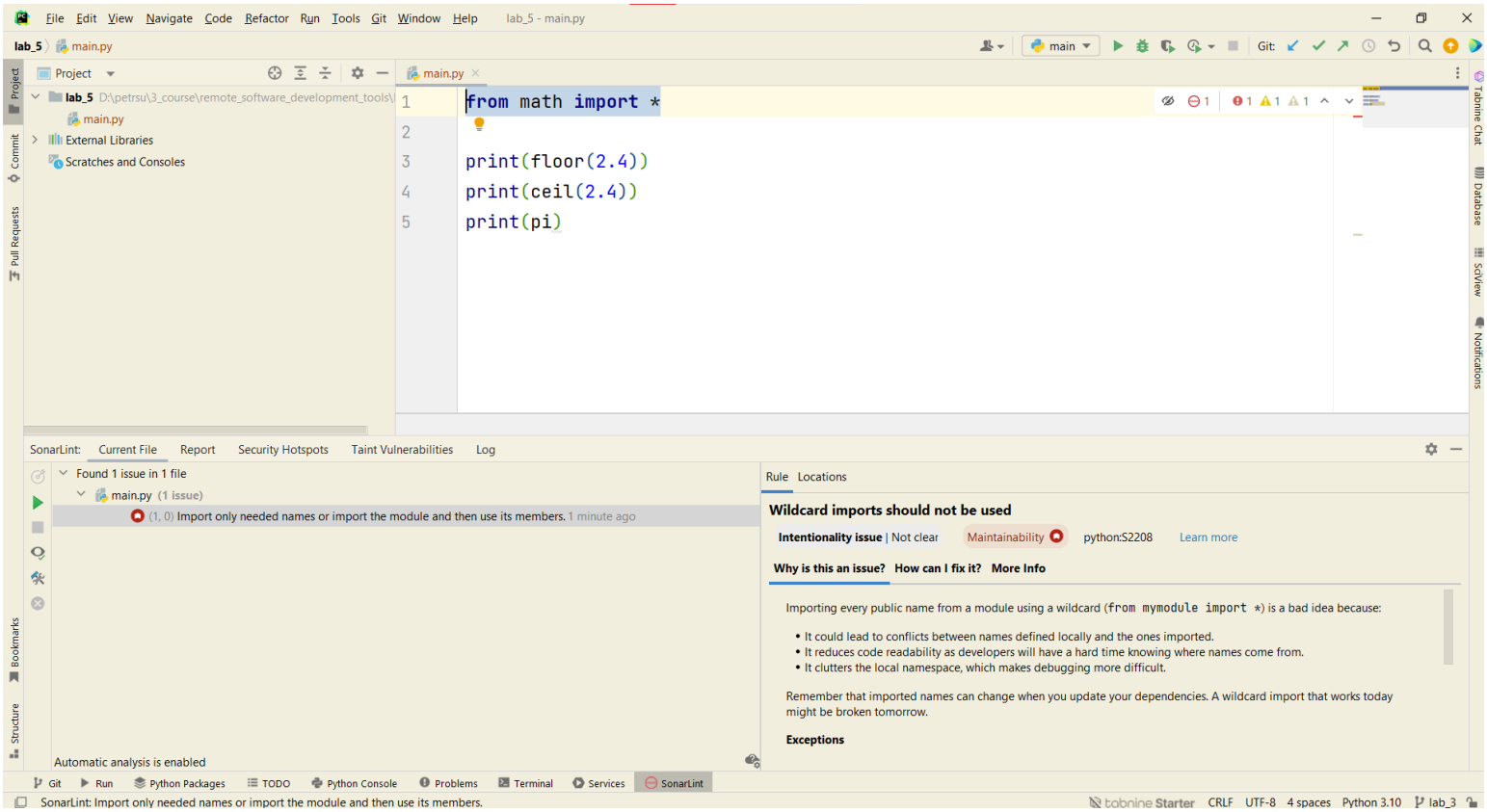
Использование import *

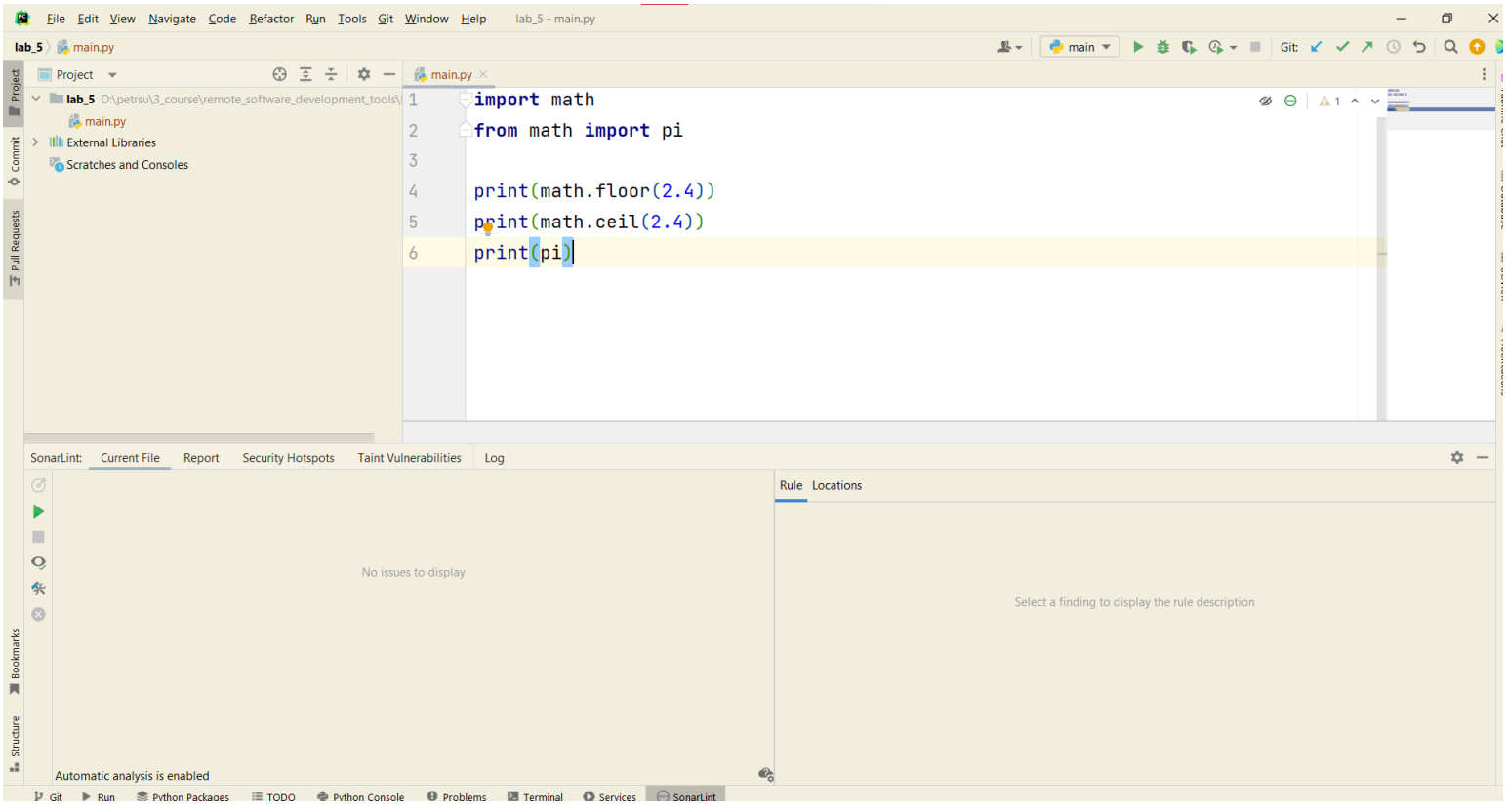
У ленивого разработчика часто возникает соблазн импортировать все из модуля, используя команду `from xyz import *`.

