**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение высшего образования «Уральский федеральный университет имени первого Президента России Б.Н. Ельцина

Институт радиоэлектроники и информационных технологий – РТФ

Центр ускоренного обучения

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №12**

**По дисциплине «Программирование»**

Студент группы РИЗ-140938у Бабенко М.А.

Преподаватель: Архипов Н.А.

**Екатеринбург, 2025 г.**

**Лабораторная работа N12**

**«Введение в многопоточность»**

Цель: получить представление о механизме многопоточности в языке программирования Java.

**Задания для самостоятельной работы**

**Задание 1**. Напишите программу, которая запускает два потока и выводит на экран их имена и текущее время в течение 10 секунд.

В главном методе создаются поток 1 и поток 2, в них загружается задача runThread и имя Поток-1 и -2. Затем, они запускаются параллельно

В методе runThread, задается время окончания в мс, для этого используем число long. Далее, построим цикл while – пока время меньше чем время ожидания – потоки продолдают работатьб. Выводится имя каждого потока и текущее время, затем приостанавливаем поток через конструкцию try catch командой Thread.sleep, указываем время в мс. Если поток прервался – выводится соответствующее сообщение в консоль. По завершении тоже будет выведено сообщение.

Результат работы программы и ее код показан на рисунке 1.

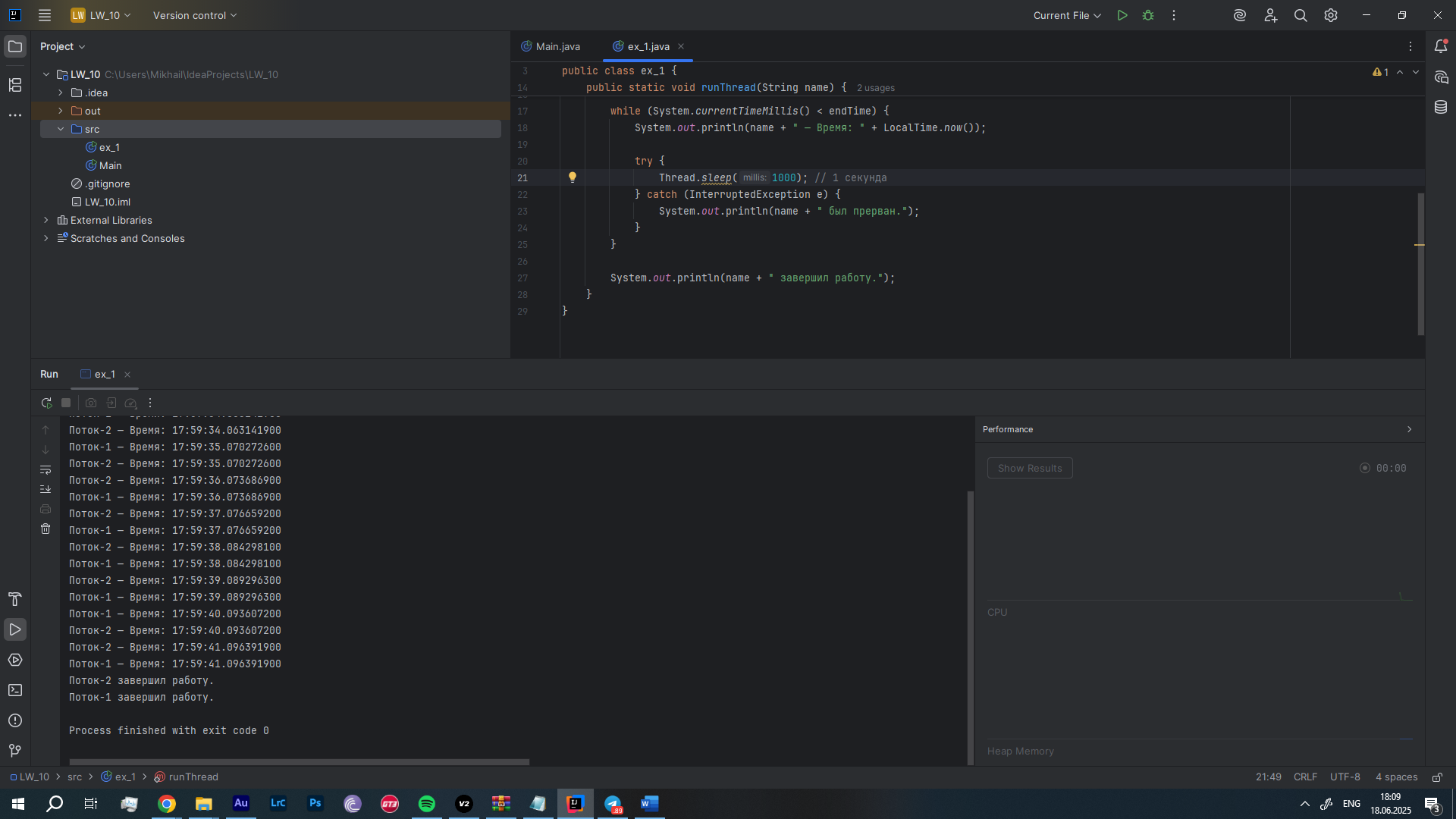


Рисунок 1 – Код и результат работы программы ex\_1

**Задание 2**. Напишите программу, которая запускает поток и выводит на экран числа от 1 до 10 с задержкой в 1 секунду между выводами.

В главном методе создаtтся поток 1, в котором имеется цикл, в котором выводится инкрементированное число, а, далее через конструкцию try catch командой Thread.sleep, указываем время задержки в мс. Если поток прервался – выводится соответствующее сообщение в консоль. По завершении тоже будет выведено сообщение.

Результат работы программы и ее код показан на рисунке 2.

