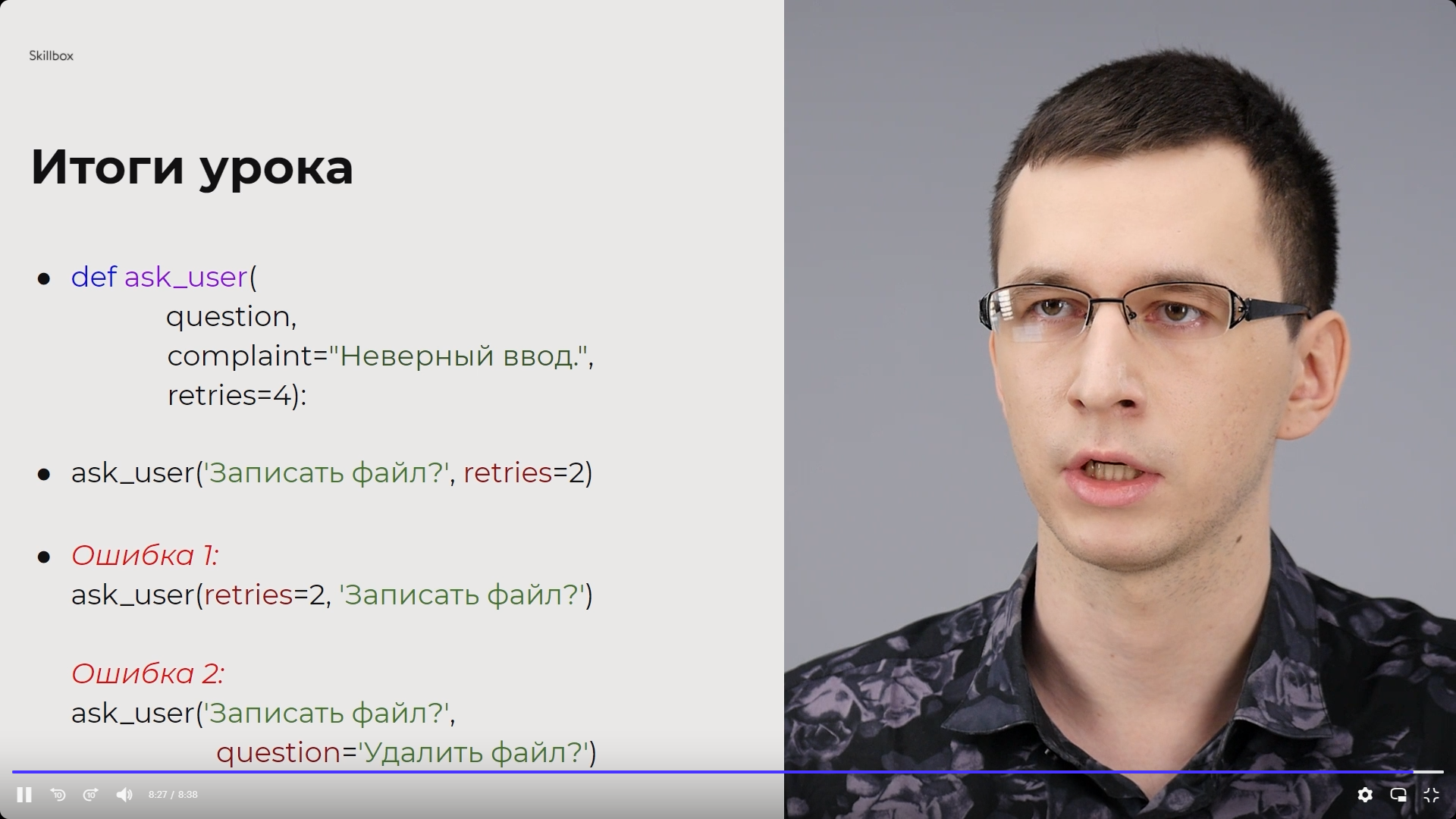
