**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 №11**

**По дисциплине «Программирование»**

Студент группы РИЗ-140938у Бабенко М.А.

Преподаватель: Архипов Н.А.

**Екатеринбург, 2025 г.**

**Лабораторная работа N11**

**«Введение в функциональное программирование и Stream API»**

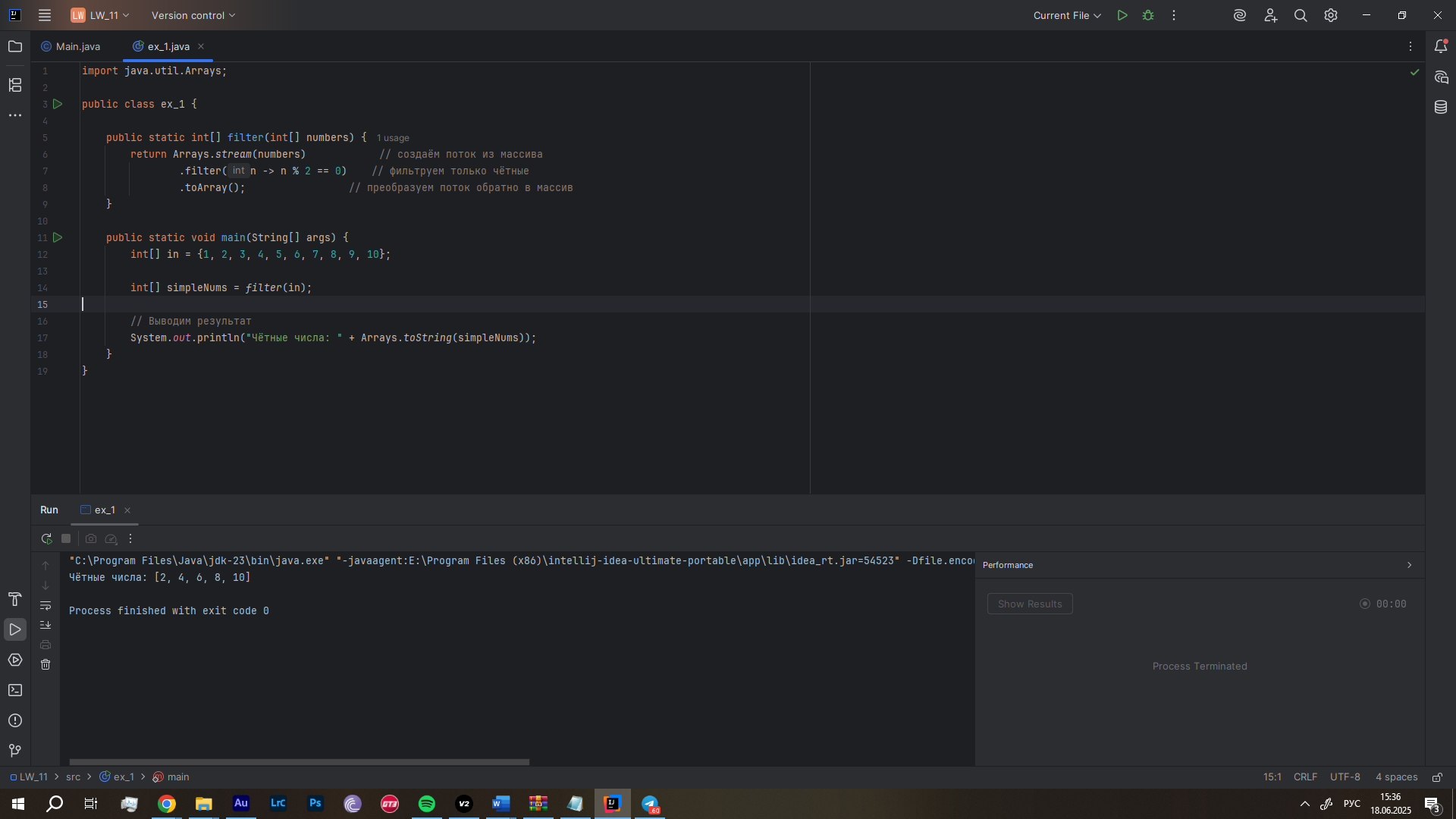
Цель: получить представление о функциональном программировании и Stream API

**Задания для самостоятельной работы**

**Задание 1.** Напишите функцию, которая принимает на вход массив целых чисел и возвращает новый массив, содержащий только четные числа из исходного массива.

В **ex\_1.java** записывается функция, фильтрующая числа массива. Создается класс ex\_1, в котором создается метод, возвращающий массив данных. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет массив данных. В нем происходит возврат созданного потока чисел Arrays.stream, в котором используются выражения, оставляющие и записывающие в массив только те n, которые делятся на два без остатка. Далее, выражением .toArray преобразуется поток обратно в массив.

Результат работы программы и ее код показан на рисунке 1.



В **ex\_2.java** записывается функция, фильтрующая числа массива. Создается класс ex\_2, в котором создается метод, возвращающий массив данных. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет массив данных. Сначала записанные в массив числа преобразуются из int в Integer, при помощи Arrays.stream. Преобразование происходит по команде .boxed, затем массив собирается в Set.

Затем происходит возврат созданного потока чисел Arrays.stream, в котором используются выражения, оставляющие и записывающие в массив только те n, которыеесть в обоих массивах. Далее, выражением .toArray преобразуется поток обратно в массив и записывается в переменную common.

Результат работы программы и ее код показан на рисунке 2.