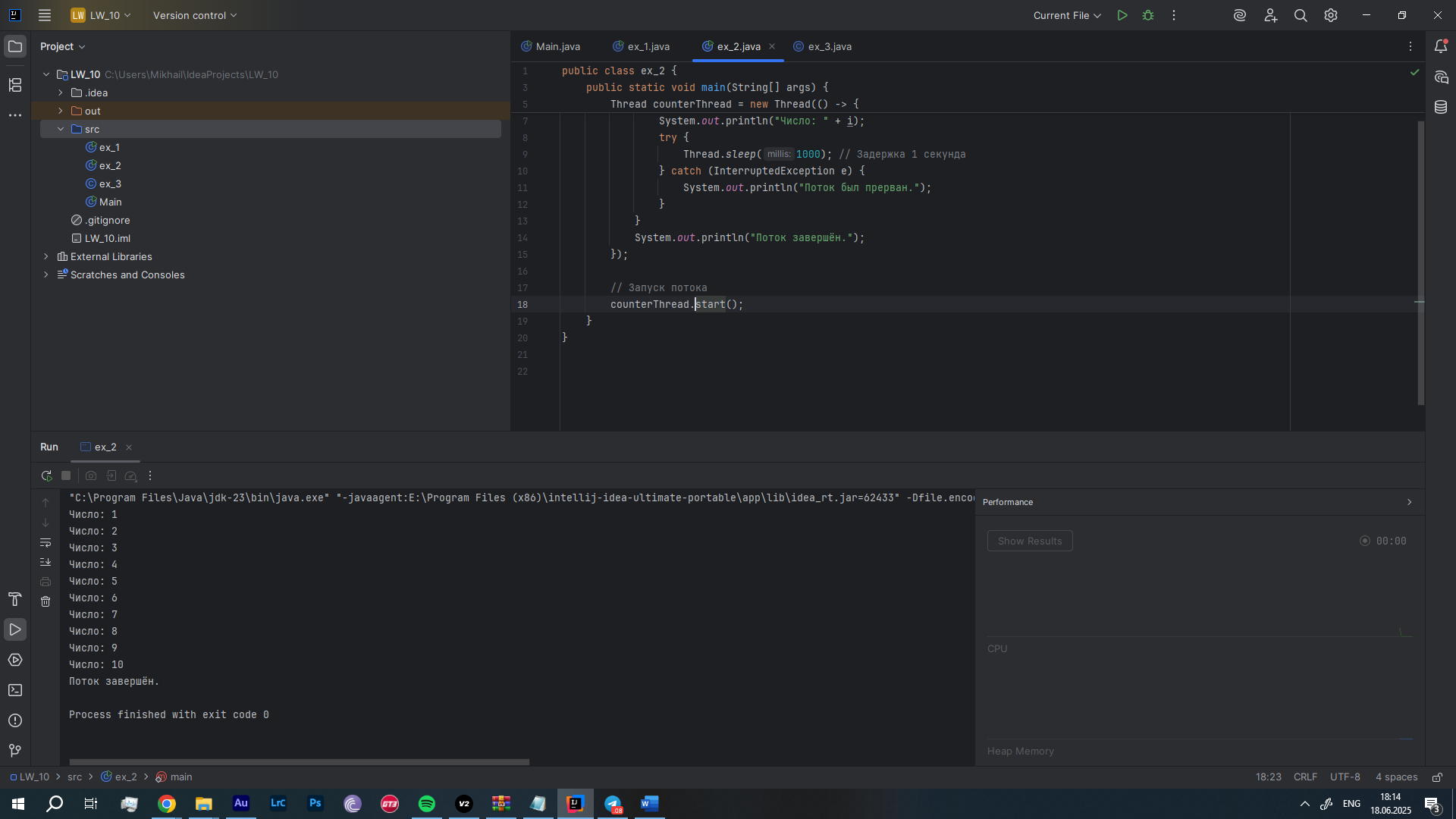


Рисунок 2 – Код и результат работы программы ex\_2

**Задание 3**. Напишите программу, которая создает два потока, которые выводят на экран числа от 1 до 10 Первый поток должен выводить только четные числа, а второй поток должен выводить только нечетные числа.

В главном методе создается поток 1, в котором имеется цикл, в котором выводится увеличивающееся на 2 с 1 число, а, далее через конструкцию try catch командой Thread.sleep, указываем время задержки в мс. Если поток прервался – выводится соответствующее сообщение в консоль.

В главном методе также создается поток 2, в котором имеется цикл, в котором выводится увеличивающееся на 2 с 2 число, а, далее через конструкцию try catch командой Thread.sleep, указываем время задержки в мс. Если поток прервался – выводится соответствующее сообщение в консоль.

По завершении тоже будет выведено сообщение.

Результат работы программы и ее код показан на рисунке 3.