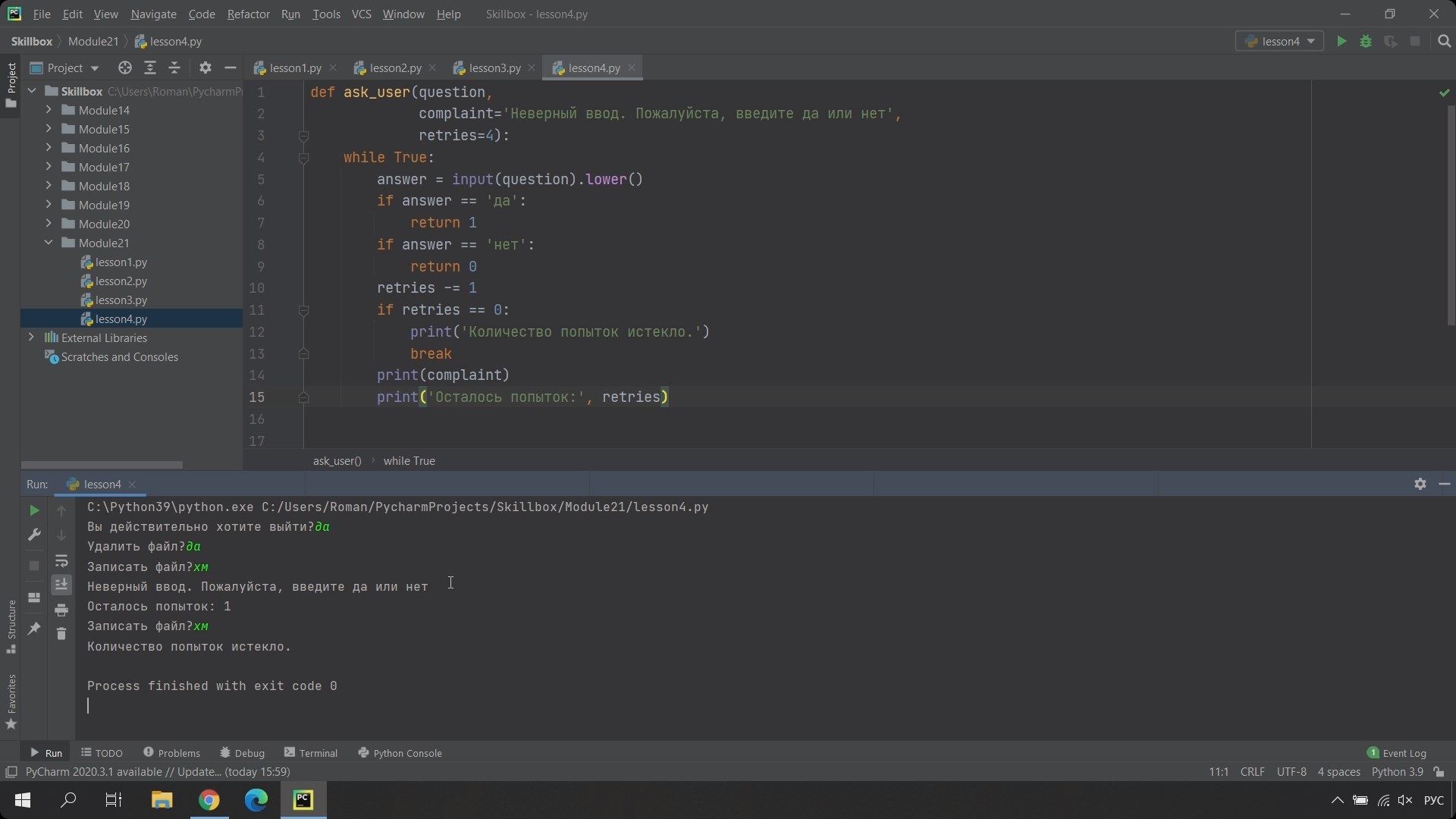
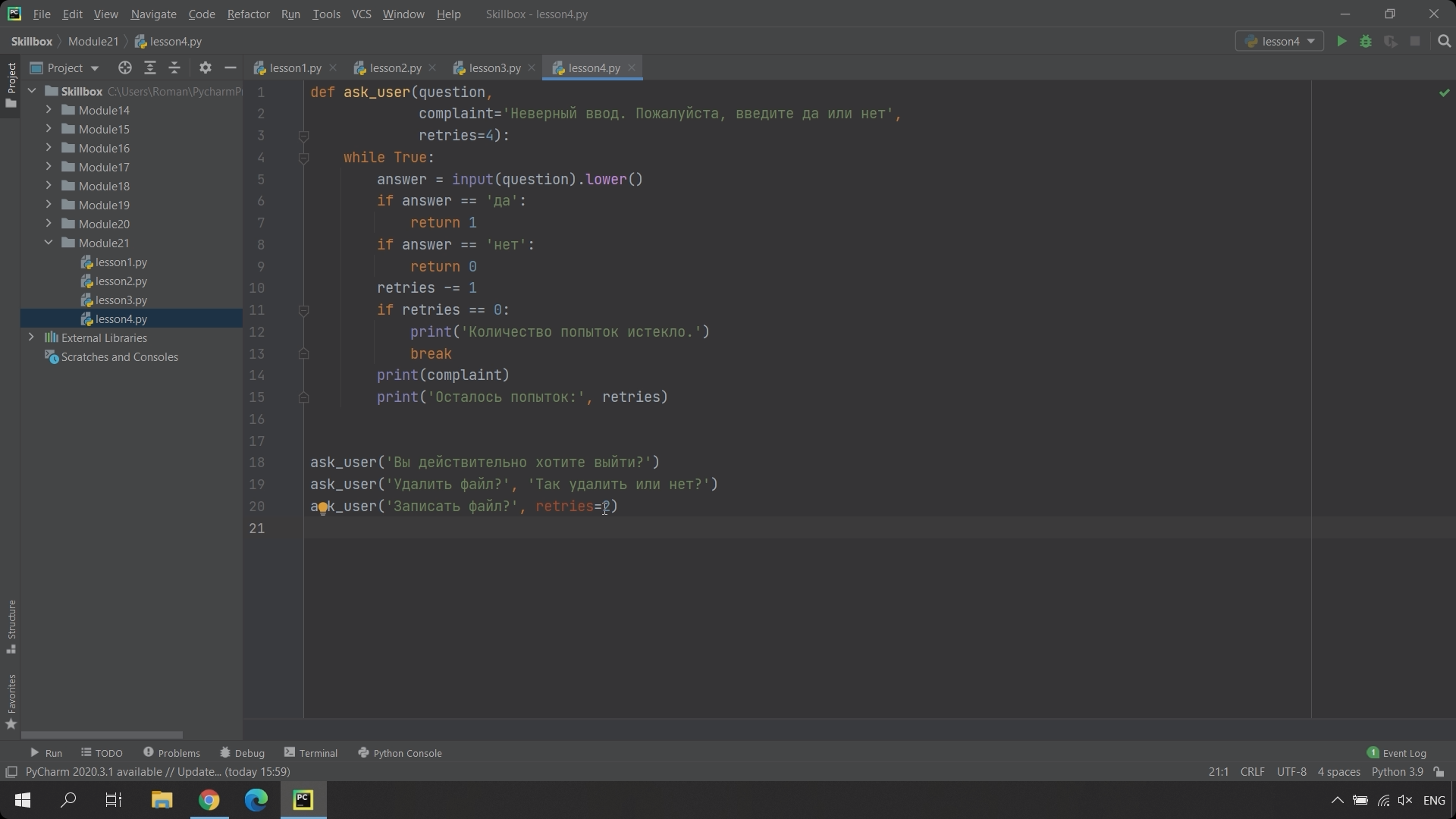
