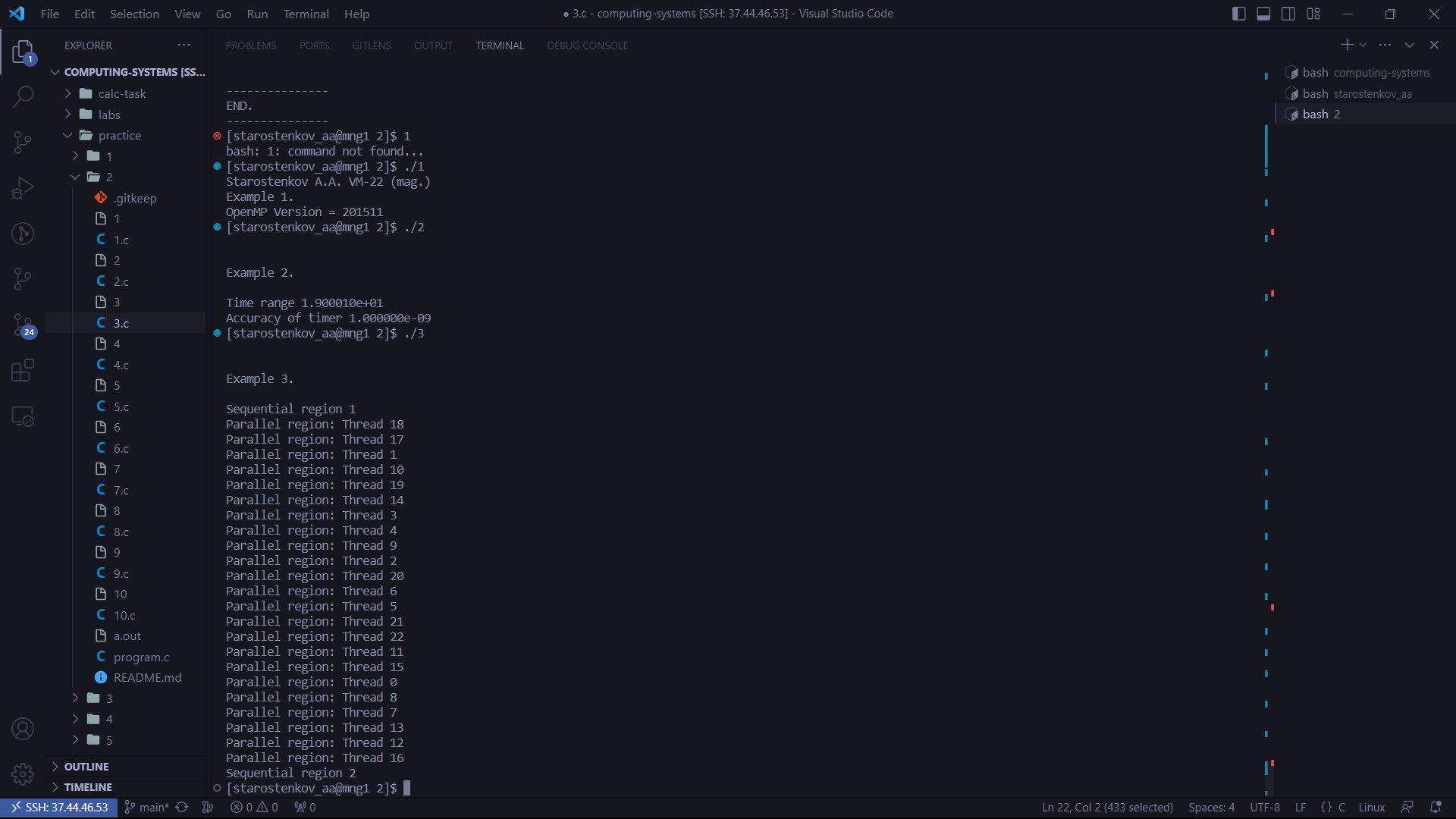


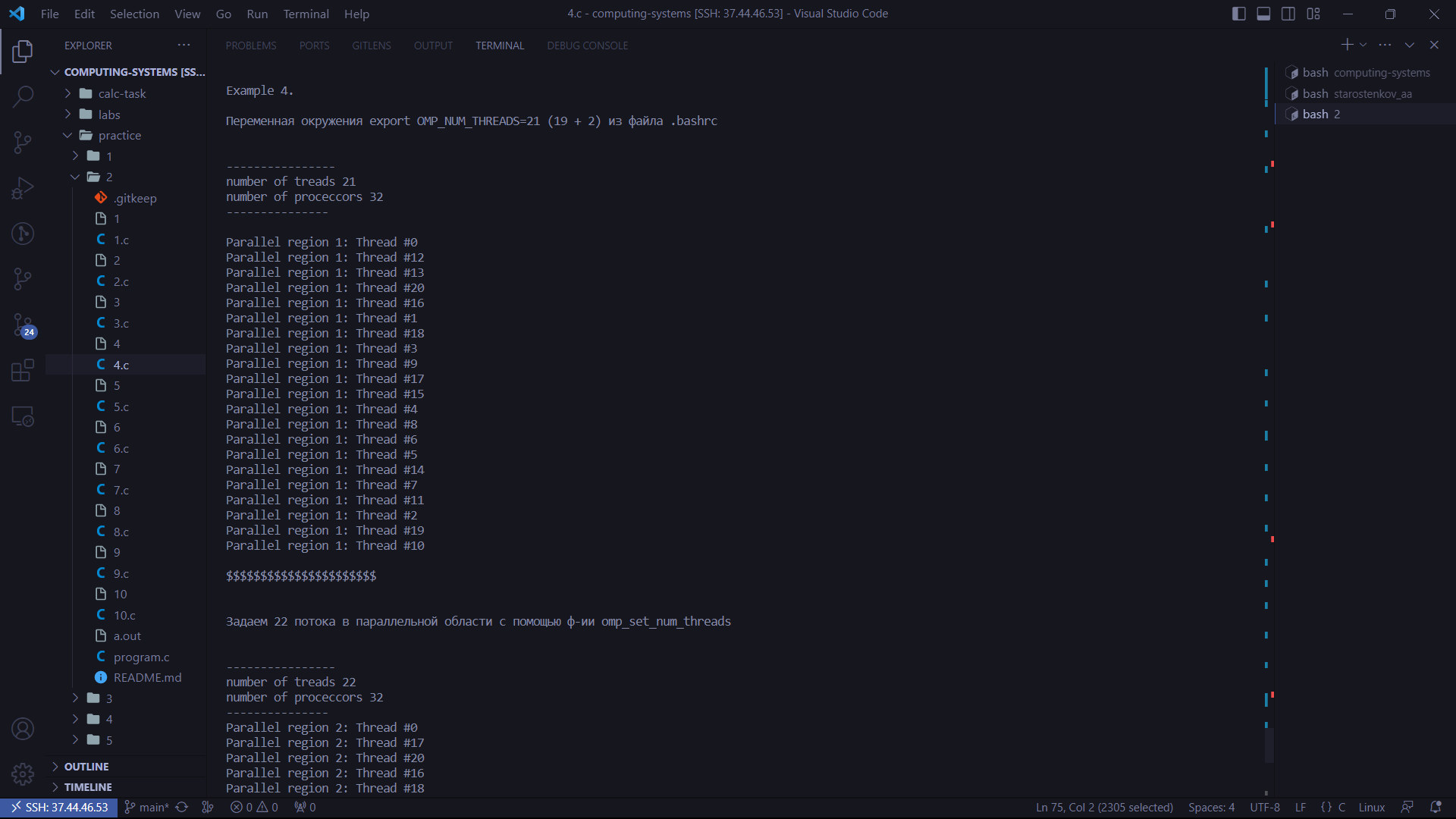
Рисунок 3 – Измененная программа

Ограничили количество потоков 23, что и видно на рисунке, вывелось 23 строки.

1.4. В примере 4 проверить определение числа нитей в параллельной области тремя способами: с помощью переменной окружения (присвоить значение «номер по журналу»+2), с помощью функции OpenMP (присвоить значение «номер по журналу»+3), c помощью опции директивы pragma omp parallel (присвоить значение «номер по журналу»+4). Оценить результаты выполнения примера 4.