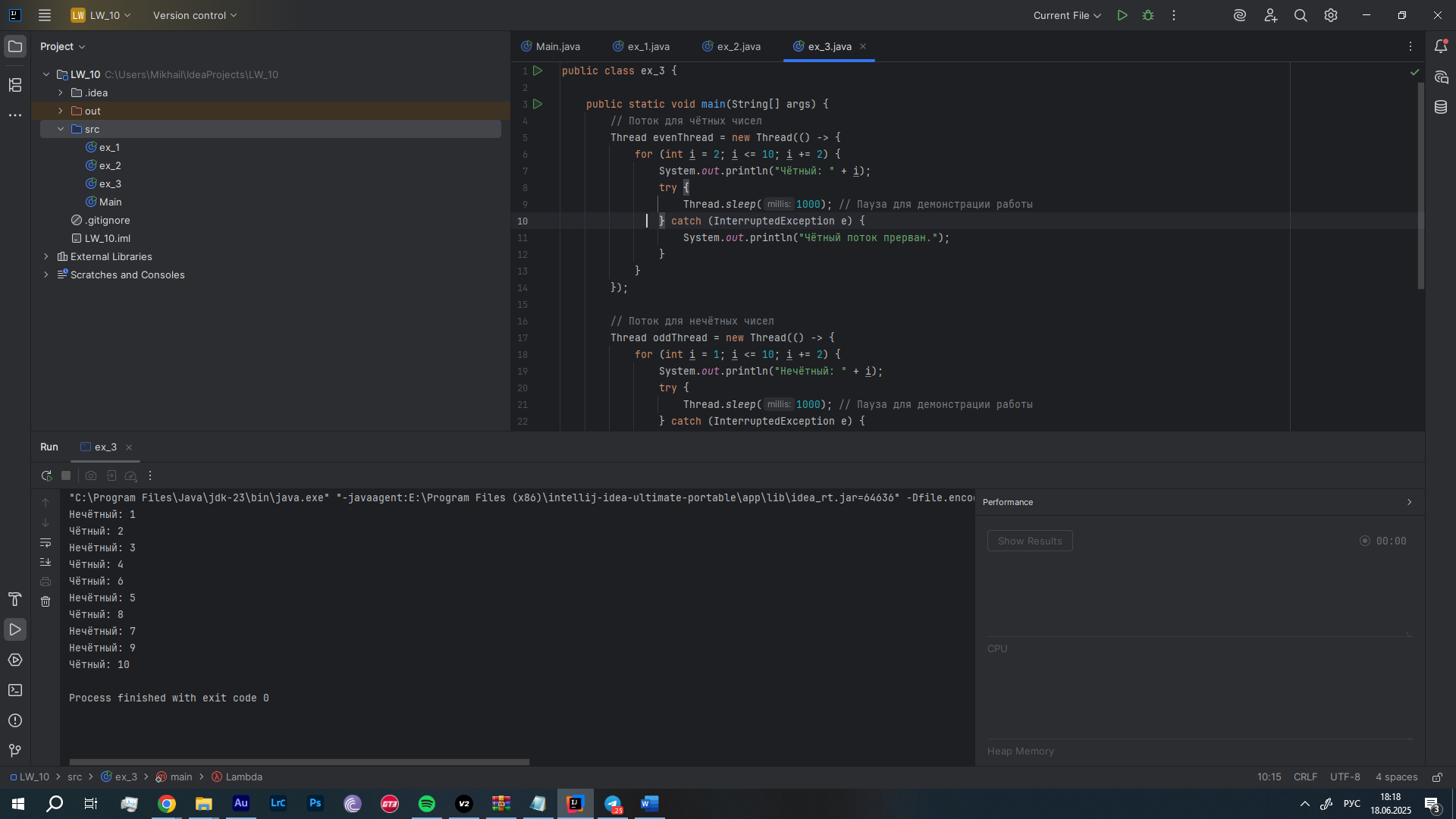


Рисунок 3 – Код и результат работы программы ex\_3

**Задание 4**. Напишите программу, которая создает 10 потоков и каждый поток выводит на экран свой номер.

В главном методе создается цикл, в котором производится счет от 1 до 10, текущее значение i присваивается созданной переменной final int (обязательно), затем создается сам поток, и выводится его номер

Результат работы программы и ее код показан на рисунке 4.