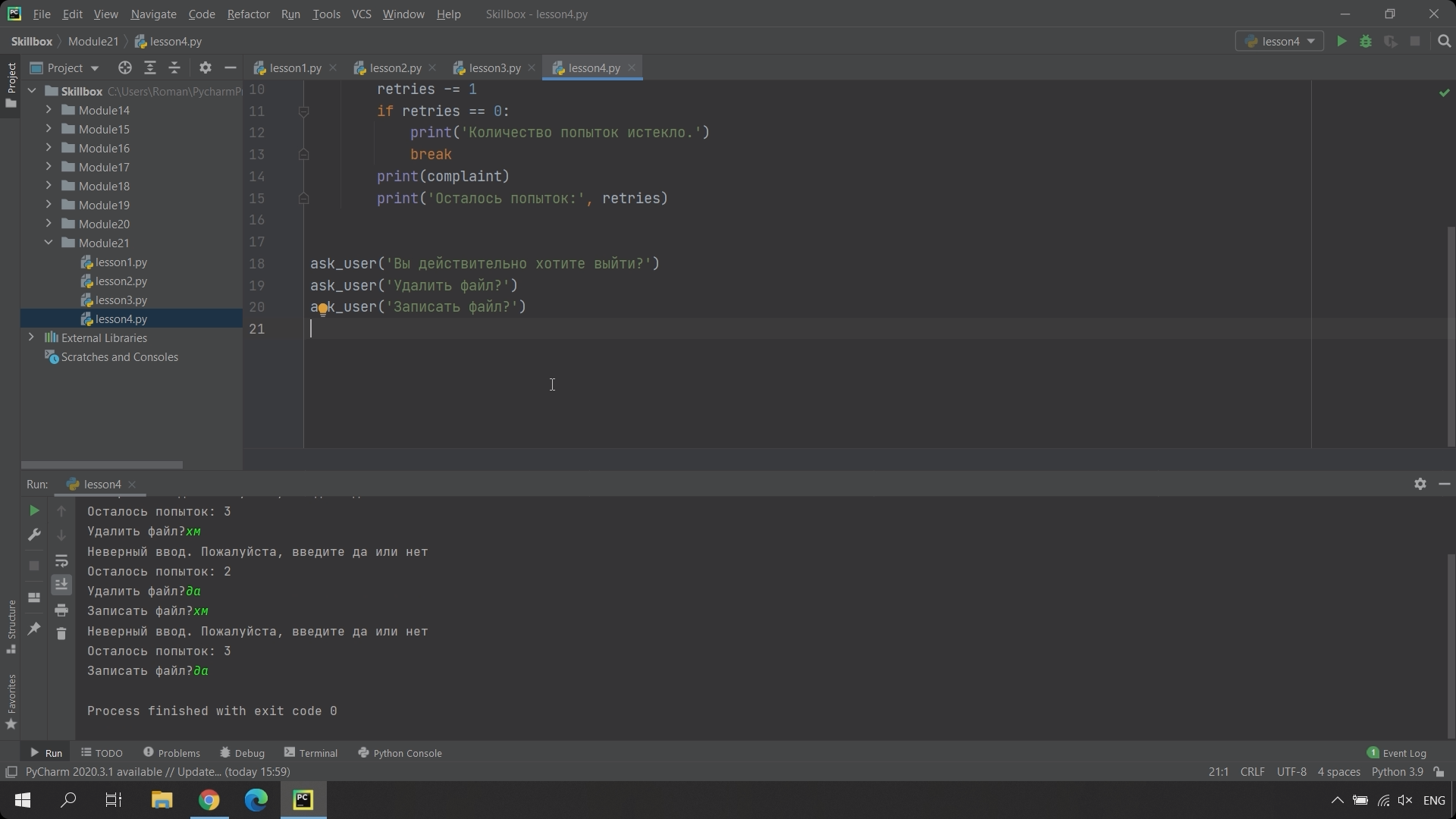