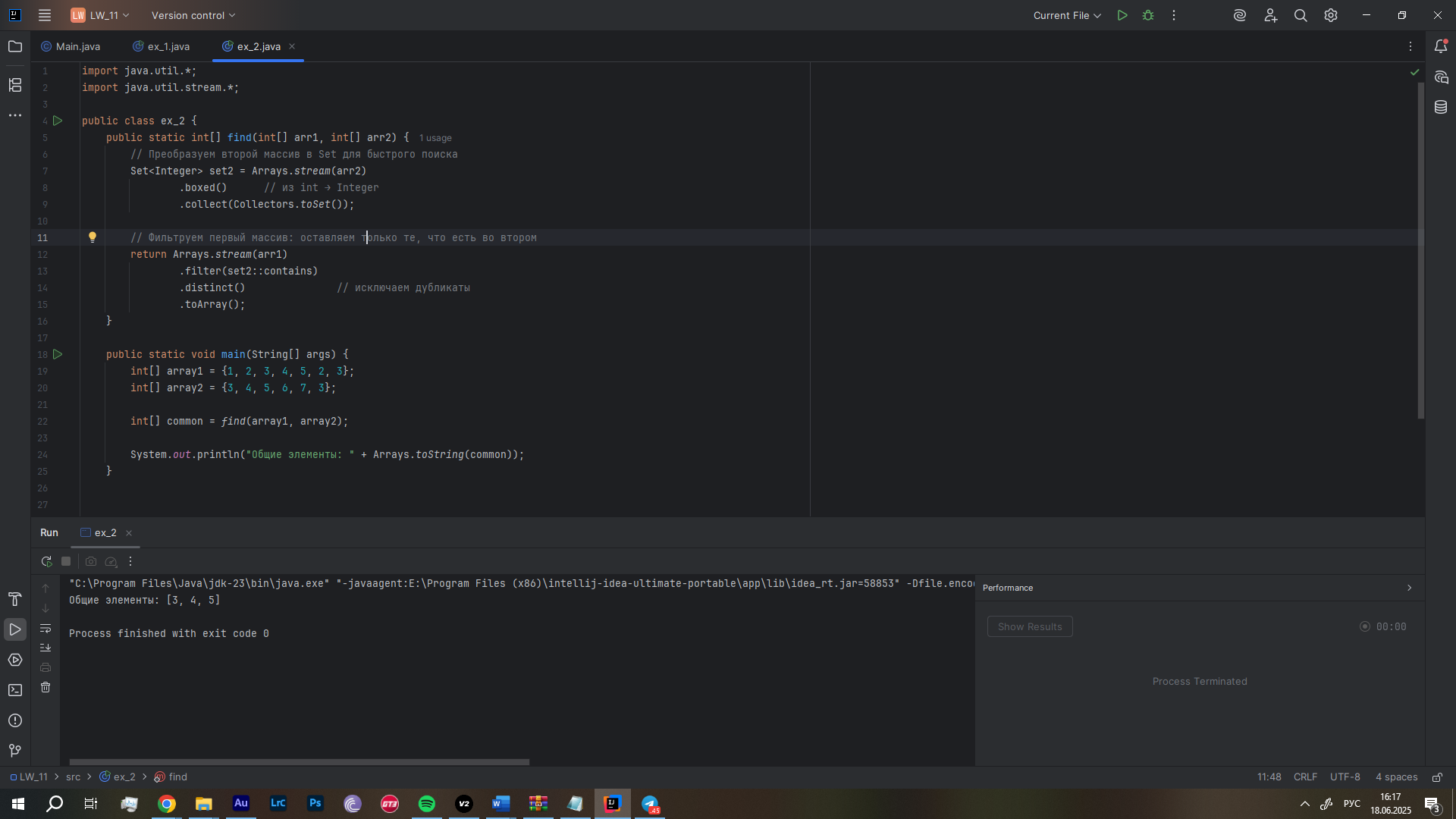


Рисунок 2 – Код и результат работы программы ex\_2

В **ex\_3.java** записывается функция, фильтрующая строки. Создается класс **ex\_2**, в котором создается метод, возвращающий список строк. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет список строк.

Затем происходит возврат созданного потока списка строк input.stream, в котором используются выражения, оставляющие и записывающие в массив только те строки, начинающиеся с большой буквы. Фильтрация такова: .filter – служебное слово, !s.isEmpty() – означает, что строка не пуста; **Character.isUpperCase(s.charAt(0))** – означает, что буква заглавная. Далее, выражением .toList преобразуется поток обратно в список и записывается в переменную capitalized.

Результат работы программы и ее код показан на рисунке 3.