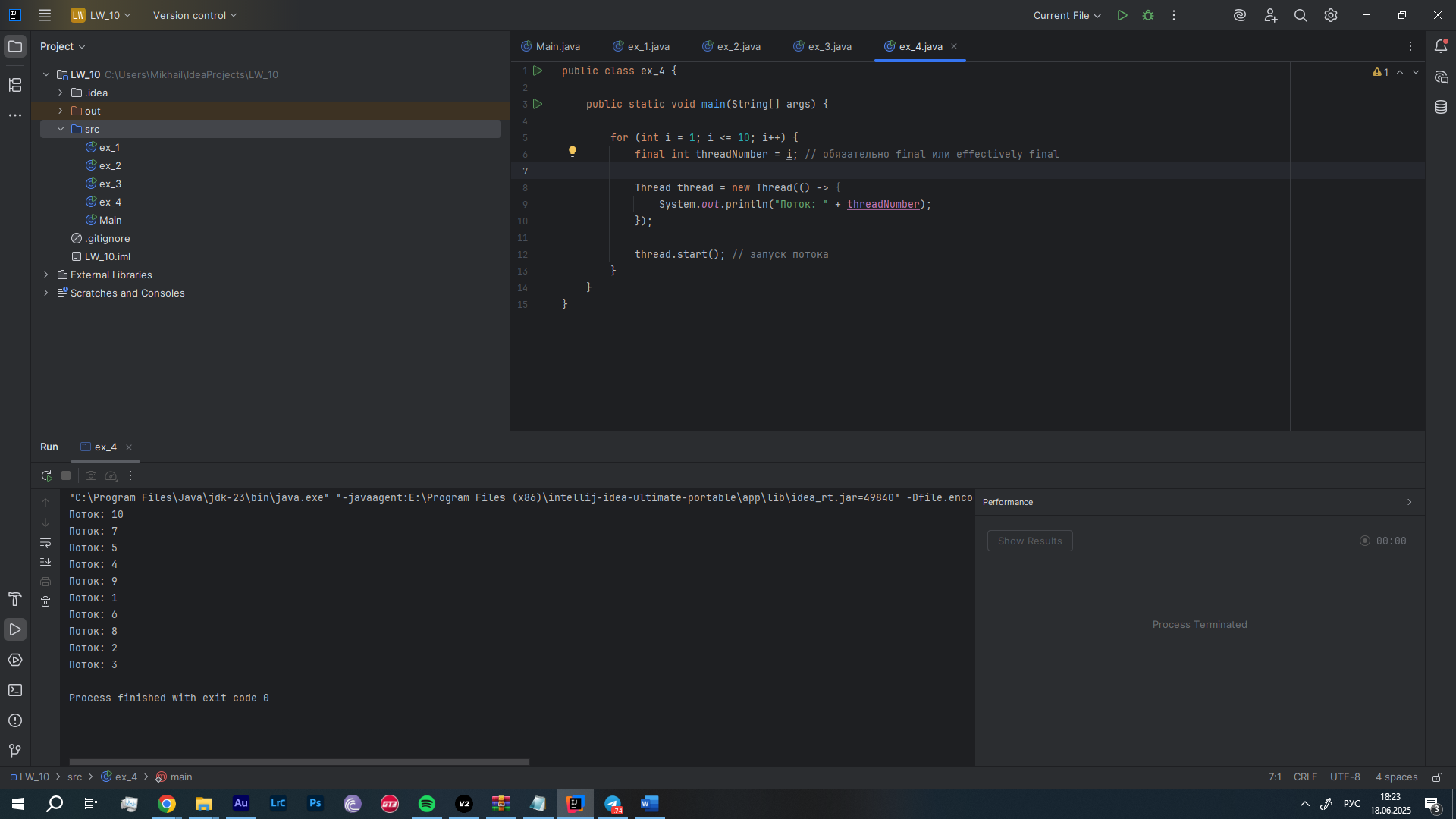


Рисунок 4 – Код и результат работы программы ex\_4

**Задание 5**. Напишите функцию, которая находит максимальный элемент в массиве целых чисел при помощи многопоточности. Количество потоков должно быть равно количеству ядер процессора.

В классе создается статичный метод, в котором будет совершен поиск максимального значения. Создается переменная типа int numThreads, которой будет присвоено значение доступных потоков процессора. Размер куска массива определяется chuckSize, а Math.ceil отвечает за то, чтобы все элементы массива попали в обработку, это округление количества элементов массива.

Затем создается пул потоков. ExecutorService упрощает управление потоками, а newFixedThreadPool(numThreads) создает фиксированное число потоков. Циклом запускается numThreads задач, внутри каждой получается свой подмассив, ищется локальные макс значения и возвращается. Далее, после завершения текущих задач получаются с каждого потока локальные максимумы, они сравниваются, и находится наибольший; после чего пул закрывается командой executor.shutdown();.

Результат работы программы и ее код показан на рисунке 5.