Это не очень хорошая практика по многим причинам. Вот лишь некоторые из них.

- Это может быть неэффективно: если модуль содержит большое количество объектов, придется долго ждать, пока все будет импортировано.
- Это может привести к конфликту между именами переменных: используя `*`, нельзя предугадать, какие объекты будут импортированы и какие у них будут имена.

Как с этим бороться? Импортируйте либо конкретный объект, который планируете использовать, либо весь модуль.

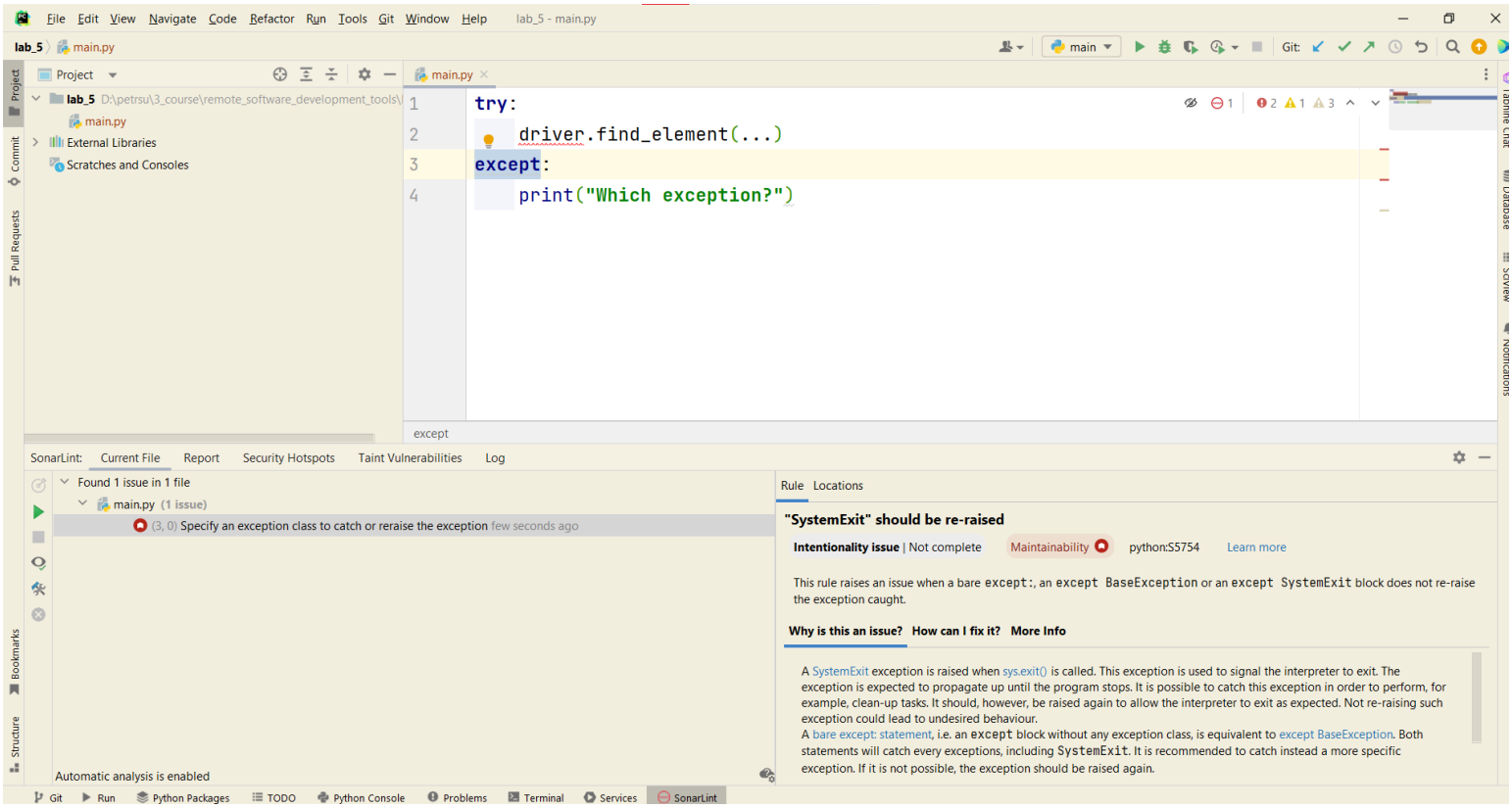


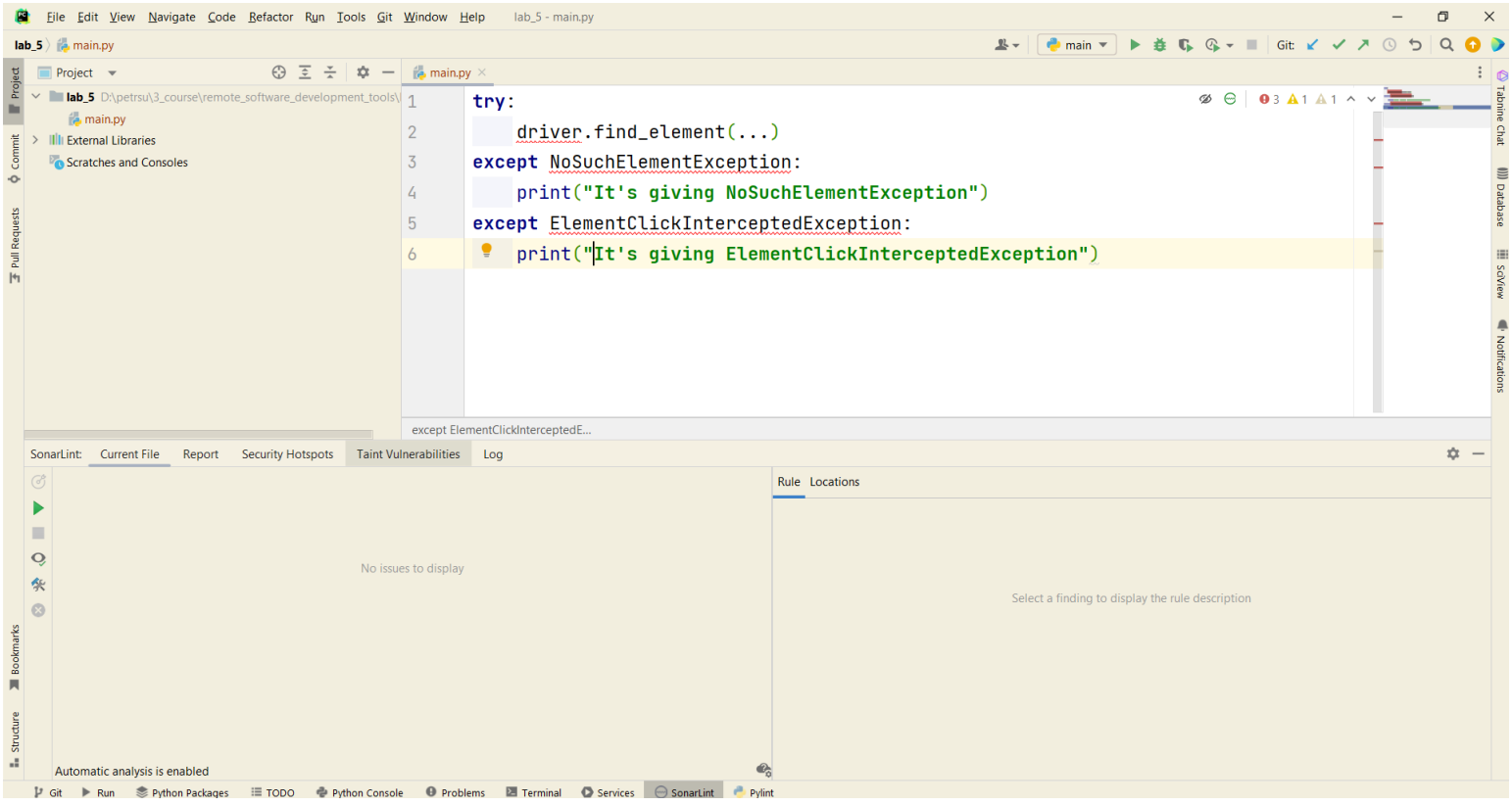


Использование try/except без определения типа исключения в блоке except

В результате использования чистого `except` будут перехватываться исключения SystemExit и KeyboardInterrupt, что затруднит прерывание программы с помощью Control-C.

Чтобы избежать этой проблемы, при использовании `try/except`, определяйте тип исключения в блоке `except`.