Код





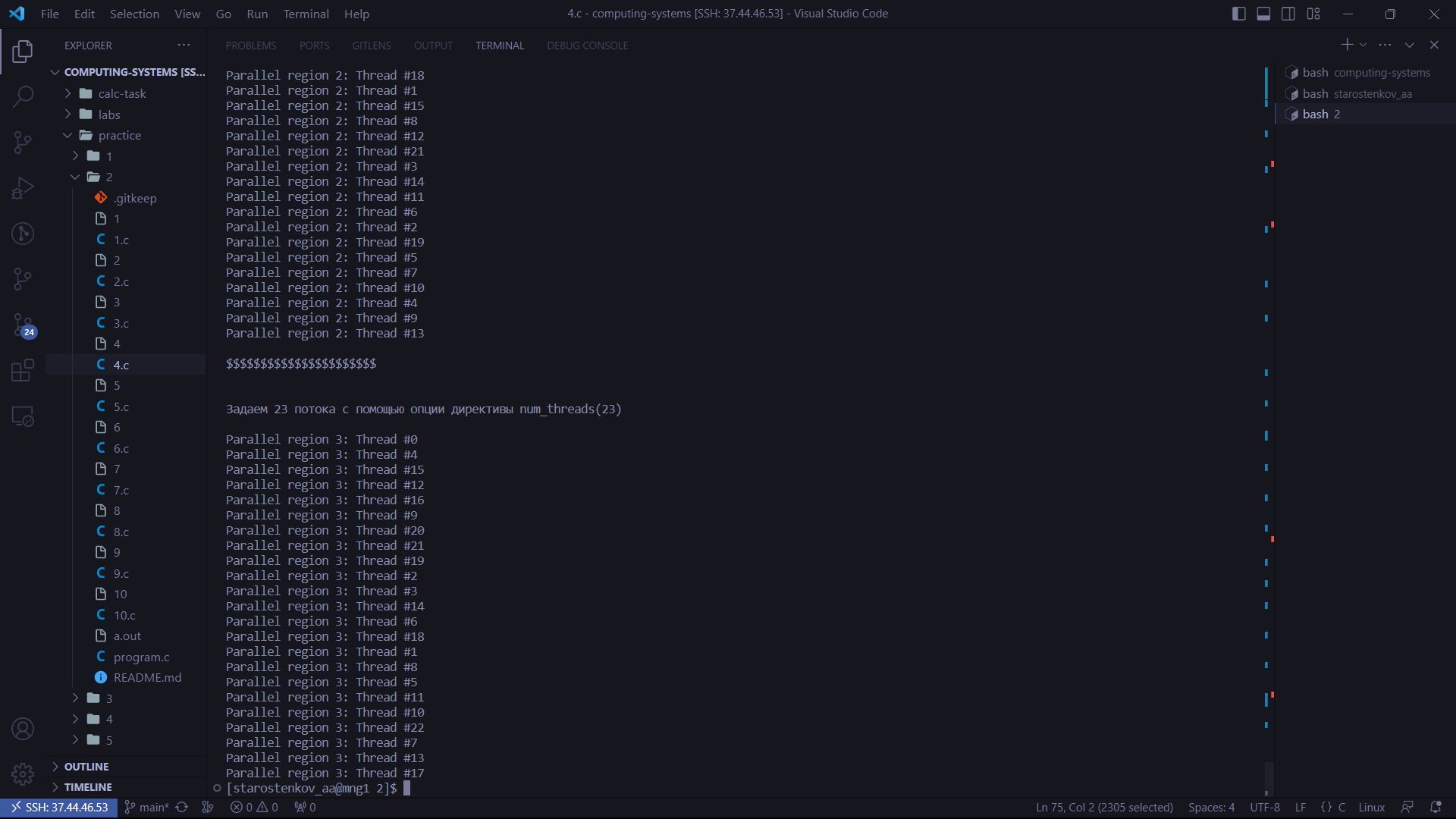
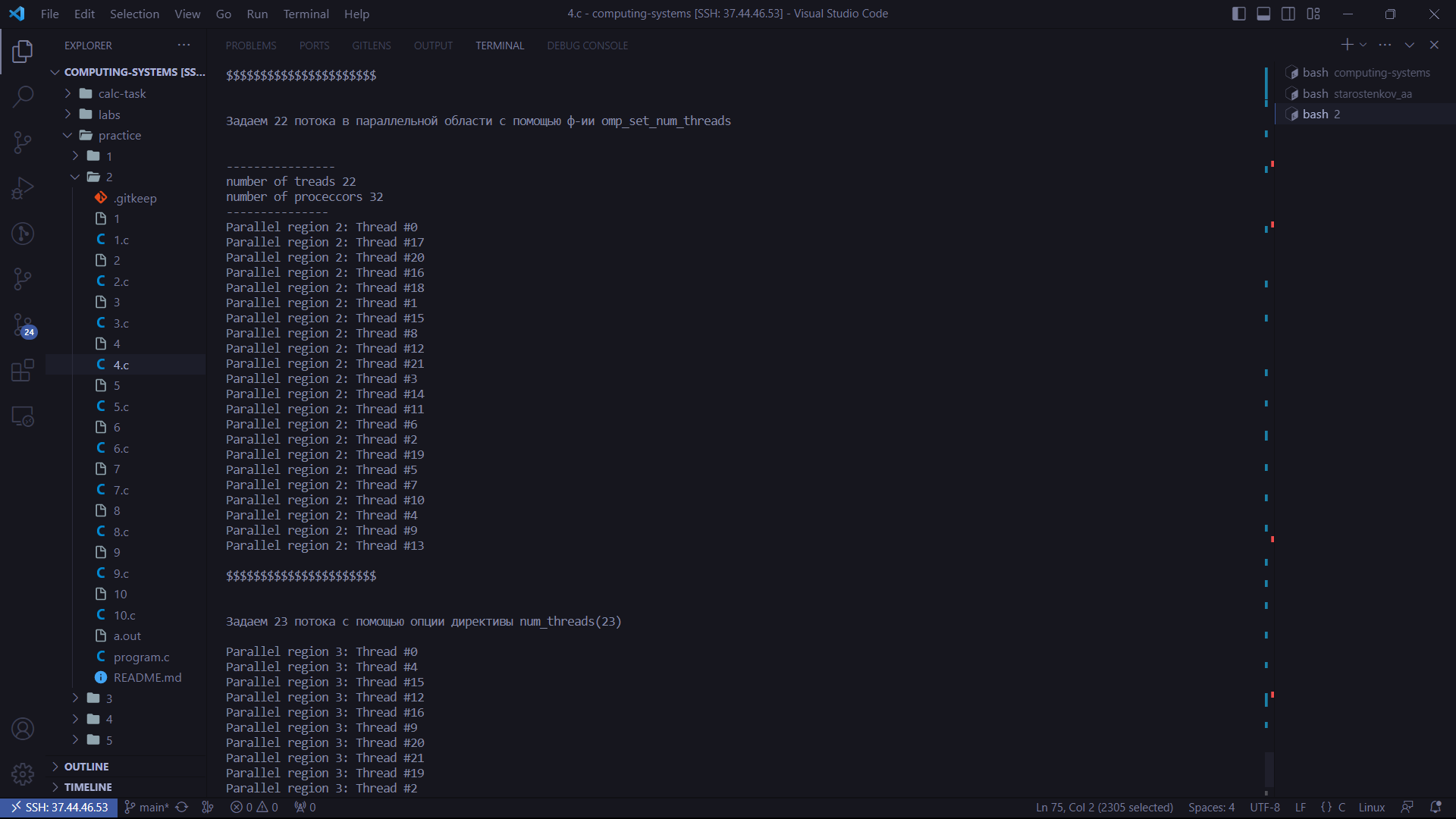
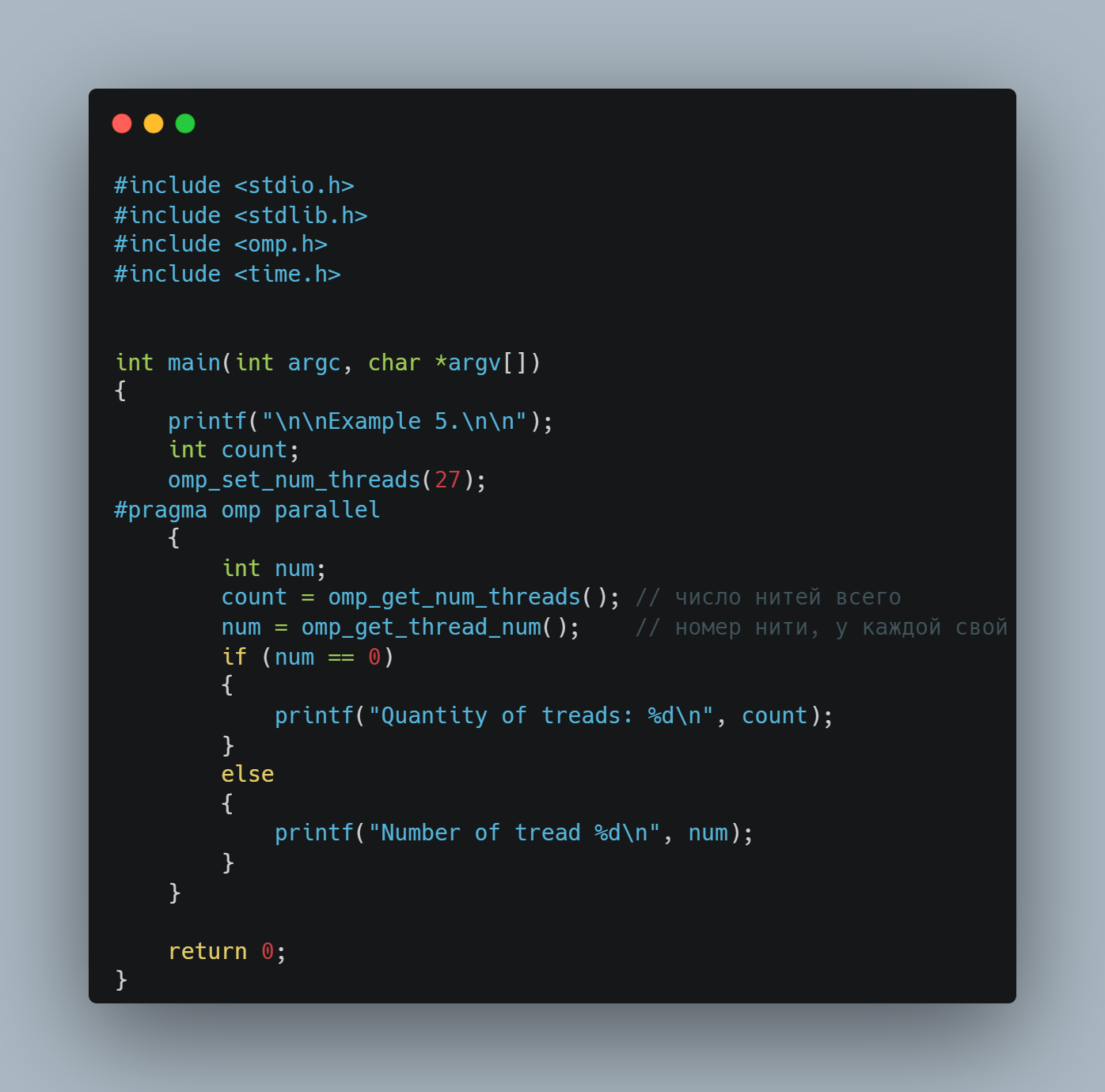


Рисунок 5 - Изменение потоков

Программа работает, число потоков изменяется тремя разными способами. Видно, что Опция num\_threads() имеет приоритет над функцией omp\_set\_num\_threads(), которая, в свою очередь, приоритетна над переменной окружения OMP\_NUM\_THREADS.