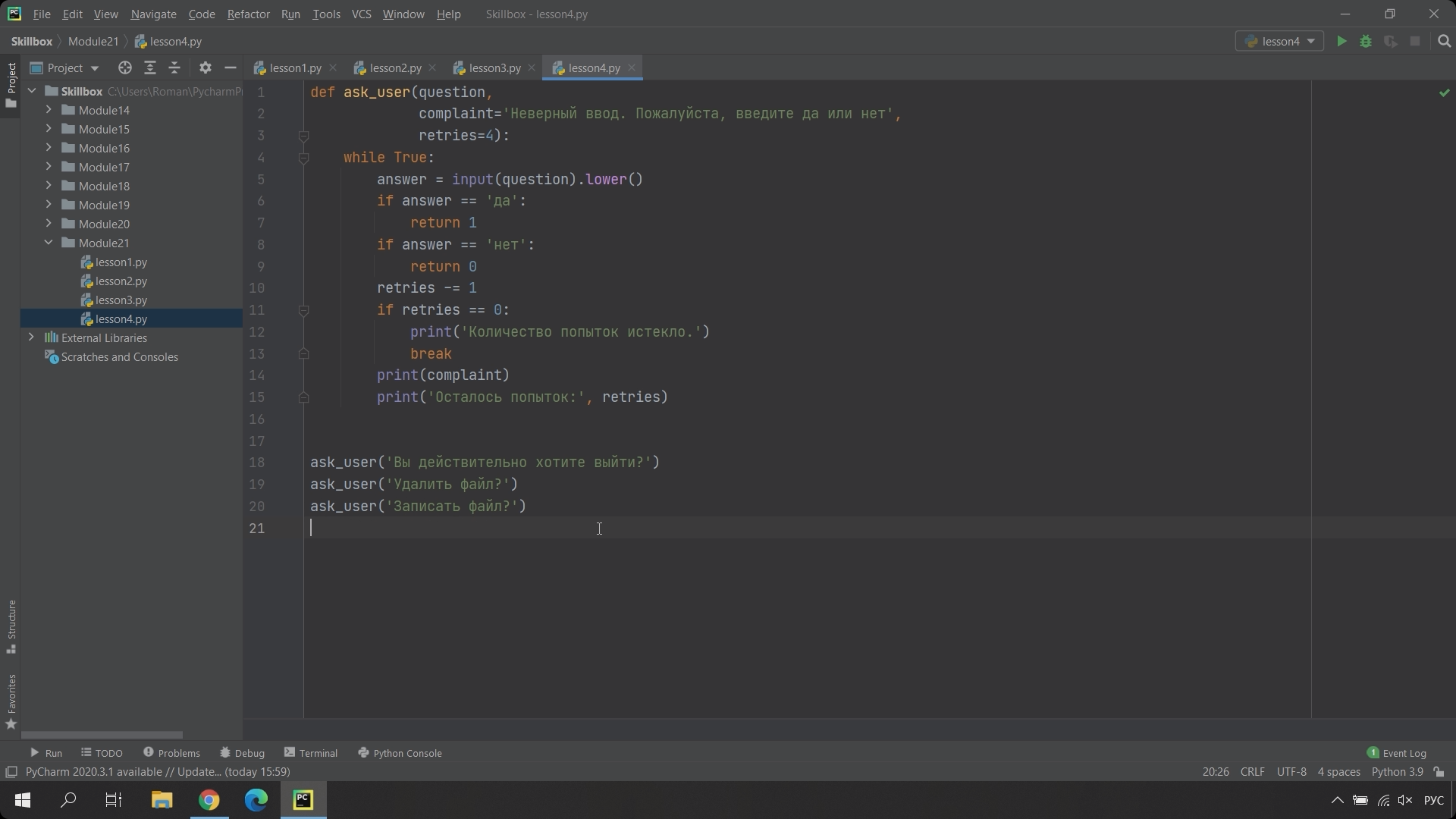
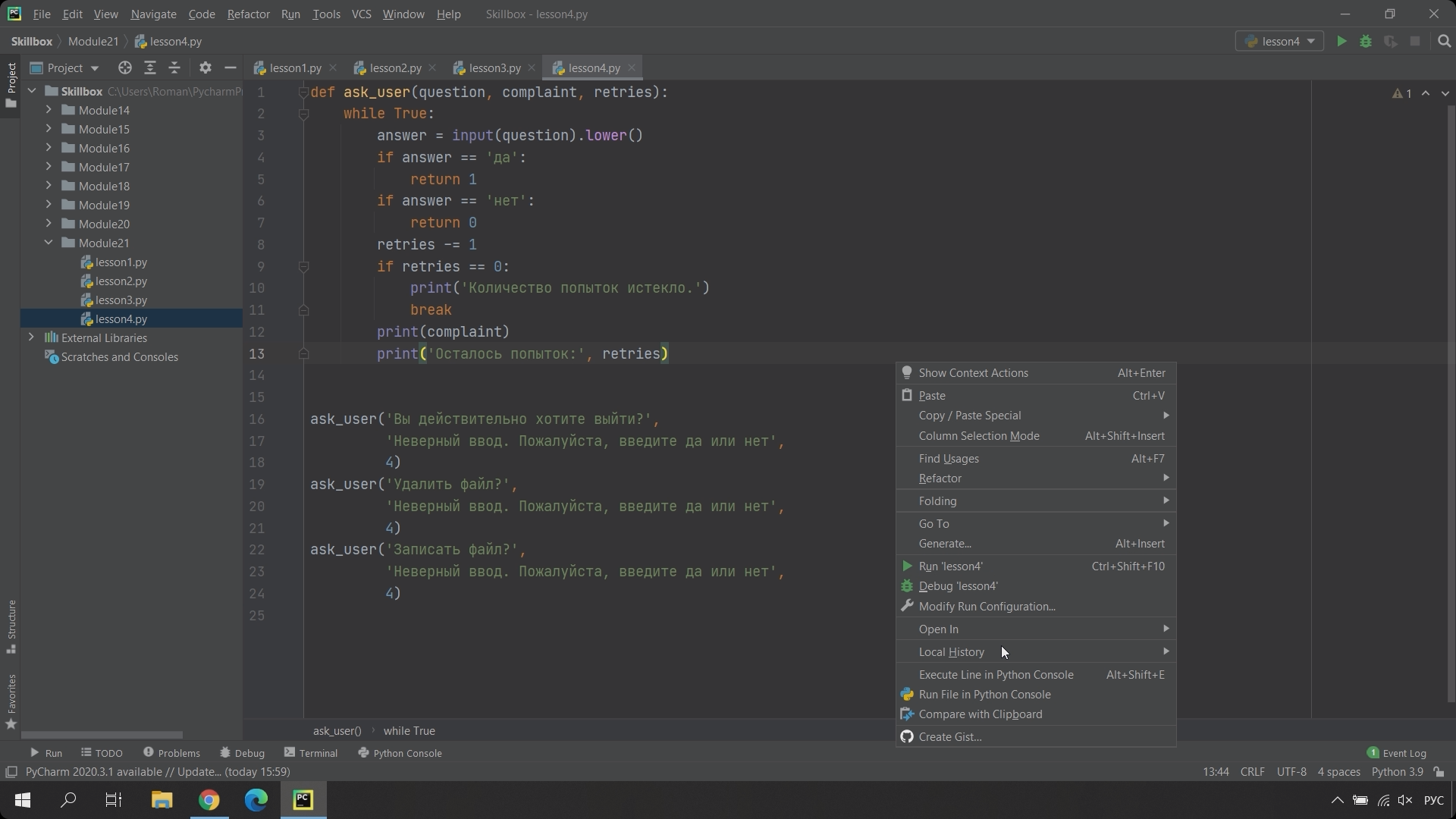