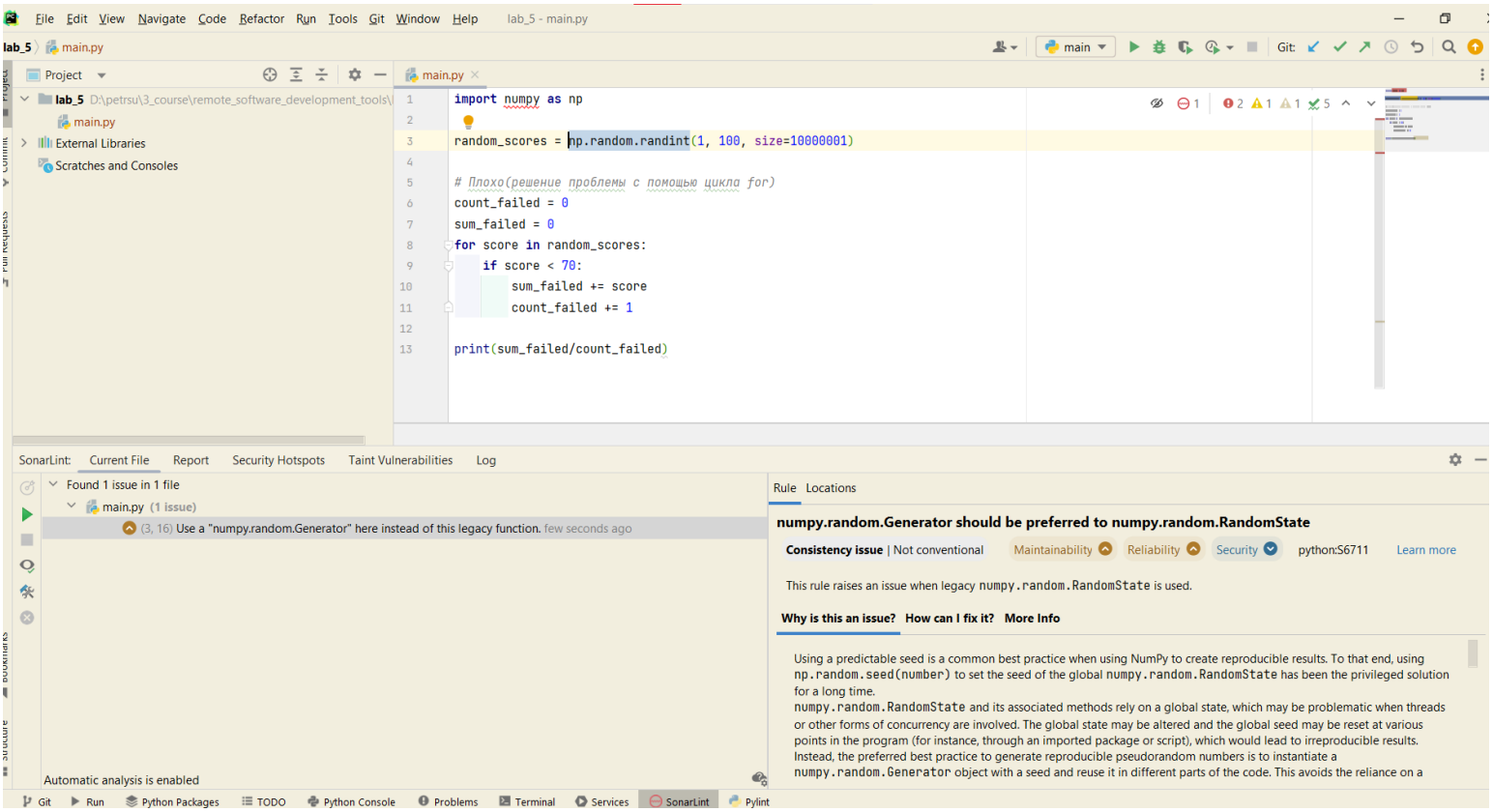


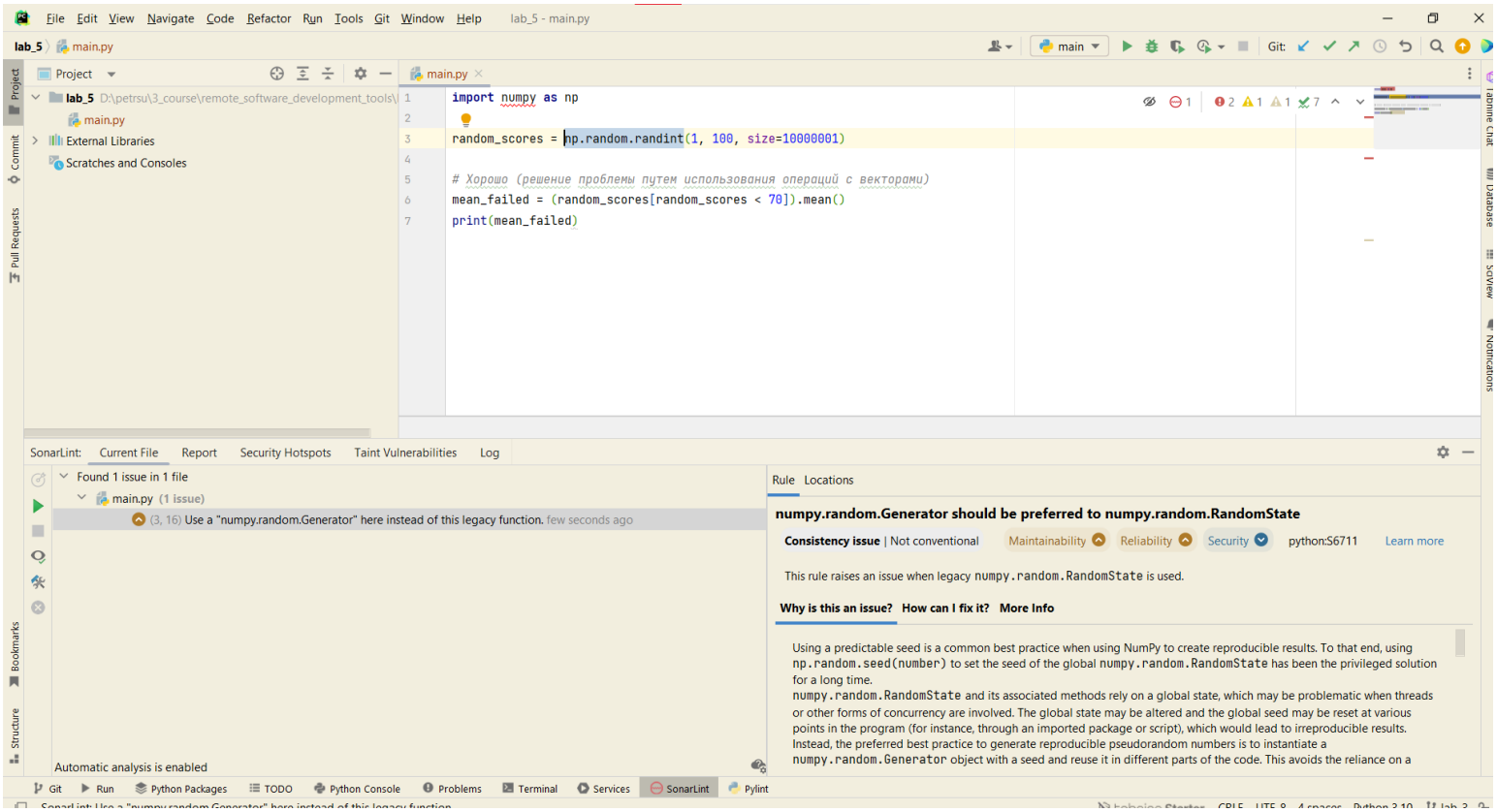


Пренебрежение Numpy при математических вычислениях

Многие разработчики забывают о существовании пакетов, которые могут сделать работу в Python проще и продуктивнее.

Одним из таких пакетов, который следует использовать при математических вычислениях, является Numpy. Он помогает решать математические операции быстрее, чем циклы `for`.