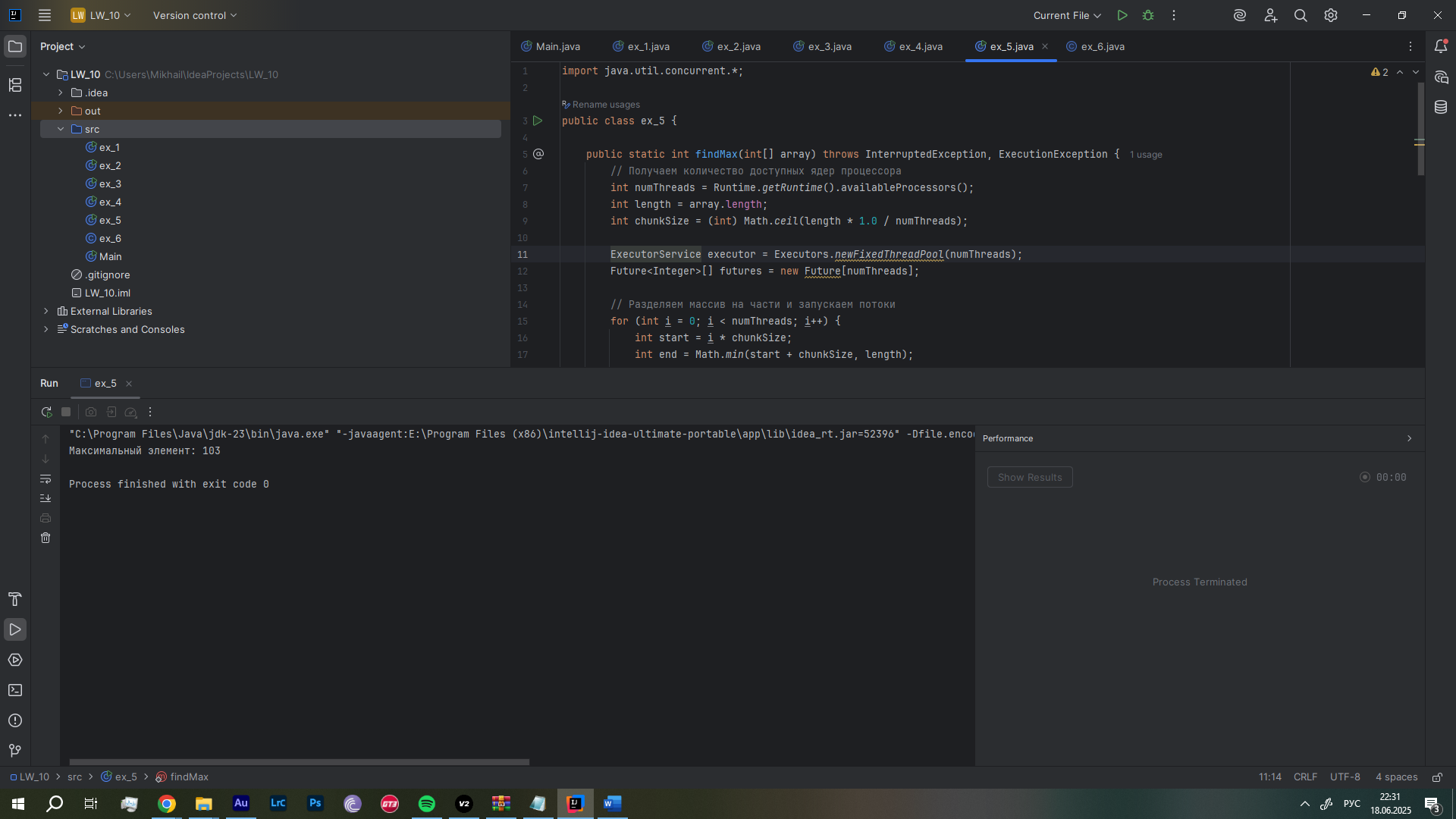


Рисунок 5 – Код и результат работы программы ex\_5

**Задание 6**. Напишите функцию, которая суммирует элементы в массиве целых чисел при помощи многопоточности. Количество потоков должно быть равно количеству ядер процессора.

В классе создается статичный метод, в котором будет совершен поиск суммы всех значений. Создается переменная типа int numThreads, которой будет присвоено значение доступных потоков процессора. Размер куска массива определяется chuckSize, а Math.ceil отвечает за то, чтобы все элементы массива попали в обработку, это округление количества элементов массива.

Затем создается пул потоков. ExecutorService упрощает управление потоками, а newFixedThreadPool(numThreads) создает фиксированное число потоков. Циклом запускается numThreads задач, внутри каждой получается свой подмассив, ищется сумма части массива. Далее, после завершения текущих задач получаются с каждого потока локальные суммы, они складываются, и сохраняются в totalsum; после чего пул закрывается командой executor.shutdown();.