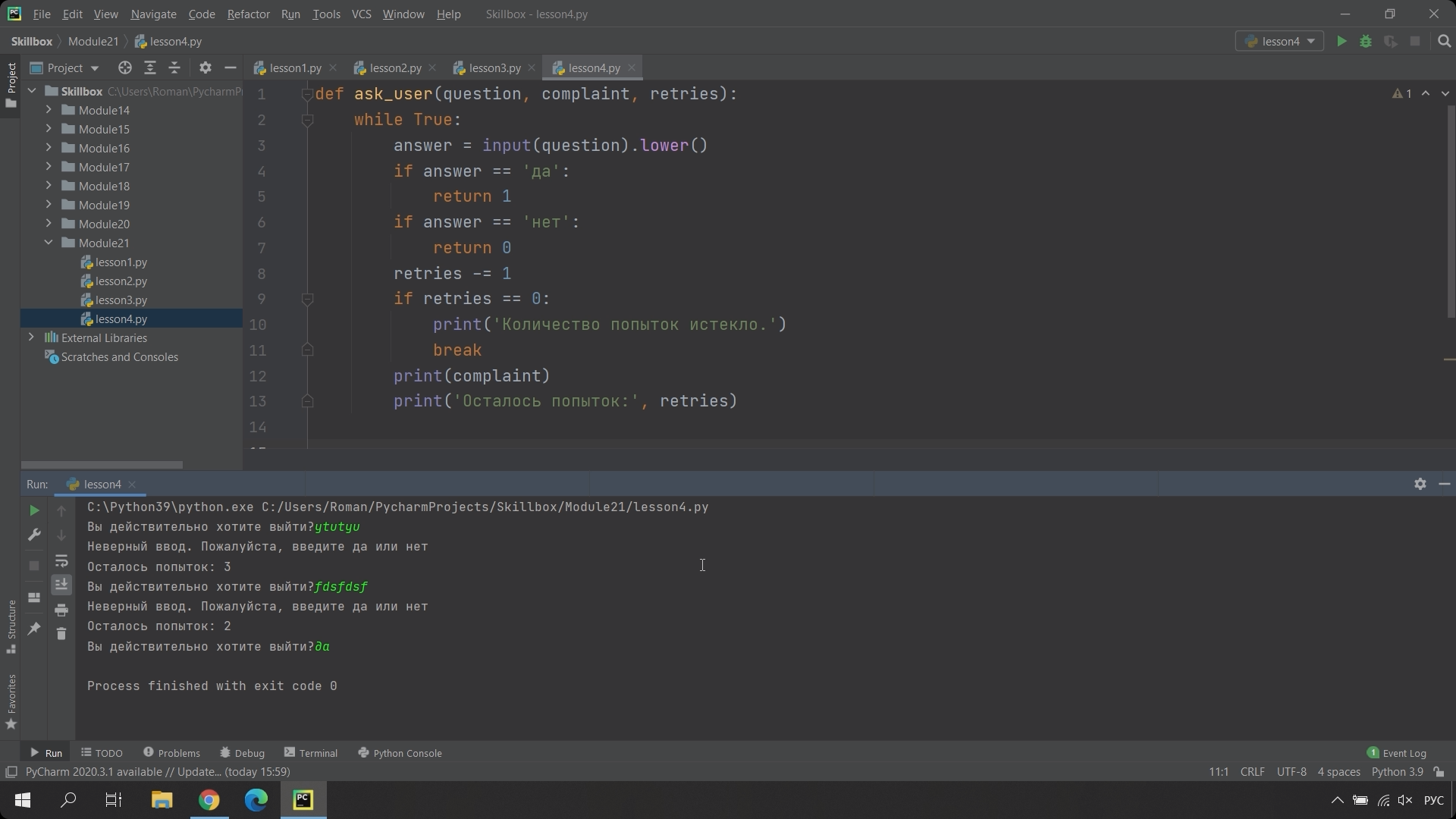