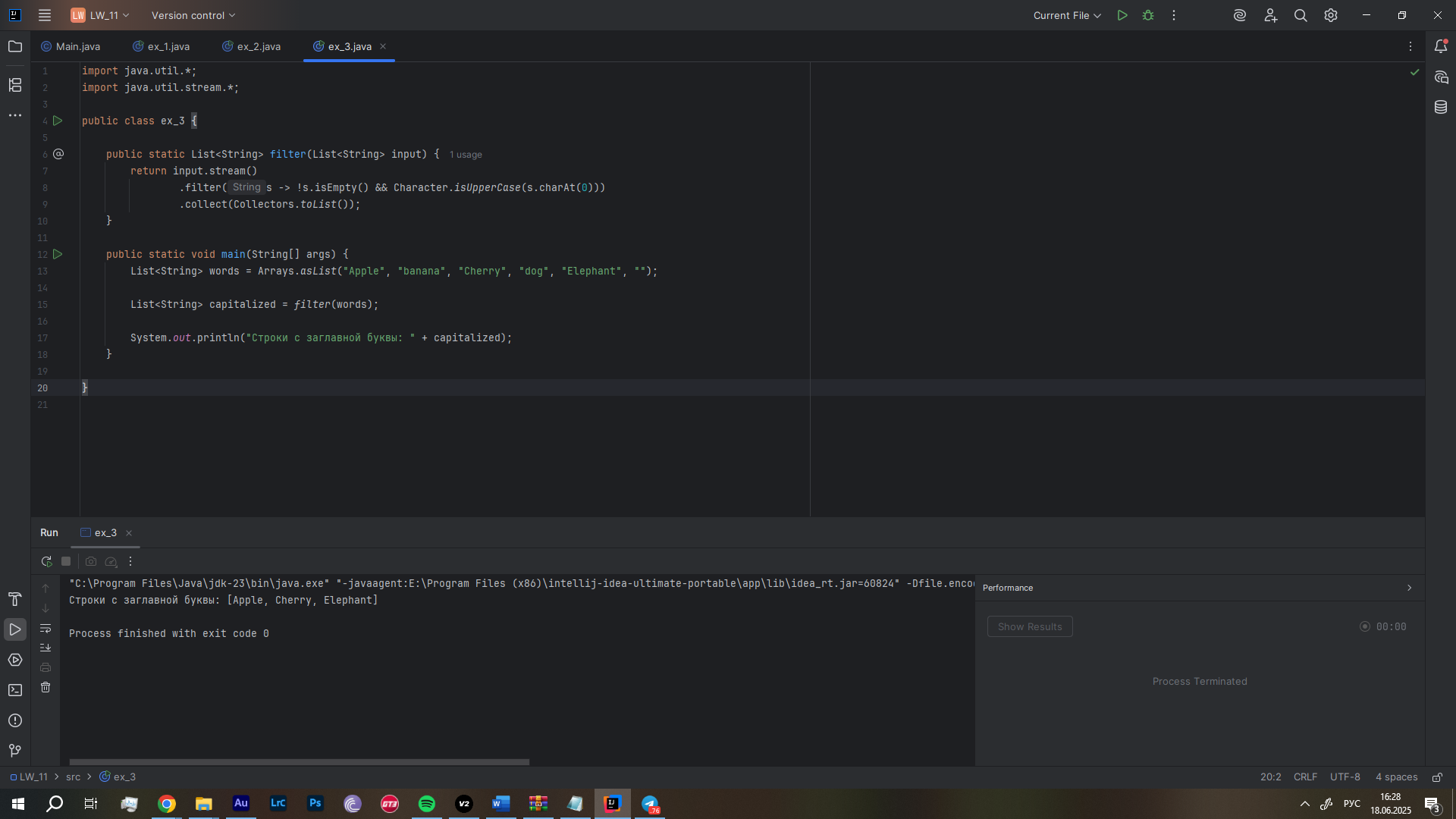


Рисунок 3 – Код и результат работы программы ex\_3

В **ex\_4.java** записывается функция, фильтрующая числа. Создается класс **ex\_4**, в котором создается метод, возвращающий список чисел. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет список чисел.

Затем происходит возврат созданного потока списка строк input.stream, в котором используются выражения, возводящие в квадрат каждое число. Далее, выражением .toList преобразуется поток обратно в список и записывается в переменную squared.

Результат работы программы и ее код показан на рисунке 4.