1.5. Пример 5 выполнить с числом потоков в параллельной области, равным номеру по журналу+8. Число потоков в параллельной области задать любым способом из рассмотренных в примере 4.

Код



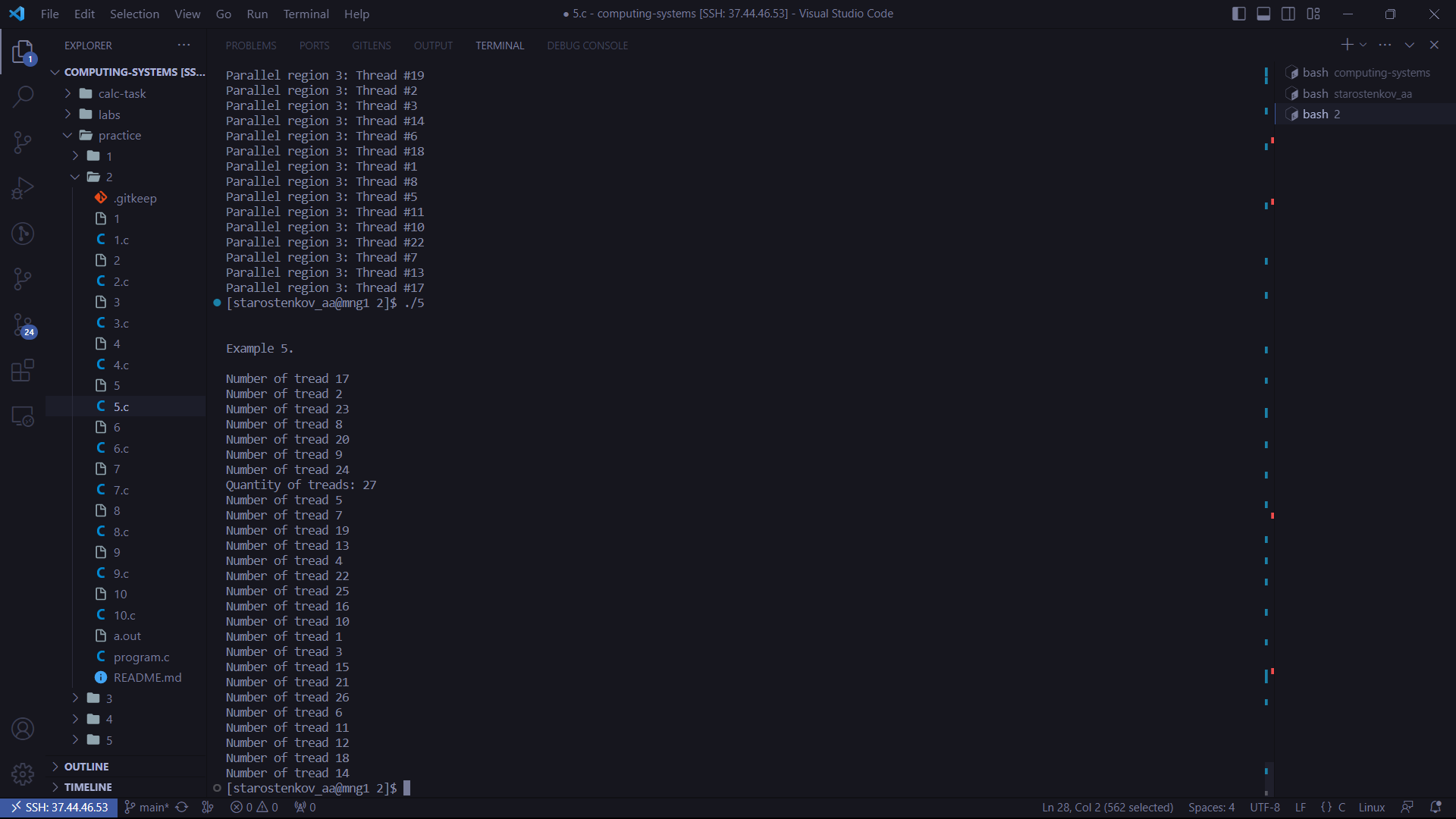


Рисунок 6 – Изменение потоков

Результаты различаются от запуска к запуску. Это происходит от того, что потоки не синхронизированы между собой и в качестве текущего берётся первый попавшийся свободный поток.

1.6. Пример 6 выполнить с числом потоков в параллельной области, равным номеру по журналу+9.

Код:



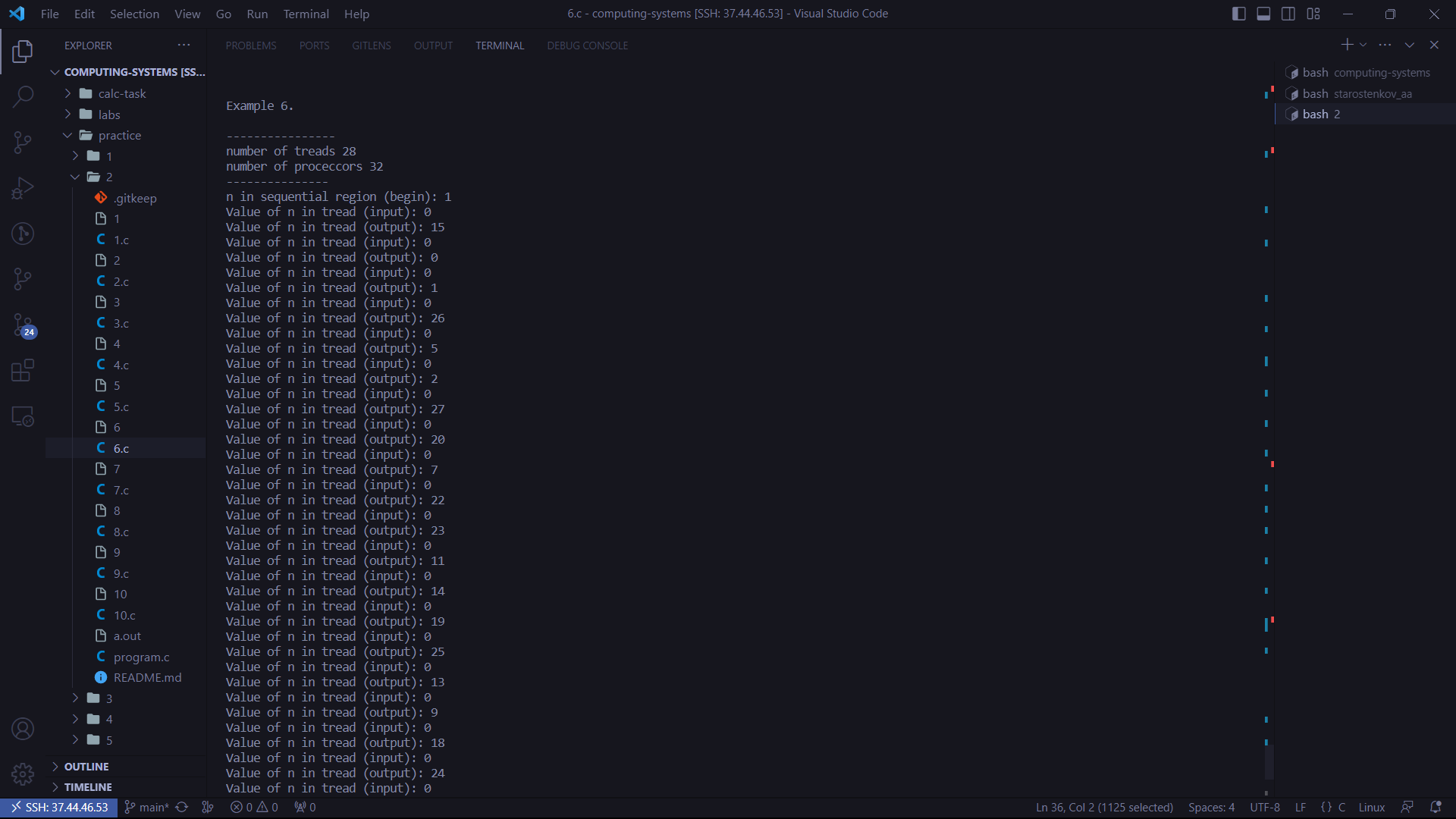
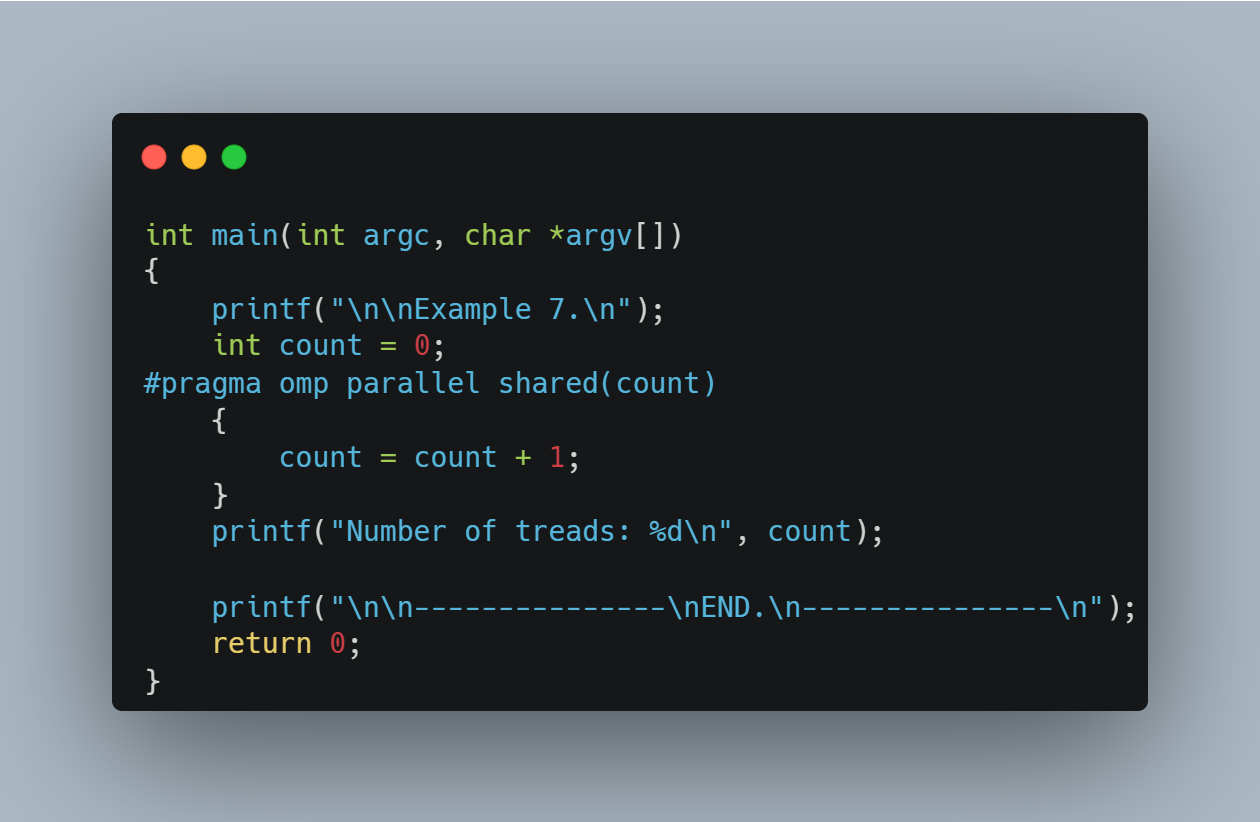


Рисунок 7 - Потоки

1.7. Программу из примера 7 запустить несколько раз. Убедиться в наличии гонок по данным.