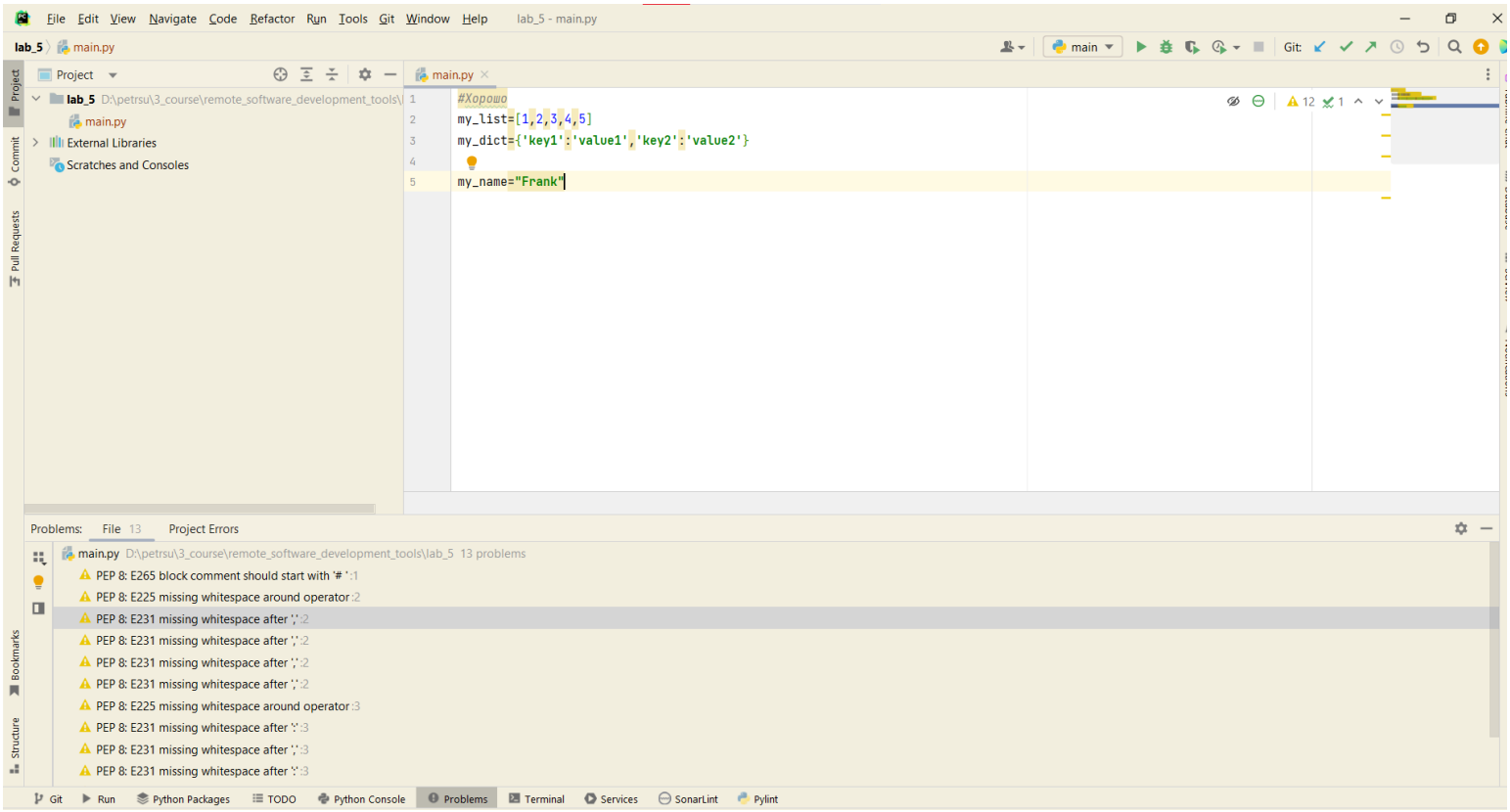


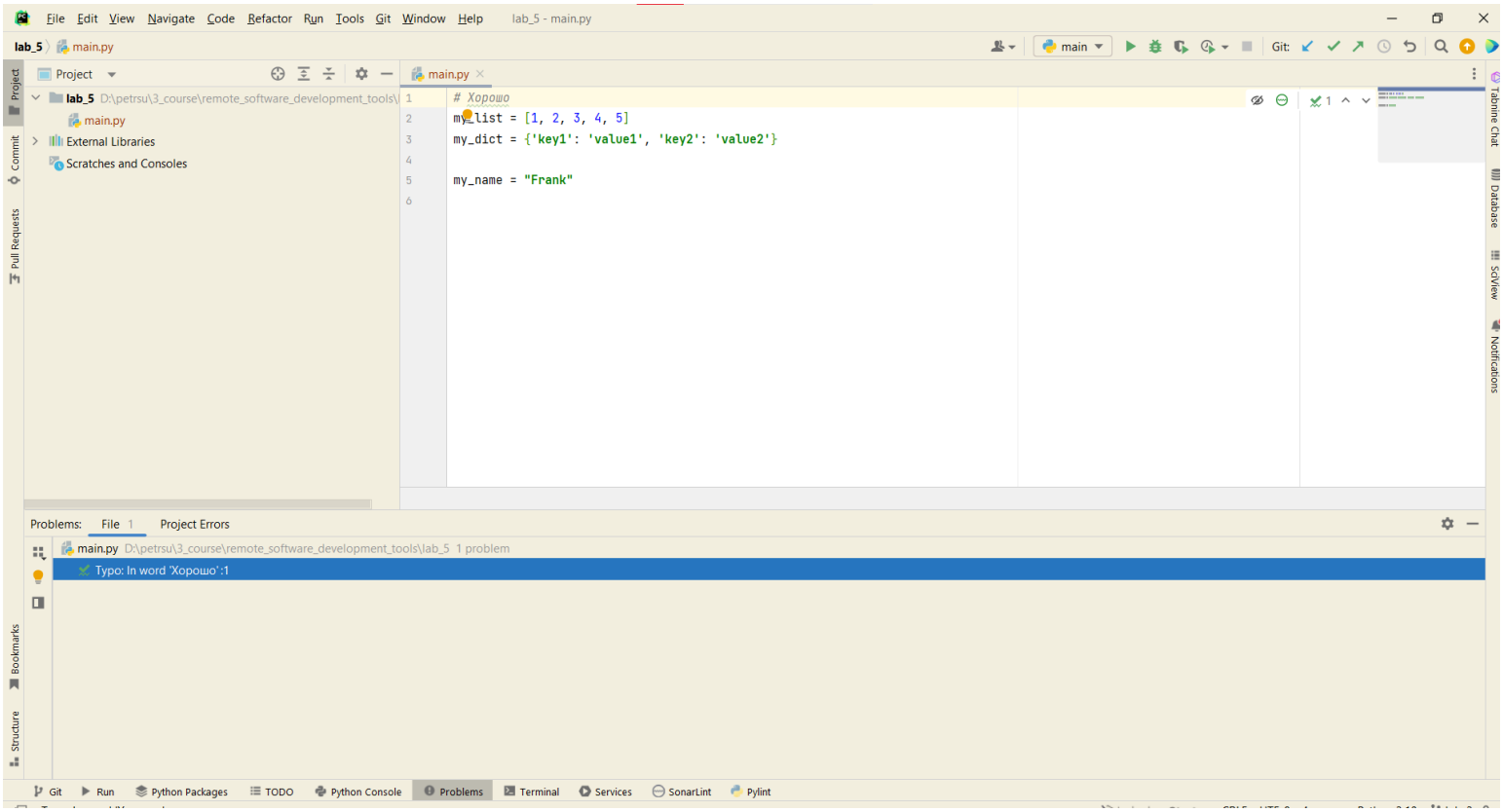
Несоблюдение PEP8

PEP8 — это документ, который должен прочитать каждый программист, изучающий Python. В нем содержатся рекомендации лучшие практики по написанию кода на этом языке (некоторые рекомендации в данной статье взяты из PEP8).

Допустим, вы используете Pycharm. Если напишете код, который не соответствует правилам PEP8, увидите волнистые подчеркивания, как на изображении ниже.

В моем случае нужно было только добавить пробелы после `,` и `:`.