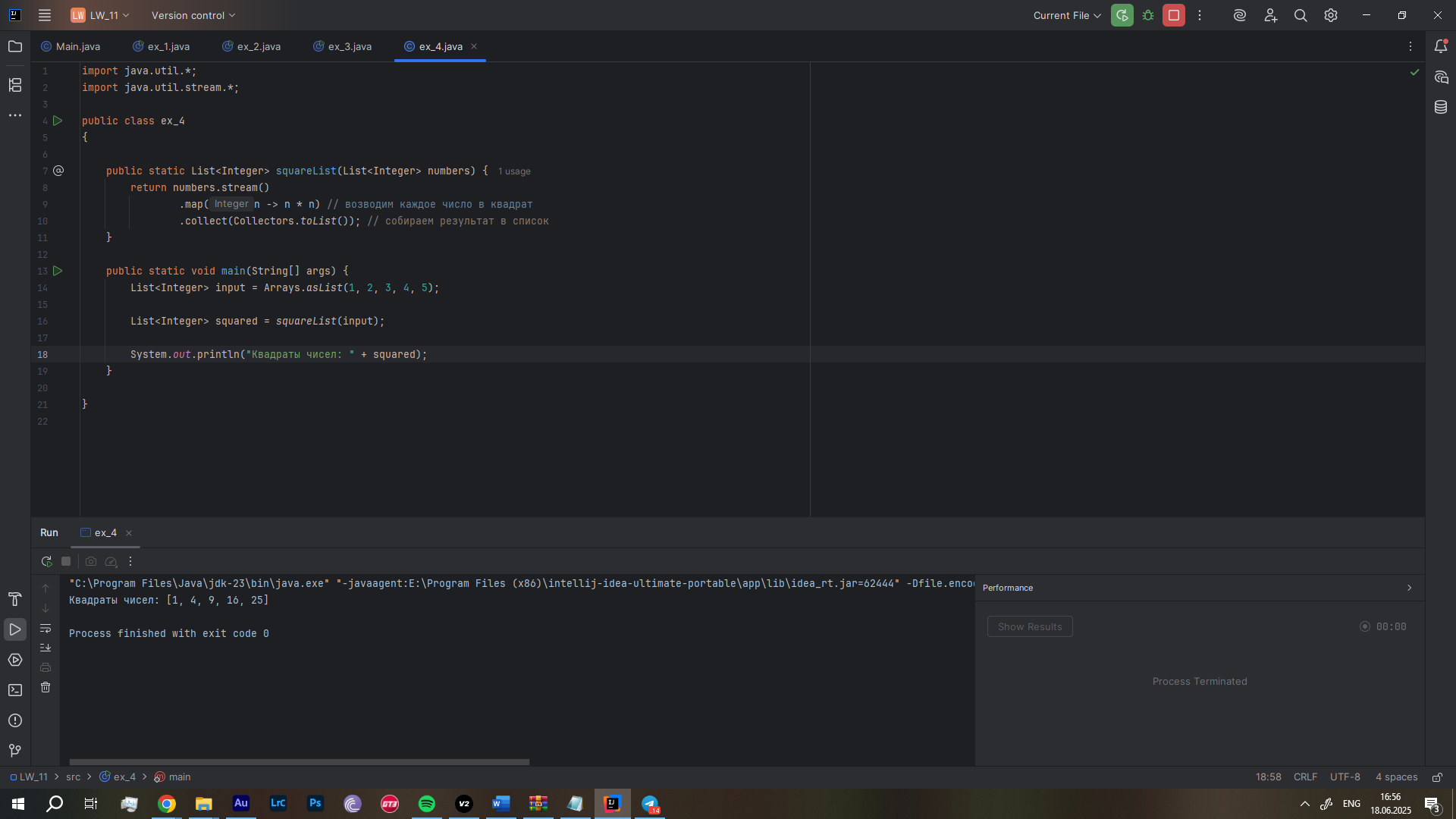


Рисунок 4 – Код и результат работы программы ex\_4

В **ex\_5.java** записывается функция, фильтрующая строки. Создается класс **ex\_5**, в котором создается метод, возвращающий список строк. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет список строк.

Затем происходит возврат созданного потока списка строк input.stream, в котором используются выражения, оставляющие и записывающие список только те строки, которые содержат искомое значение заданной построки. Фильтрация такова: .filter – служебное слово, проверяется содержат ли строки субстроку. Далее, метод вызывается из main.

Код программы и ее работа показаны на рисунке 5.

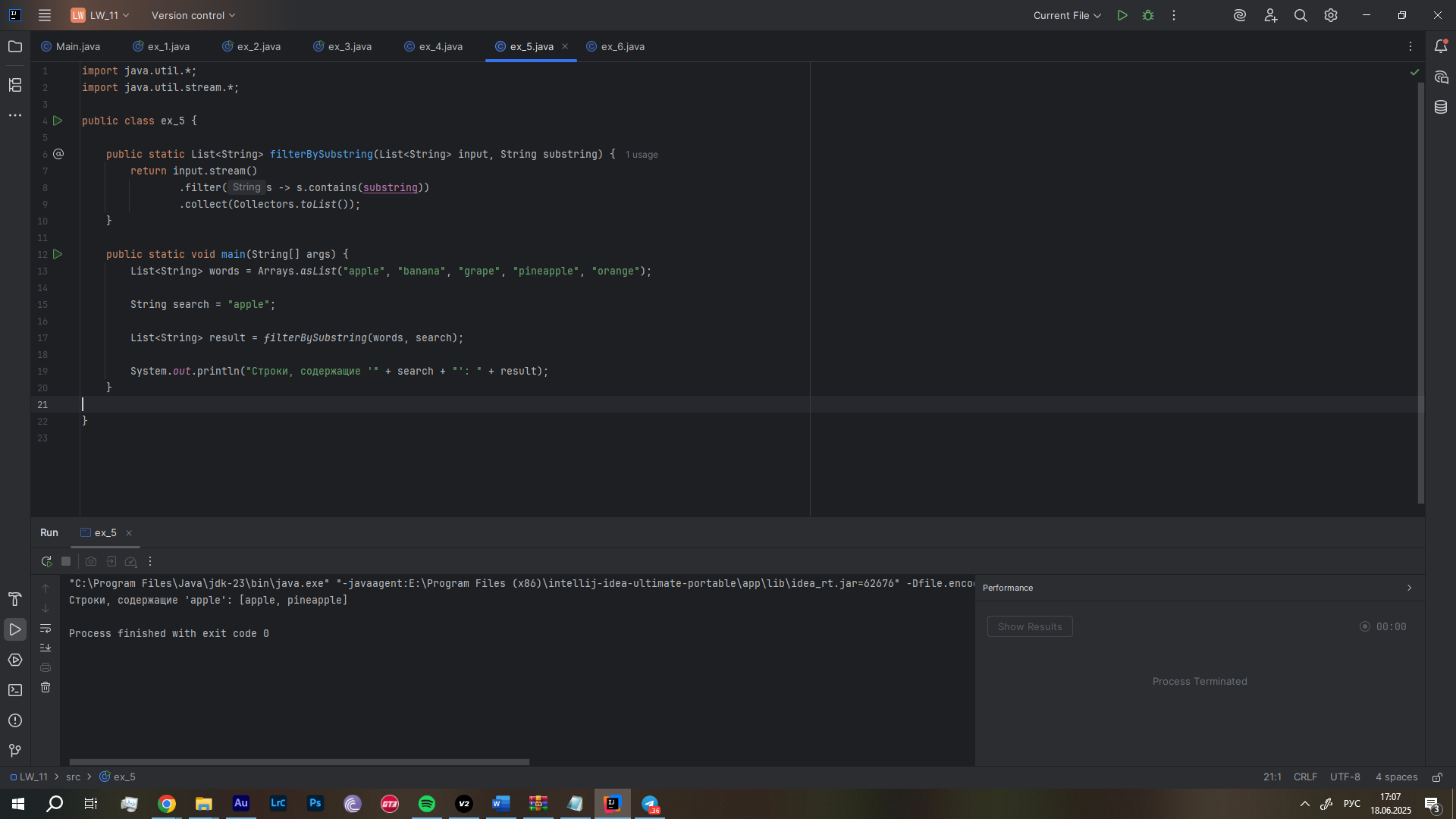


Рисунок 5 – Код и результат работы программы ex\_5

**Задание 6**. Напишите функцию, которая принимает на вход список целых чисел и возвращает новый список, содержащий только те числа, которые делятся на заданное число без остатка.