Код:



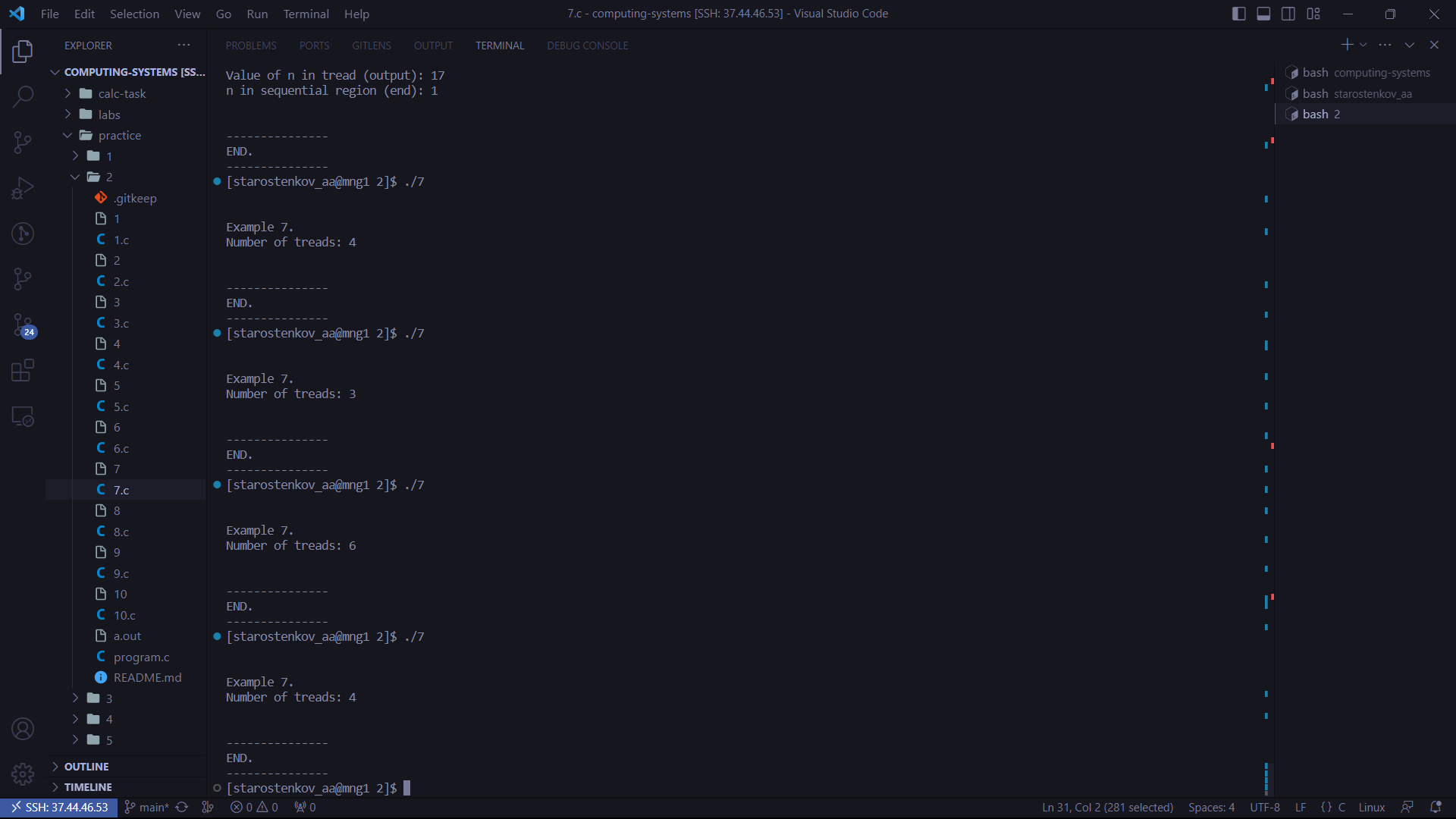


Рисунок 8 – Гонки данных

Наблюдаются гонки данных, выводится в программе то 4, то 5 поток.

1.8. Примеры 9 выполнить с числом потоков в параллельной области, равным номеру по журналу+7. Выполнить программу с опцией reduction и без нее. Объяснить полученные результаты.