Результат работы программы и ее код показан на рисунке 6.

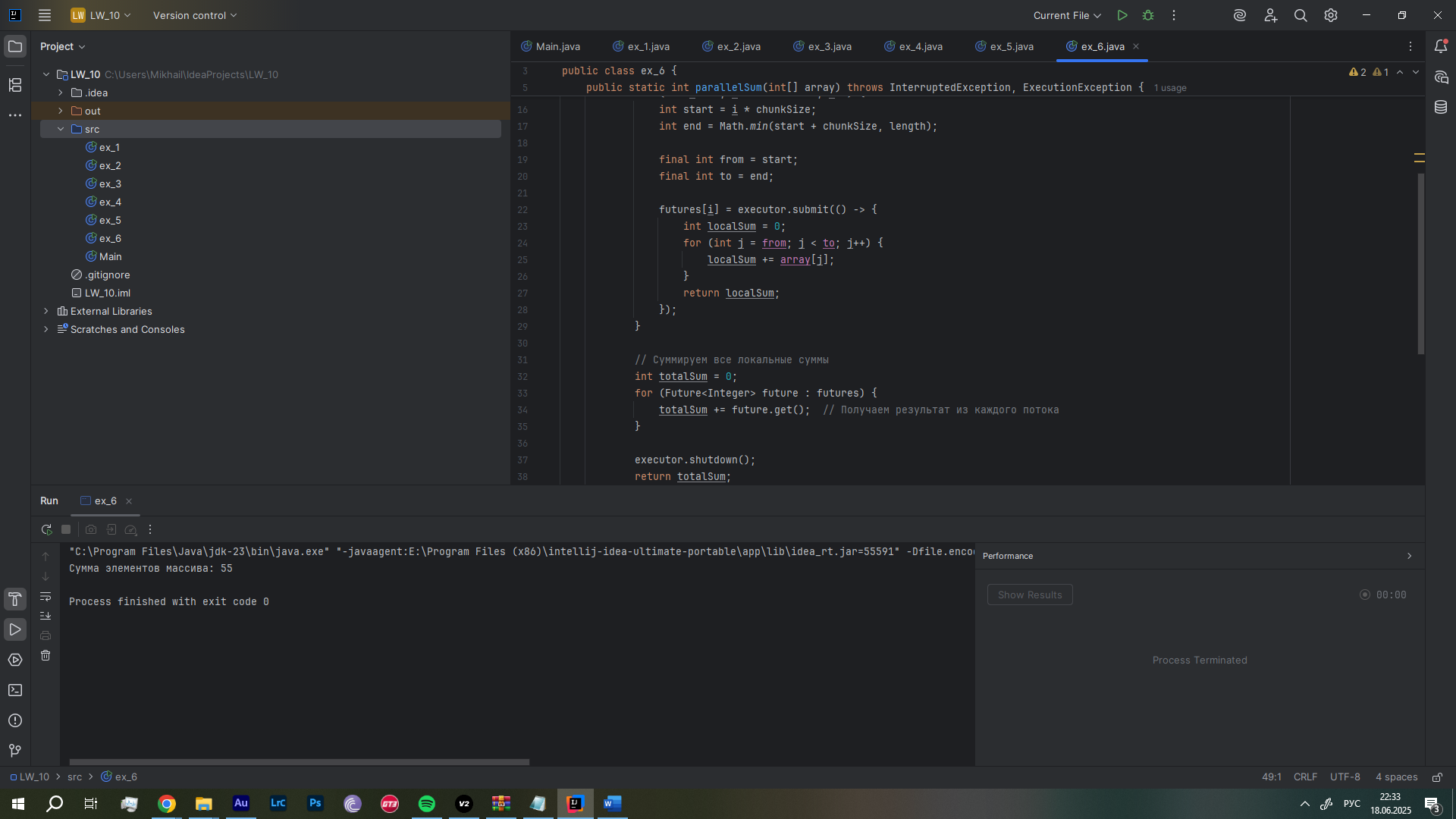


Рисунок 6 – Код и результат работы программы ex\_2

**Вывод**

В результате лабораторной работы были получены представления о механизме многопоточности в языке программирования Java.