В **ex\_6.java** записывается функция, фильтрующая числа. Создается класс **ex\_6**, в котором создается метод, возвращающий список чисел. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет список чисел.

Затем происходит возврат созданного потока списка строк input.stream, в котором используются выражения, сверяющее делится ли без остатка выражение на заданную переменную. Далее, выражением .toList преобразуется поток обратно в список и вызывается методом из main.

Результат работы программы и ее код показан на рисунке 6.

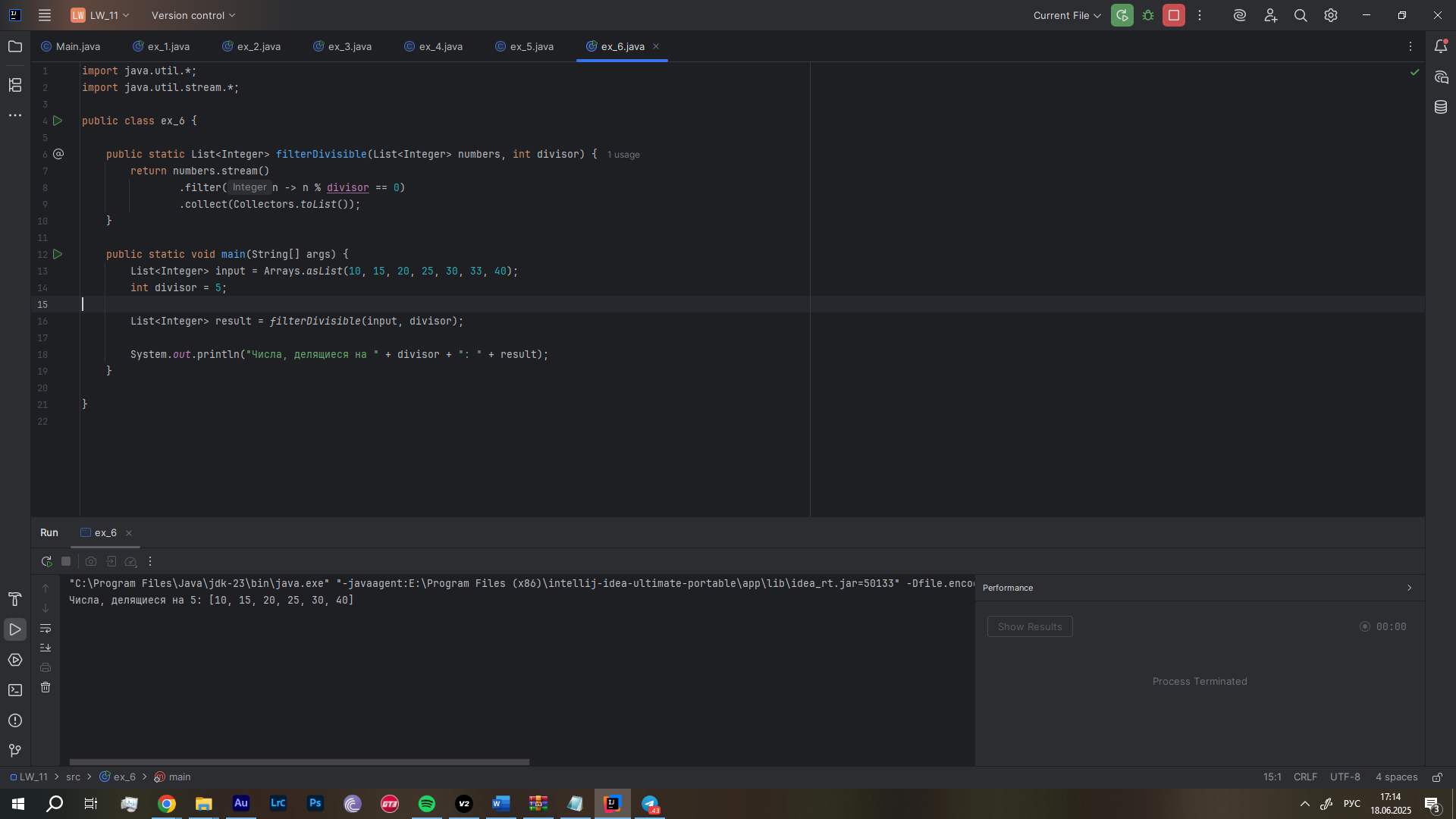


Рисунок 6 – Код и результат работы программы ex\_6

**Задание 7**. Напишите функцию, которая принимает на вход список строк и возвращает новый список, содержащий только те строки, которые имеют длину больше заданного значения.

В **ex\_7.java** записывается функция, фильтрующая строки. Создается класс **ex\_7**, в котором создается метод, возвращающий список строк. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет список строк.

Затем происходит возврат созданного потока списка строк input.stream, в котором используются выражения, оставляющие и записывающие список только те строки, которые длиннее заданного значения. Фильтрация такова: **.filter** – служебное слово, длиннее ли они чем переменна **minLength**. Далее, метод вызывается из main.

Результат работы программы и ее код показан на рисунке 7.