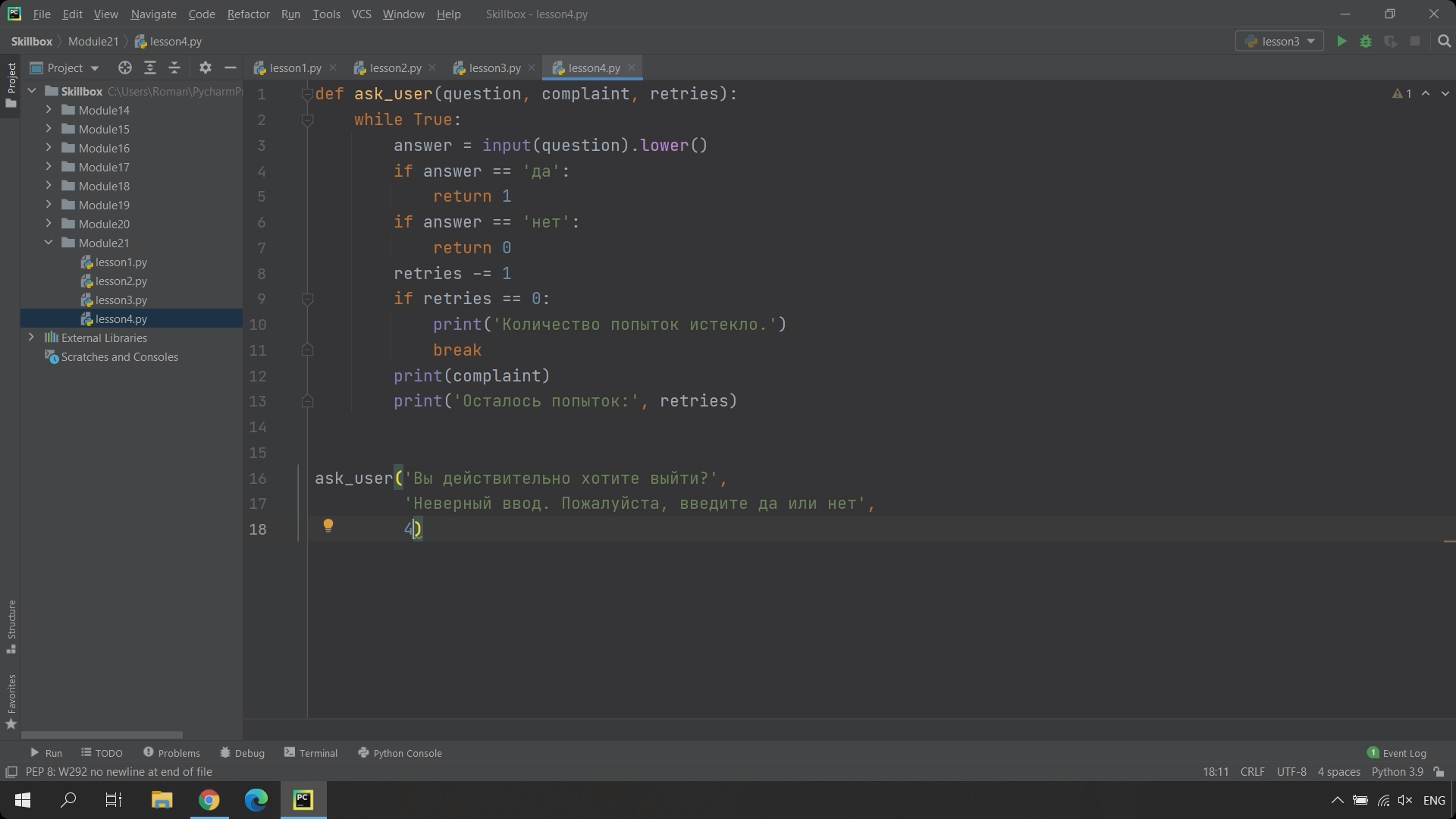
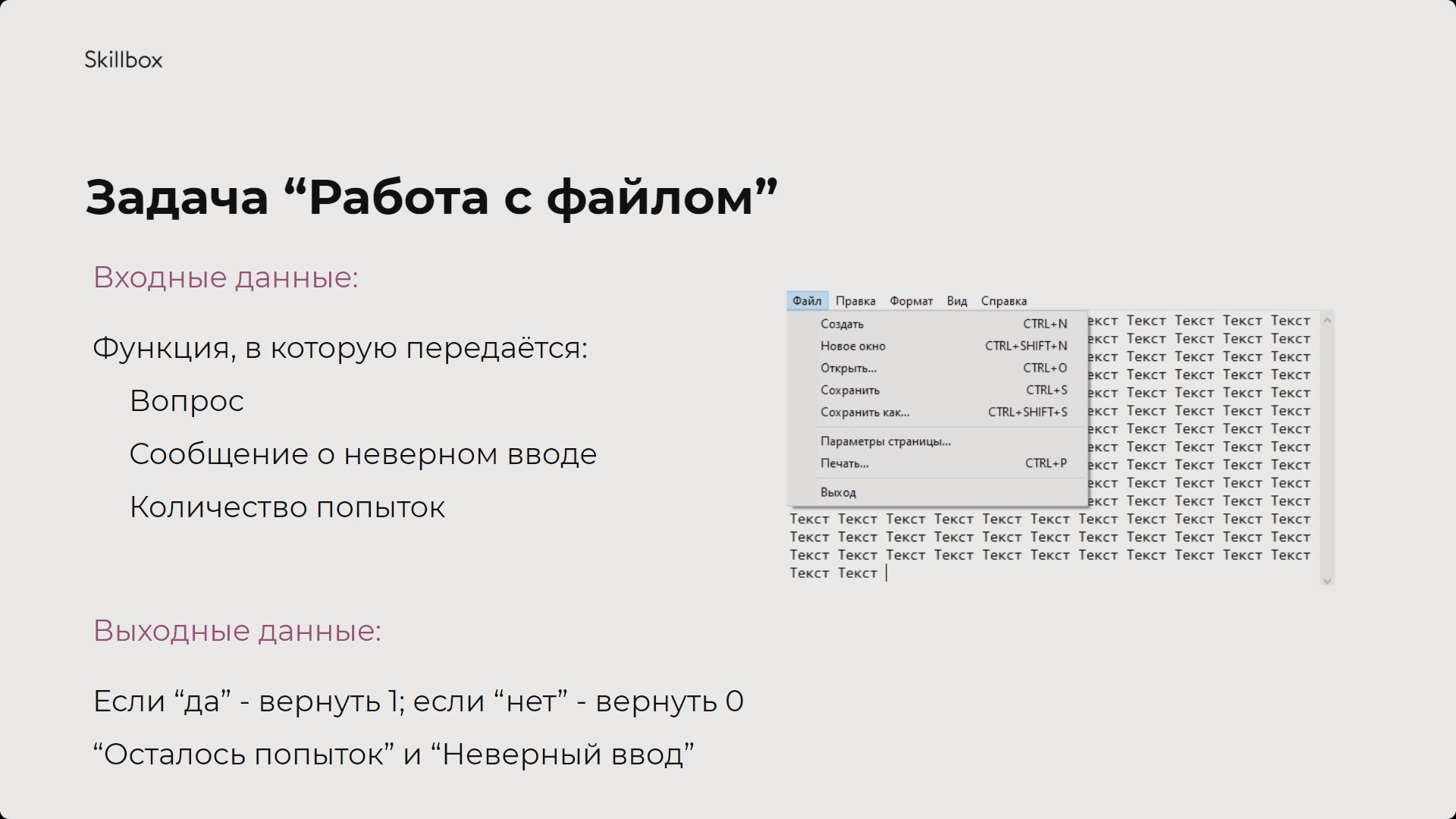
**21.6 Именованные аргументы и значения по умолчанию**