Код



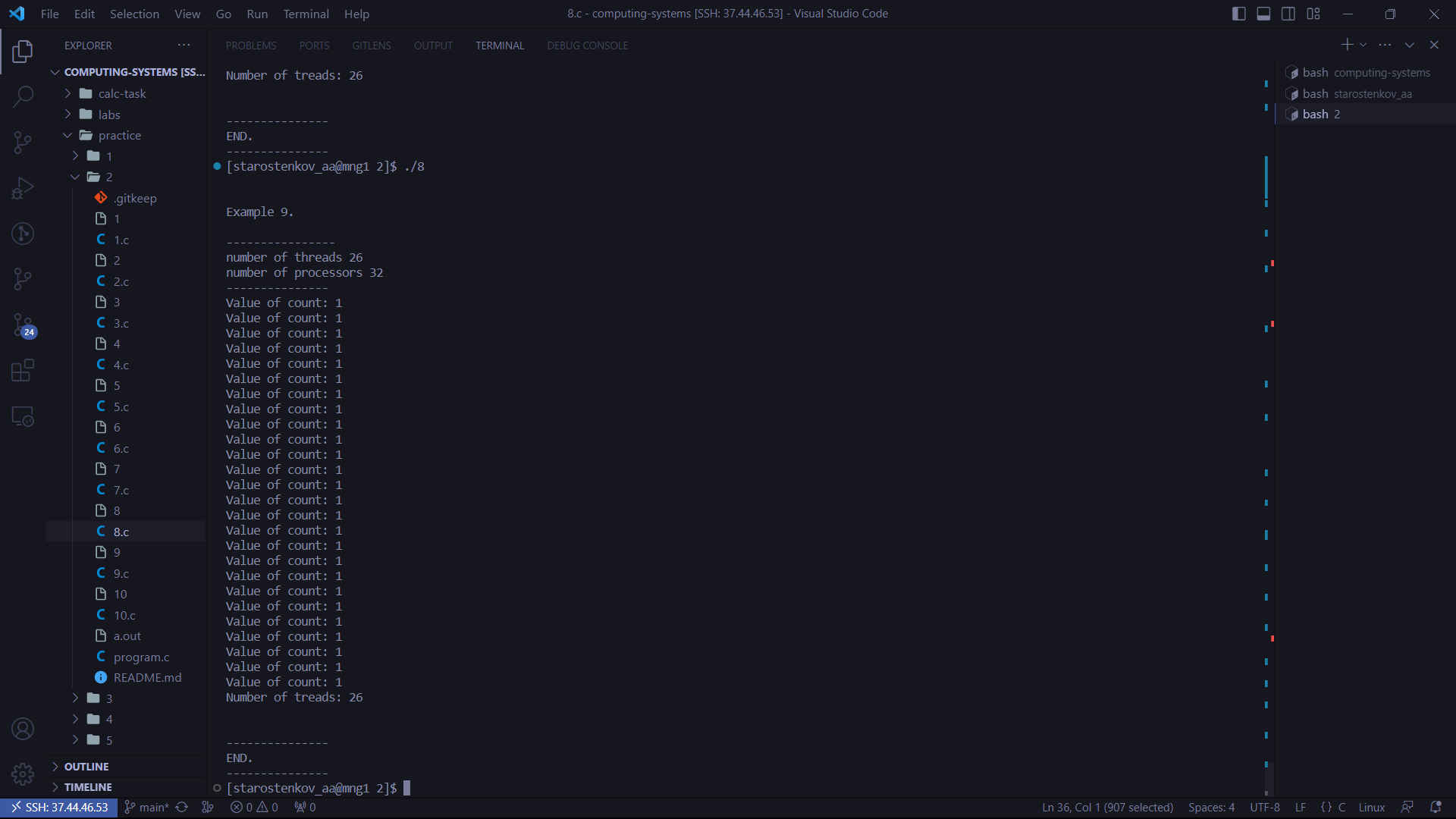


Рисунок 9 - С reduction



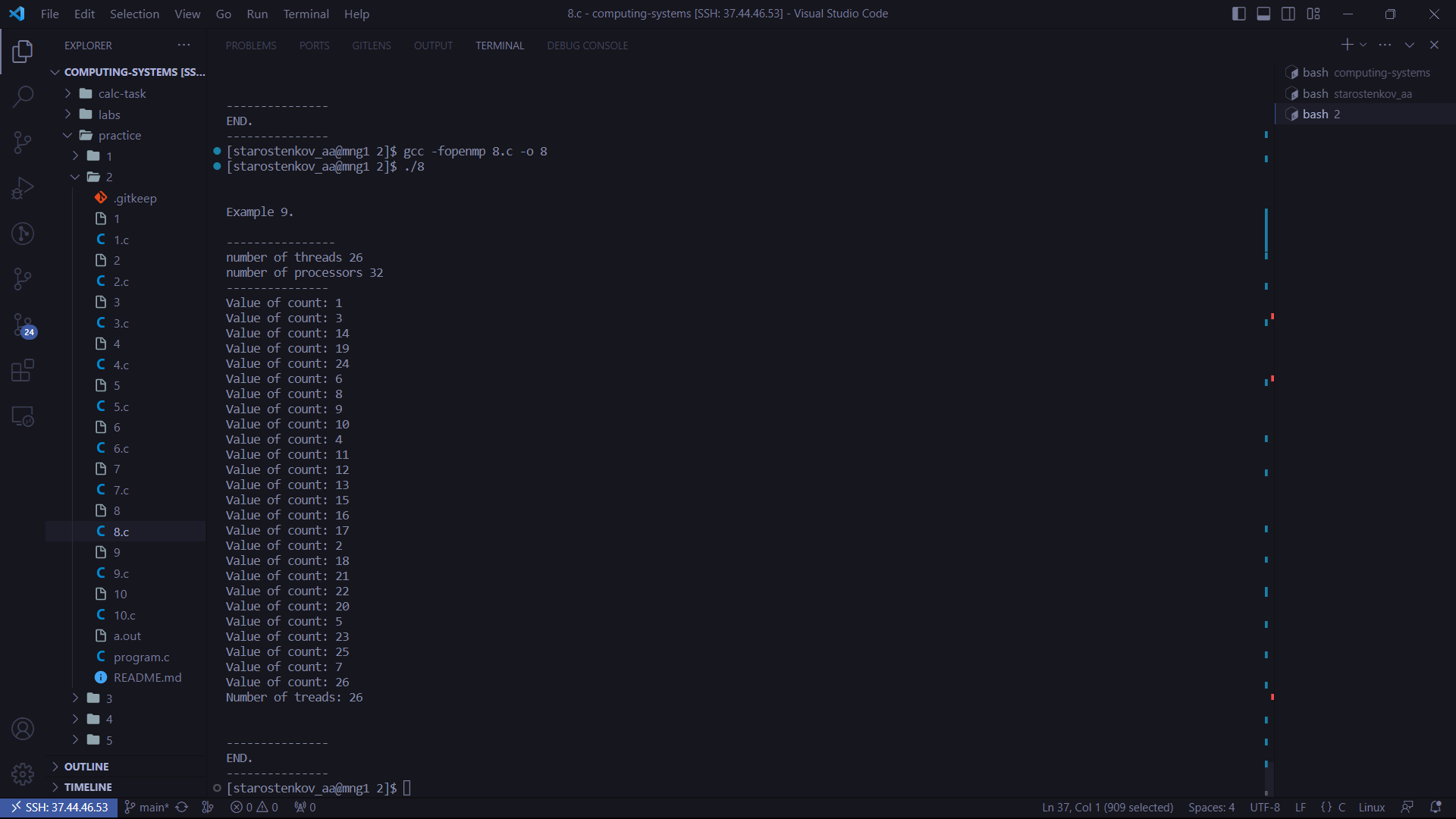


Рисунок 10 – Без reduction

При использовании опции reduction произошло корректное суммирование значений переменной count от каждого потока. Без опции reduction программа превратилась в ошибочную и непредсказуемую: переменная count стала глобальной, потоки стали беспорядочно её перезаписывать, в том числе, «вклиниваясь» во время записи этой же переменной другими потоками. Извлечь полезных результатов из такой программы нельзя.

1.9. Пример 9.1 выполнить без модификаций.