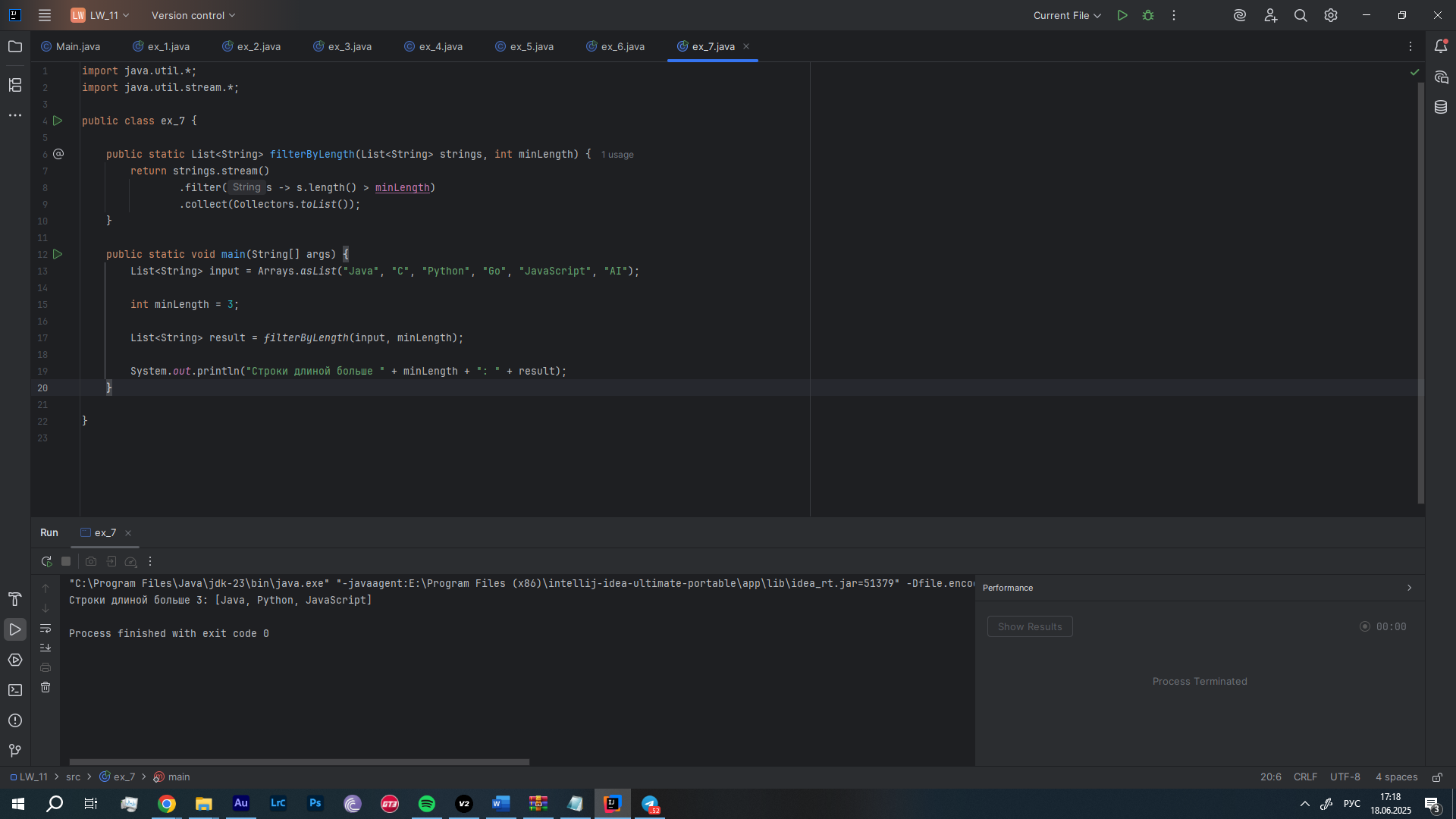


Рисунок 7 – Код и результат работы программы ex\_7

**Задание 8.** Напишите функцию, которая принимает на вход список целых чисел и возвращает новый список, содержащий только те числа, которые больше заданного значения.

В ex\_8.java записывается функция, фильтрующая числа. Создается класс ex\_8, в котором создается метод, возвращающий список чисел. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет список чисел.

Затем происходит возврат созданного потока списка чисел input.stream, в котором используются выражения, сверяющее больше ли числа заданного значения. Далее, выражением .toList преобразуется поток обратно в список и записывается в result.