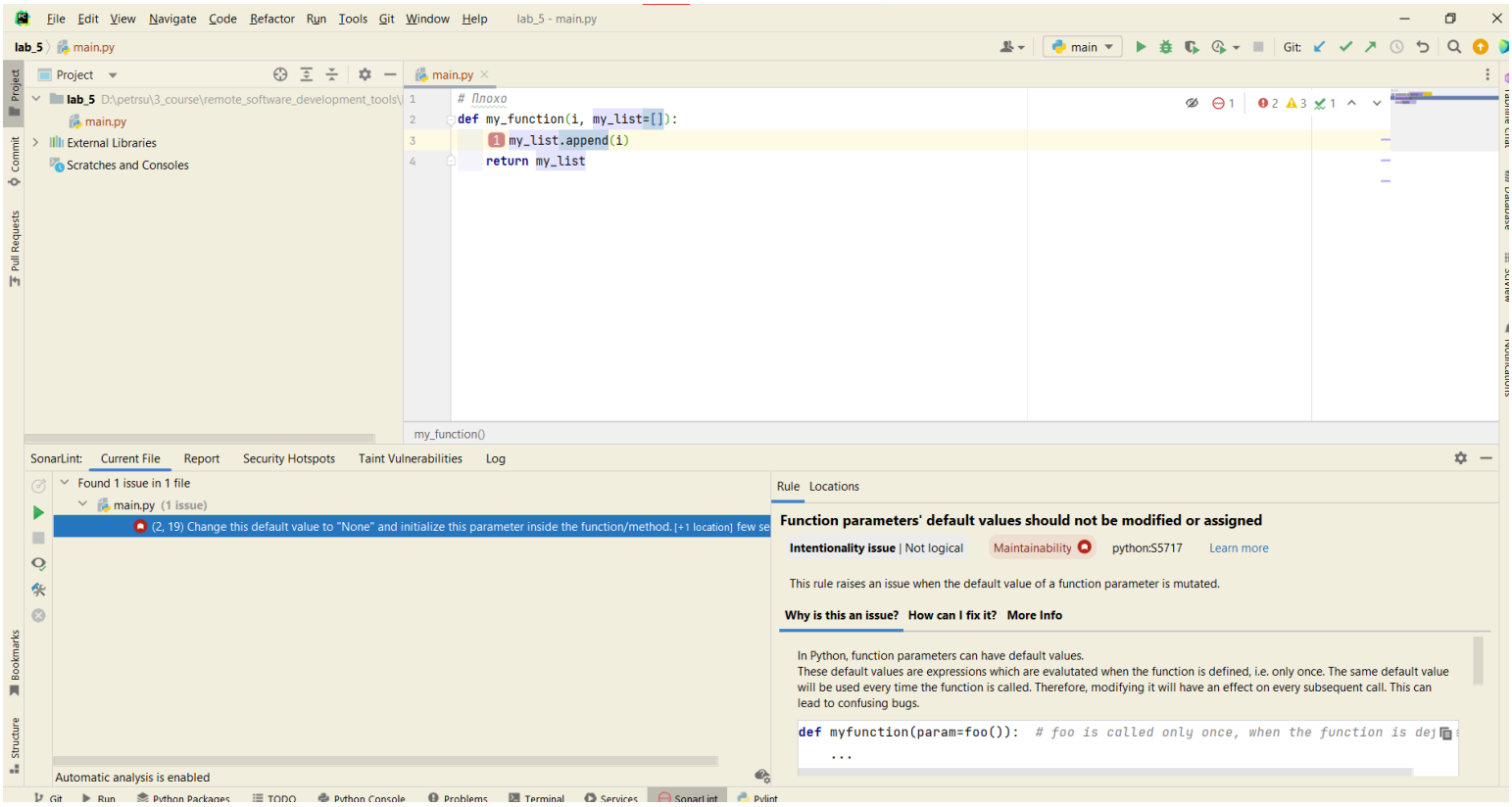


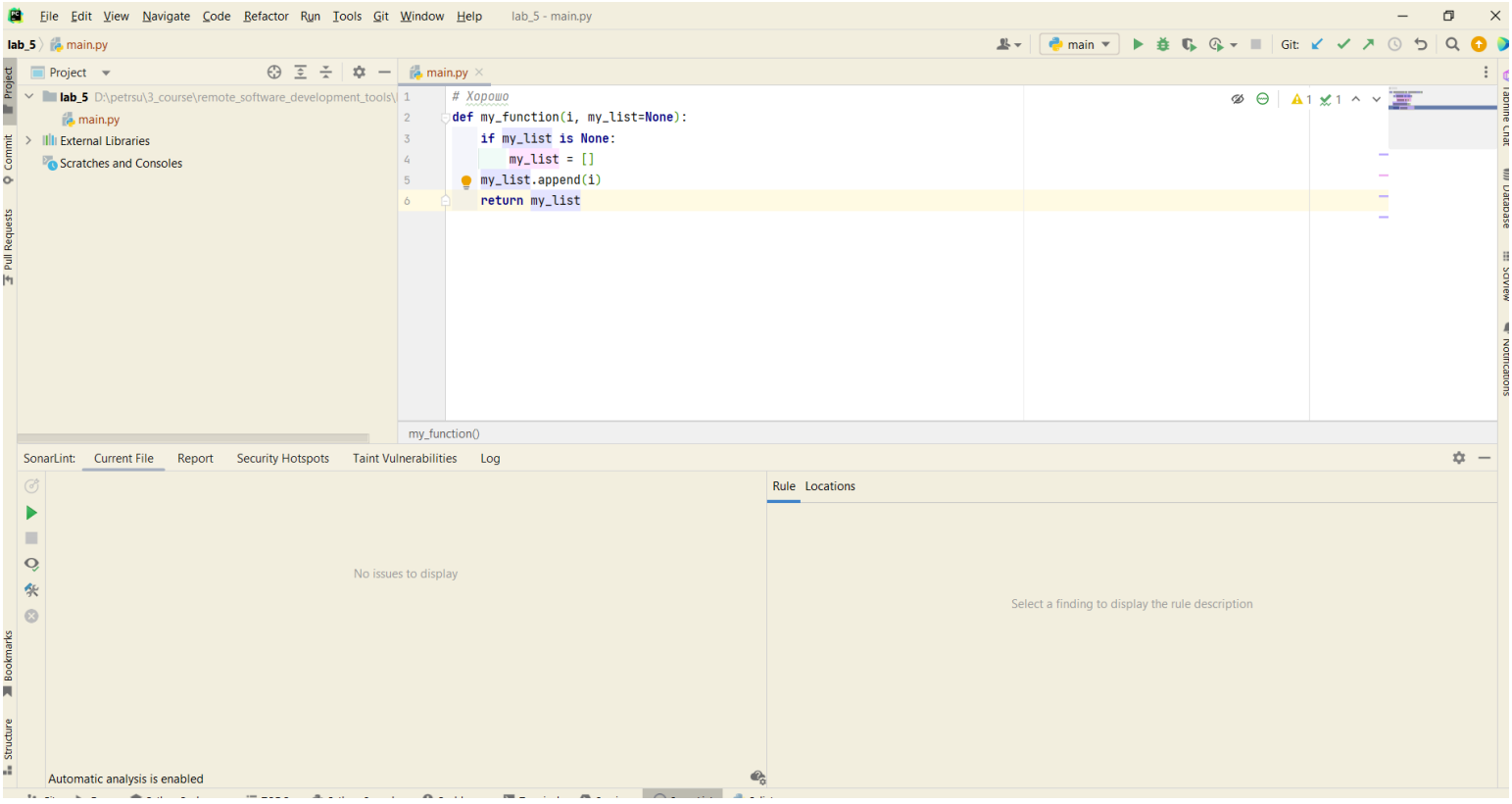
Использование изменяемых значений по умолчанию

Включение в функцию в качестве параметра по умолчанию изменяемого значения (например, списка) может привести к неожиданному поведению кода.

В приведенном выше коде каждый раз при вызове функции `my_function` параметр `my_list` будет сохранять значения из предыдущих вызовов (скорее всего, будет инициироваться пустой список при каждом вызове функции).

Чтобы избежать такого поведения, нужно установить параметр `my_list` равный None и включить блок `if` ниже:

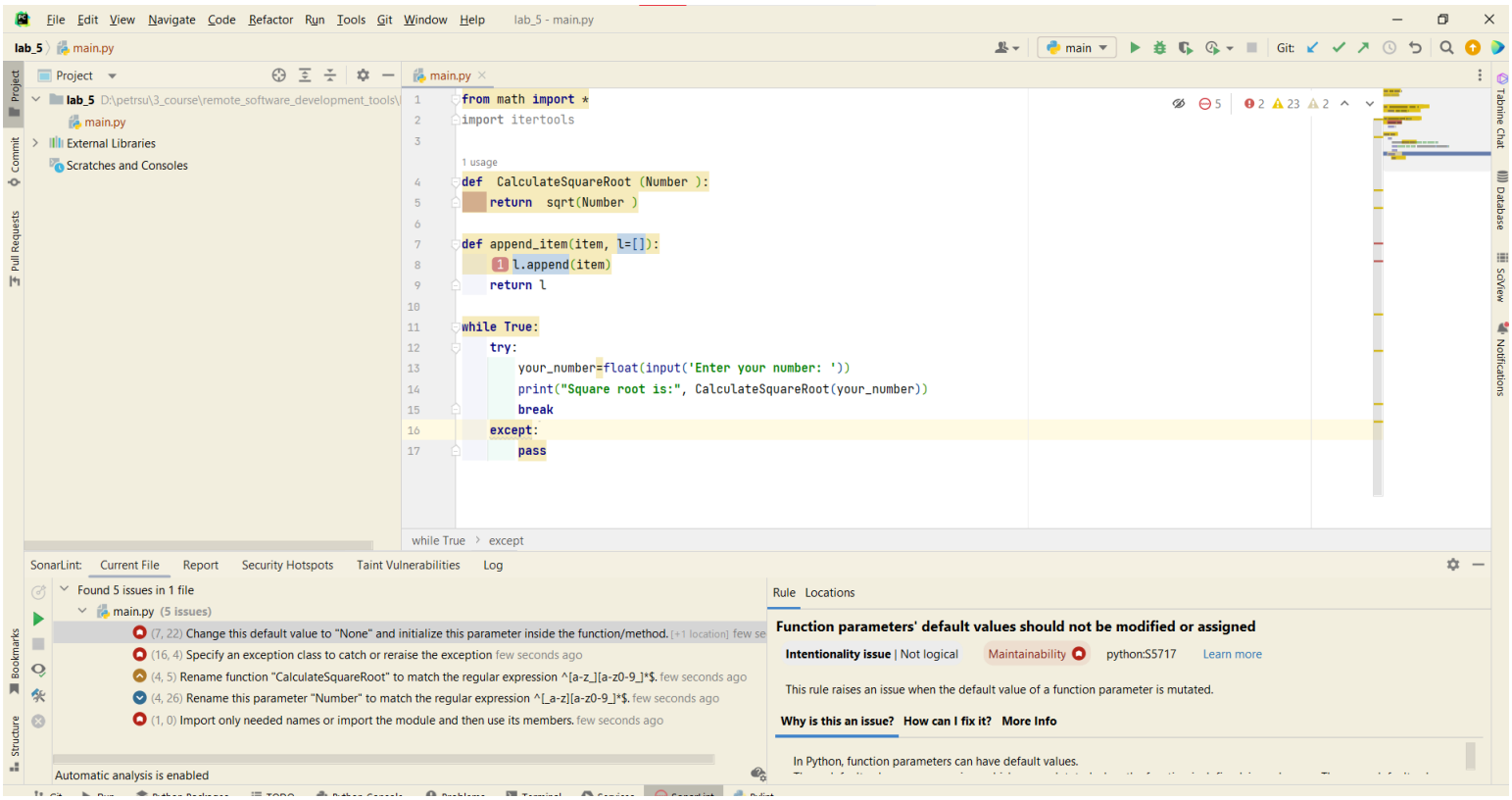


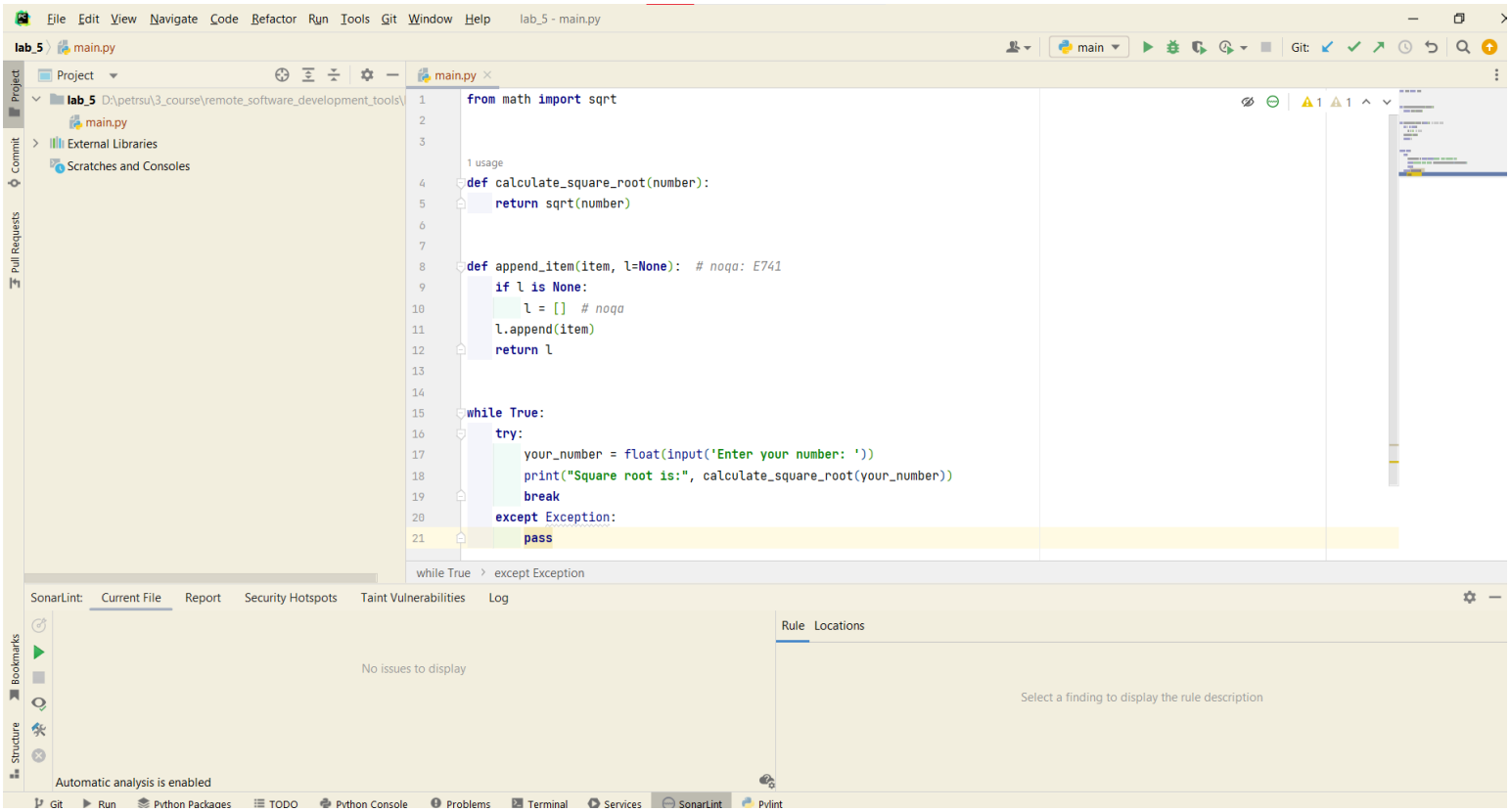


Программа с как можно большим количеством “плохих практик”

Возможно, вам не видно всего, но в этом коде точно есть следующие "запахи кода":

- `import *` — импортирование всех имен из модуля, хотя используется из них только одно;
- `import itertools` — ненужный импорт;
- во множестве мест стоят лишние или отсутствующие пробелы;
- название функции написано в стиле PascalCase;
- в некоторых местах используются табы для отступов;
- используется список (изменяемый объект) в качестве значения аргумента функции по умолчанию;
- используется слишком “широкое” выражение `except:` без указания конкретного исключения.





Материалы

- https://thecleverprogrammer.com/2020/12/27/music-player-gui-with-python/
- https://medium.com/nuances-of-programming/10-ошибок-которые-выдают-новичков-в-python-ea5356dcde0a
- https://semakin.dev/2020/05/python_linters/

Плагины

- https://www.geeksforgeeks.org/5-best-pycharm-plugins-for-development/
- https://habr.com/ru/articles/687482/
- https://duckly.com/blog/best-plugins-for-pycharm-2022/
- https://design-hero.ru/articles/155761/