#### Практика

##### **Задача 1. Работа с файлом**

Вы пишете небольшое приложение для работы с файлами. Реализуйте функцию, которая может принимать на вход три аргумента: вопрос пользователю (на который нужно ответить да или нет), сообщение о неправильном вводе и количество попыток. Вопрос — обязательный позиционный аргумент, остальные — со значениями по умолчанию. При корректном ответе функция может возвращать что угодно — например, число 1 при ответе «да» или 0 при ответе «нет».

В основной программе вызовите функцию минимум три раза: только с вопросом, с вопросом и сообщением об ошибке, с вопросом и количеством попыток.

Пример работы программы:

Вы действительно хотите выйти? что

Неверный ввод. Пожалуйста, введите 'да' или 'нет'.

Осталось попыток: 3

Вы действительно хотите выйти? да

Удалить файл? не знаю

Так удалить или нет?

Осталось попыток: 3

Удалить файл? нет

Записать файл? ага

Неверный ввод. Пожалуйста, введите 'да' или 'нет'.

Осталось попыток: 1

Записать файл? да

##### **Задача 2. Накопление значений**

При работе со значениями по умолчанию и изменяемыми типами данных нужно знать и остерегаться ещё одной интересной штуки.

Напишите функцию с двумя аргументами: первый — число num, позиционный аргумент; второй — список lst, по умолчанию он пустой. В теле функции в список добавляется число num и сам список выводится на экран.

В основной программе вызовите функции три раза только с одним аргументом (числом), например так:

add\_num(5)

add\_num(10)

add\_num(15)

И посмотрите, что произойдёт.

После этого сделайте значение lst по умолчанию None и поправьте функцию, чтобы она работала правильно.

##### **Задача 3. Помощь другу**

Нашего друга попросили написать функцию, которая на вход принимает список всякого мусора. Ему нужно подготовить из этого списка список словарей, чтобы его коллеги смогли дальше продолжить обработку данных. Вот список правил, что нужно сделать с изначальным списком:

Если в списке встретился словарь, то оставляем его.

Если в списке встретилась строка, то из неё нужно сделать словарь и положить его в итоговый список, например  “abc” → {“abc”: “abc”}.

С числами нужно сделать то же самое, что и со строками.

Всё остальное выкидываем из нашего списка.

Друг написал программу, но в ней ошибка, так как она что-то не то выводит :( Нужна ваша помощь, вот сама программа:

def create\_dict(data, template=dict()):

    if isinstance(data, dict):

        return data

    if isinstance(data, int) or isinstance(data, float) or isinstance(data, str):

        return template[data] = data

def data\_preparation(old\_list):

    new\_list = []

    for i\_element in old\_list:

        new\_list.append(create\_dict(i\_element))

    return new\_list

data = [“sad”, {“sds”: 23}, {43}, [12, 42, 1], 2323]

data = data\_preparation(data)

print(data)

Исправьте программу и убедитесь, что всё работает верно.