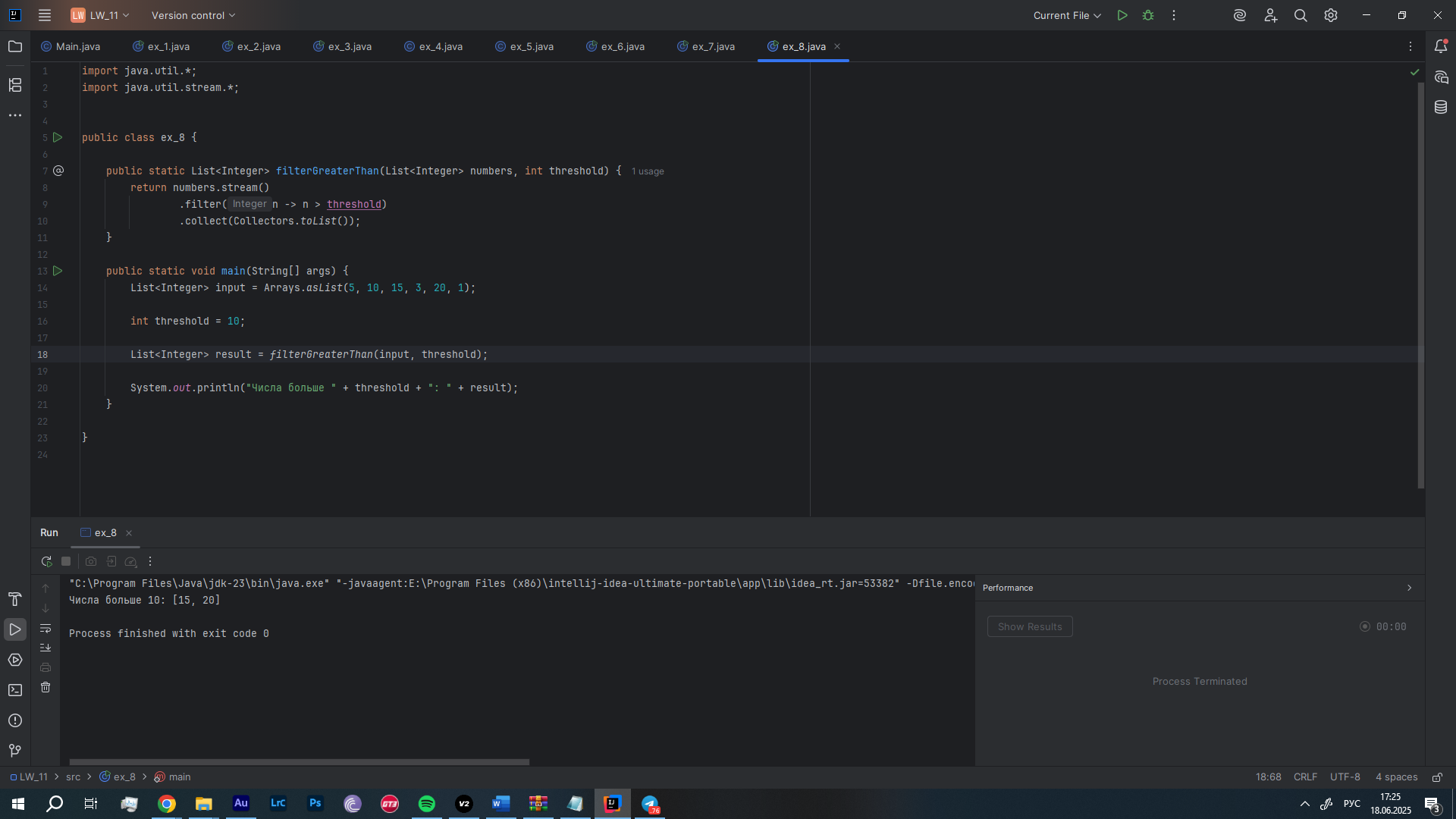


Рисунок 8 – Код и результат работы программы ex\_8

**Задание 9.** Напишите функцию, которая принимает на вход список строк и возвращает новый список, содержащий только те строки, которые содержат только буквы (без цифр и символов).

В **ex\_9.java** записывается функция, фильтрующая строки. Создается класс **ex\_9**, в котором создается метод, возвращающий список строк. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет список строк.

Затем происходит возврат созданного потока списка строк input.stream, в котором используются выражения, оставляющие и записывающие список только те строки, которые длиннее заданного значения. Фильтрация такова: **.filter** – служебное слово, затем проверяется, какие символы содержит строка, если в ней только буквы – они записываются в поток, который преобразуется в список. Далее, метод вызывается из main и записывается в result.