Код



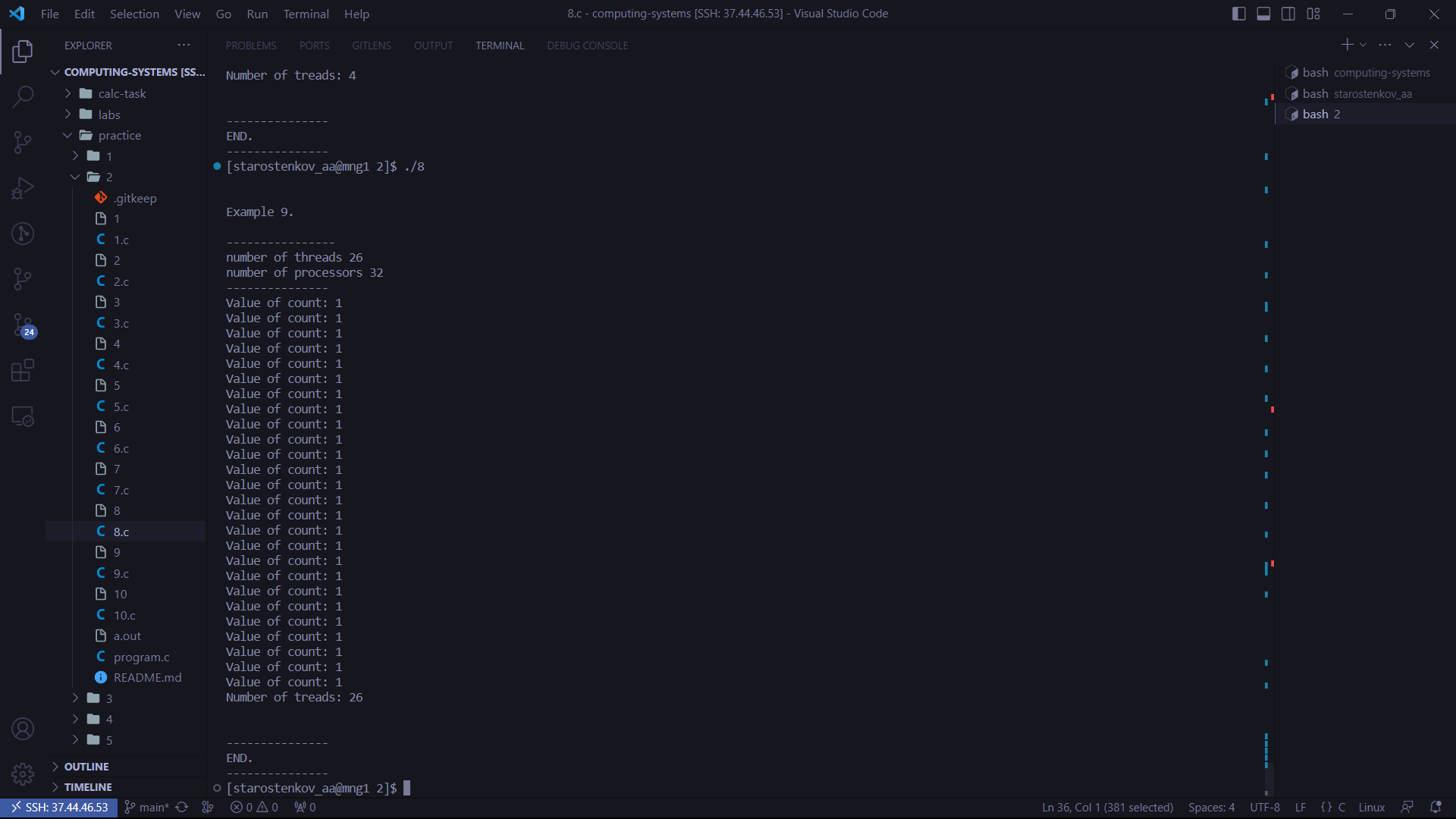


Рисунок 11 - Потоки

1.10. Пример 10 выполнить без модификаций. Определить число итераций цикла, реализованных каждой нитью.

Код



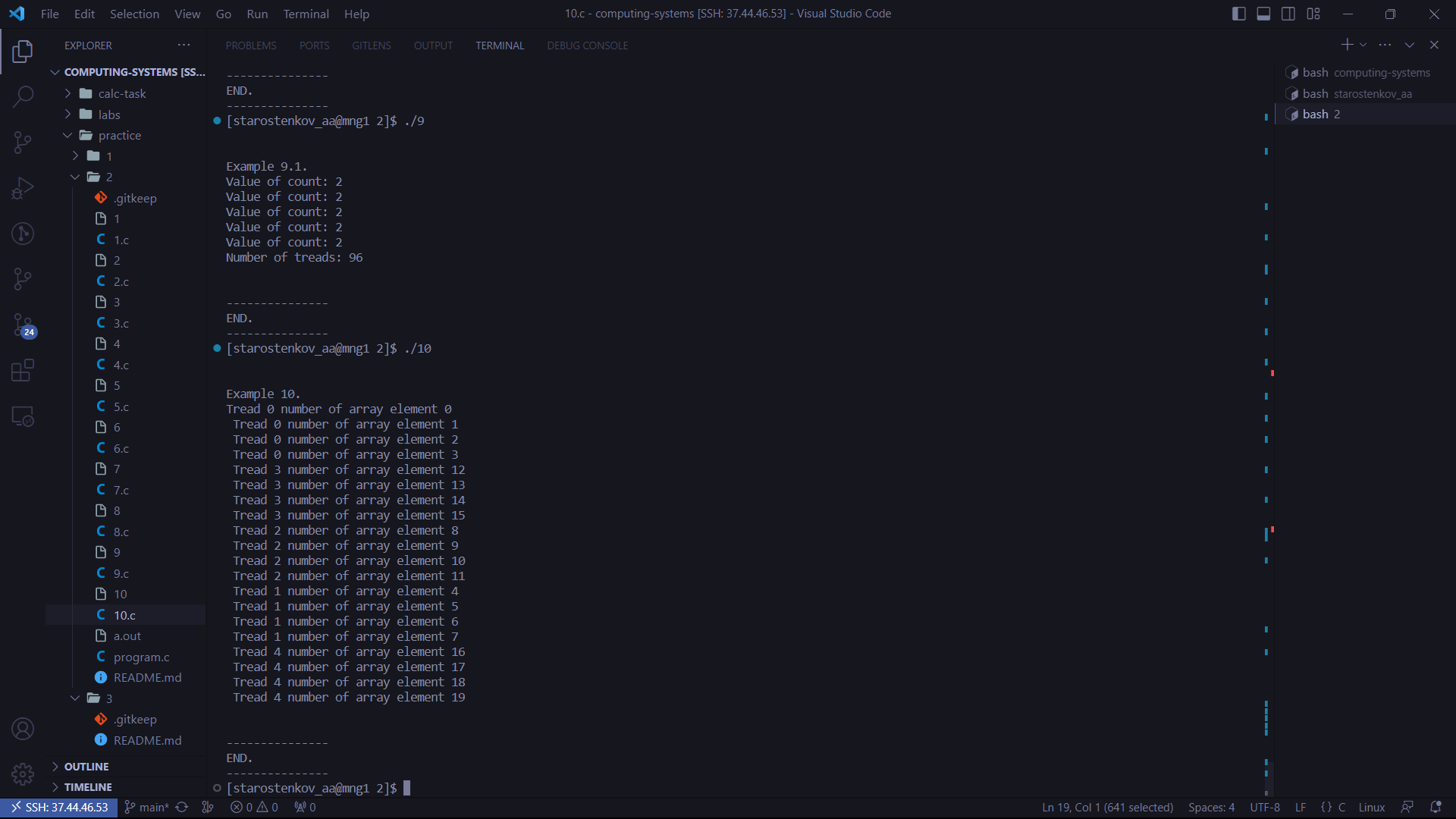


Рисунок 12 – работы директивы for

Эта программа демонстрирует работу директивы for. 5 потоков суммируют два вектора по 20 элементов. OpenMP позаботился о том, чтобы равномерно распределить выполнение цикла между потоками. Число 20 делится на 5, так что каждому потоку выпало равное количество выполнений.