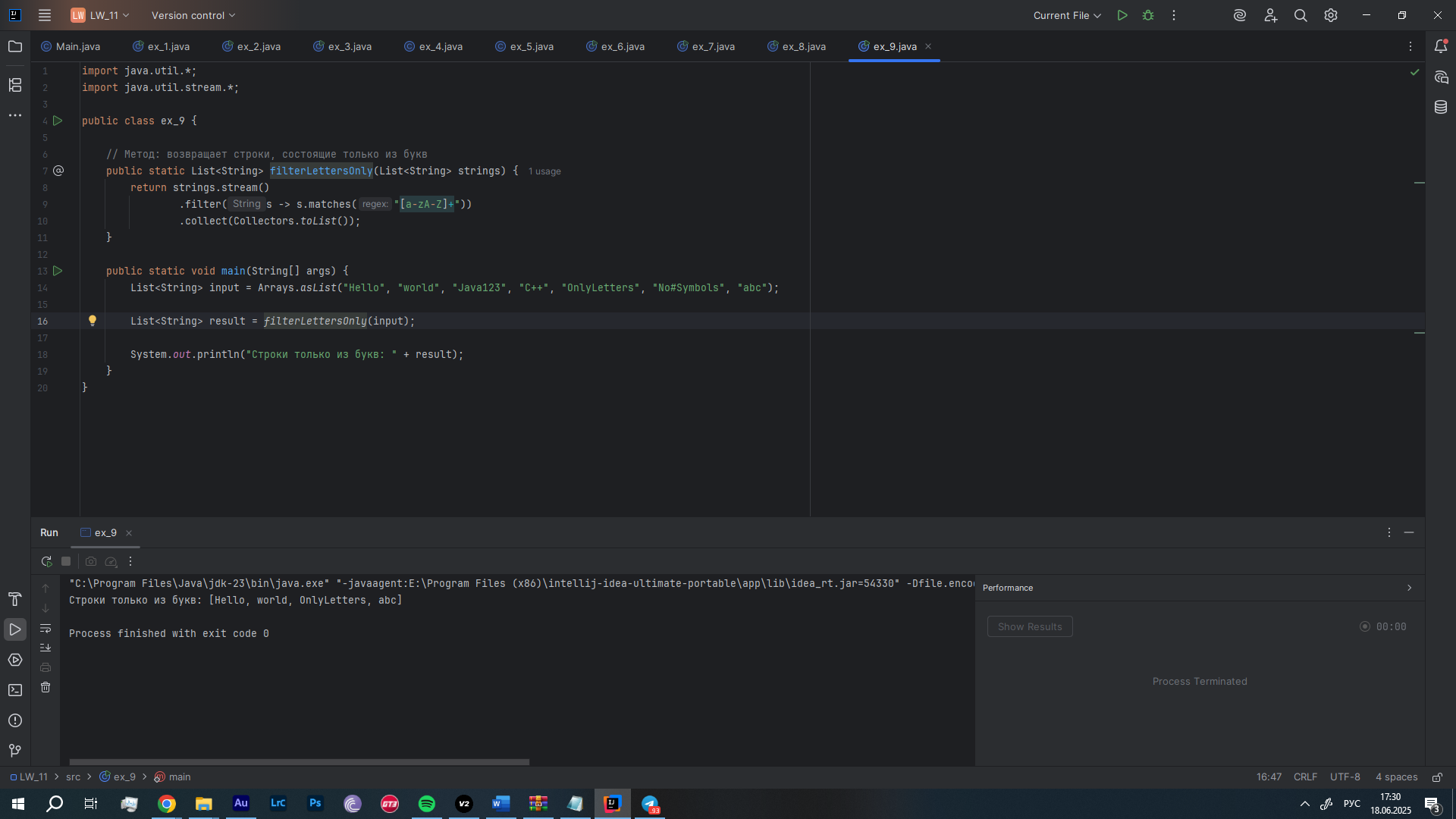


Рисунок 9 – Код и результат работы программы ex\_9

**Задание 10**. Напишите функцию, которая принимает на вход список целых чисел и возвращает новый список, содержащий только те числа, которые меньше заданного значения.

В ex\_10.java записывается функция, фильтрующая числа. Создается класс ex\_10, в котором создается метод, возвращающий список чисел. Делаем его публичным статическим, чтобы он был доступен извне, аргументом для него будет список чисел.

Затем происходит возврат созданного потока списка чисел input.stream, в котором используются выражения, сверяющее больше ли элементы заданного переменной значения. Далее, выражением .toList преобразуется поток обратно в список и вызывается методом из main.

Результат работы программы и ее код показан на рисунке 17.

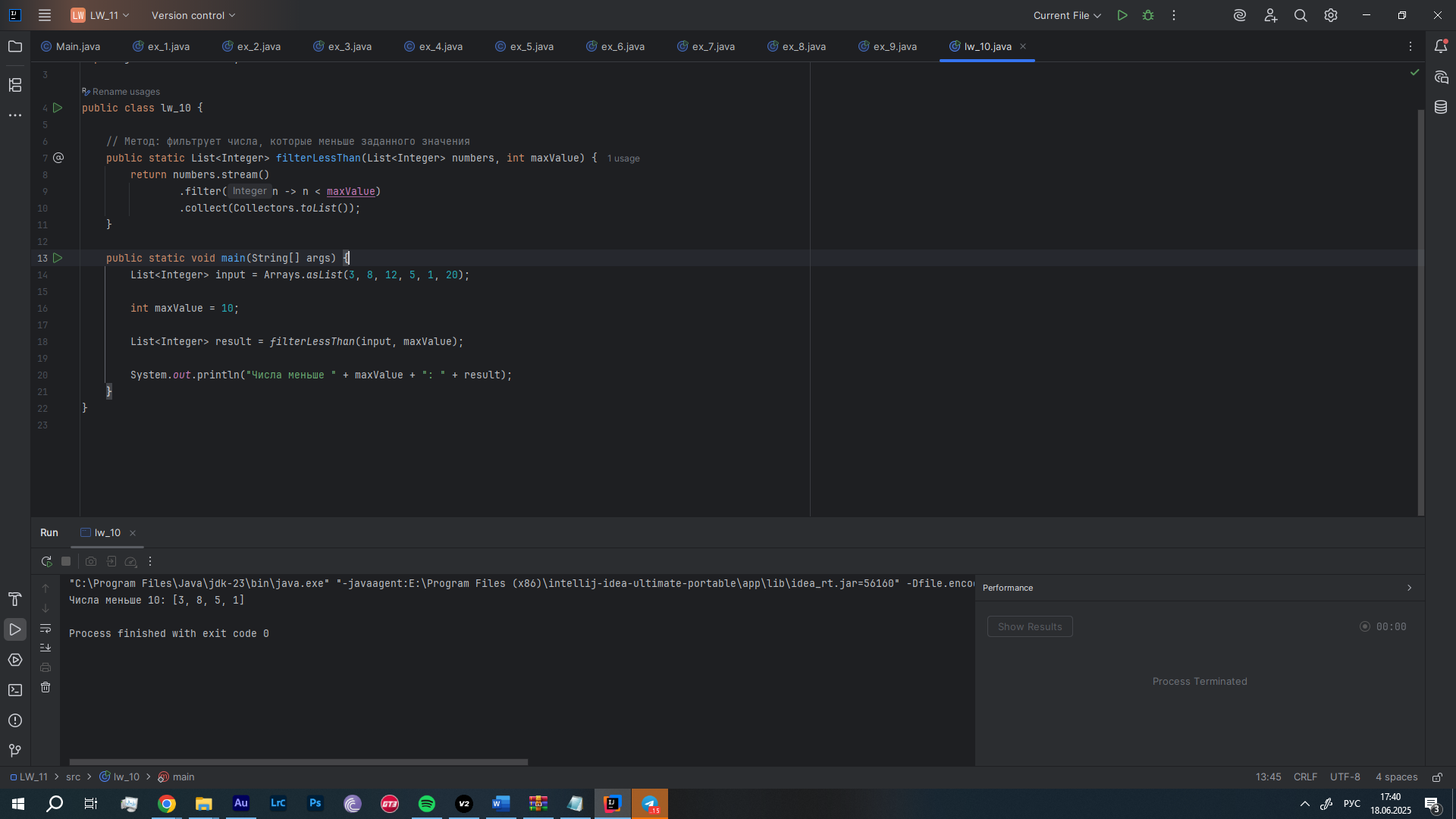


Рисунок 10 – Код и результат работы программы ex\_10

**Вывод**

В результате лабораторной работы были получены представление о функциональном программировании и Stream API.