|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Теоретическая информатика и компьютерные технологии

**Отчет по лабораторной работе № 4**

***по курсу «Алгоритмы биоинфоматики»***

**«Алгоритм Нидлмана-Вунша с аффинным штрафом»**

Студент ИУ9-71Б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  Максимов Е.Н.

(Группа) (Подпись, дата) (Фамилия И.О.)

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Страшнов П. В. (Подпись, дата) (Фамилия И.О.)

*Москва, 2022 г.*

# Цель работы

Реализовать алгоритм Нидлмана-Вунша с аффинным штрафом за пропуски.

# Постановка задачи

Выполнить выравнивание двух аминокислотных или нуклеотидных последовательностей.

# Теоретическая часть

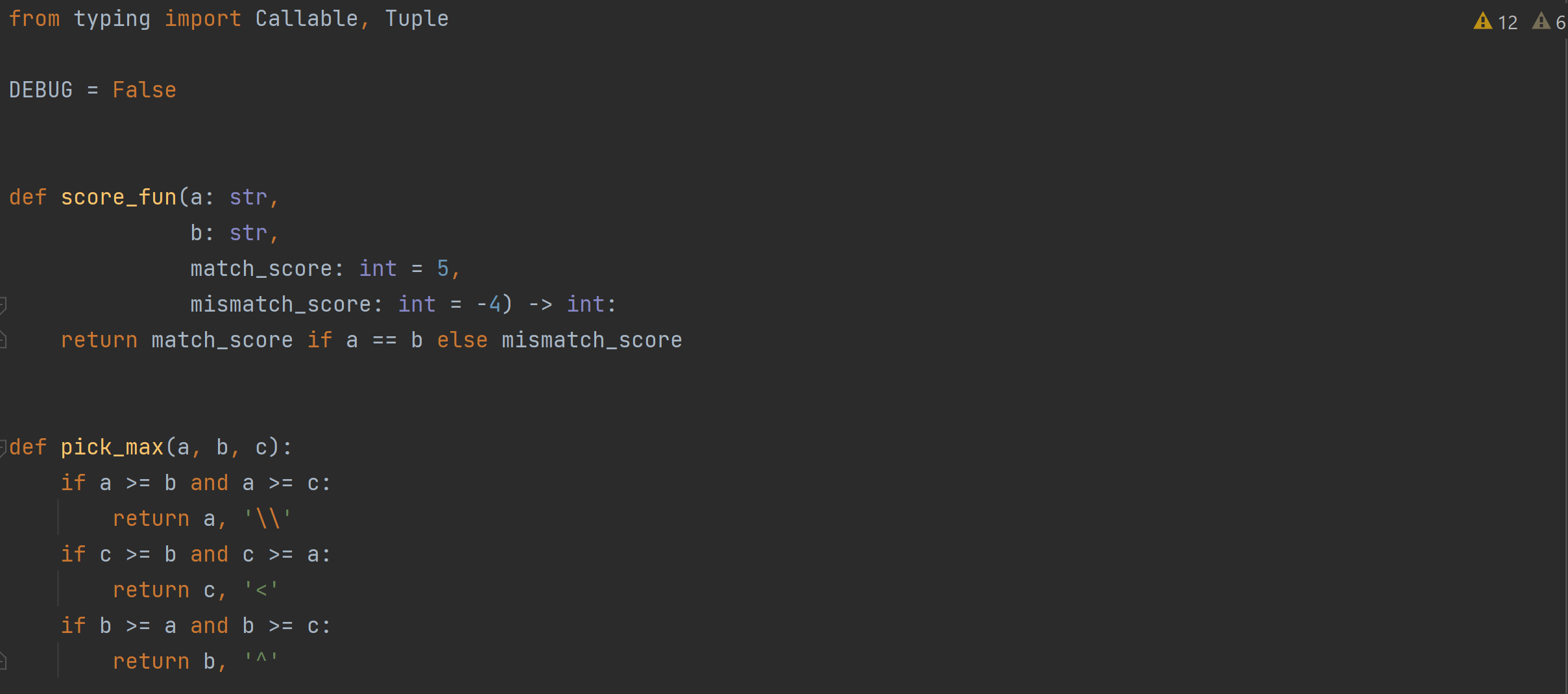
Алгоритм является аналогичным алгоритму Нидлмана-Вунша изменение заключается в подсчёте гэпа: , где *open* – штраф за открытие (первый гэп), *length* – число последовательных гэпов в одной строке, *extend* – штраф за продолжение гэпа.

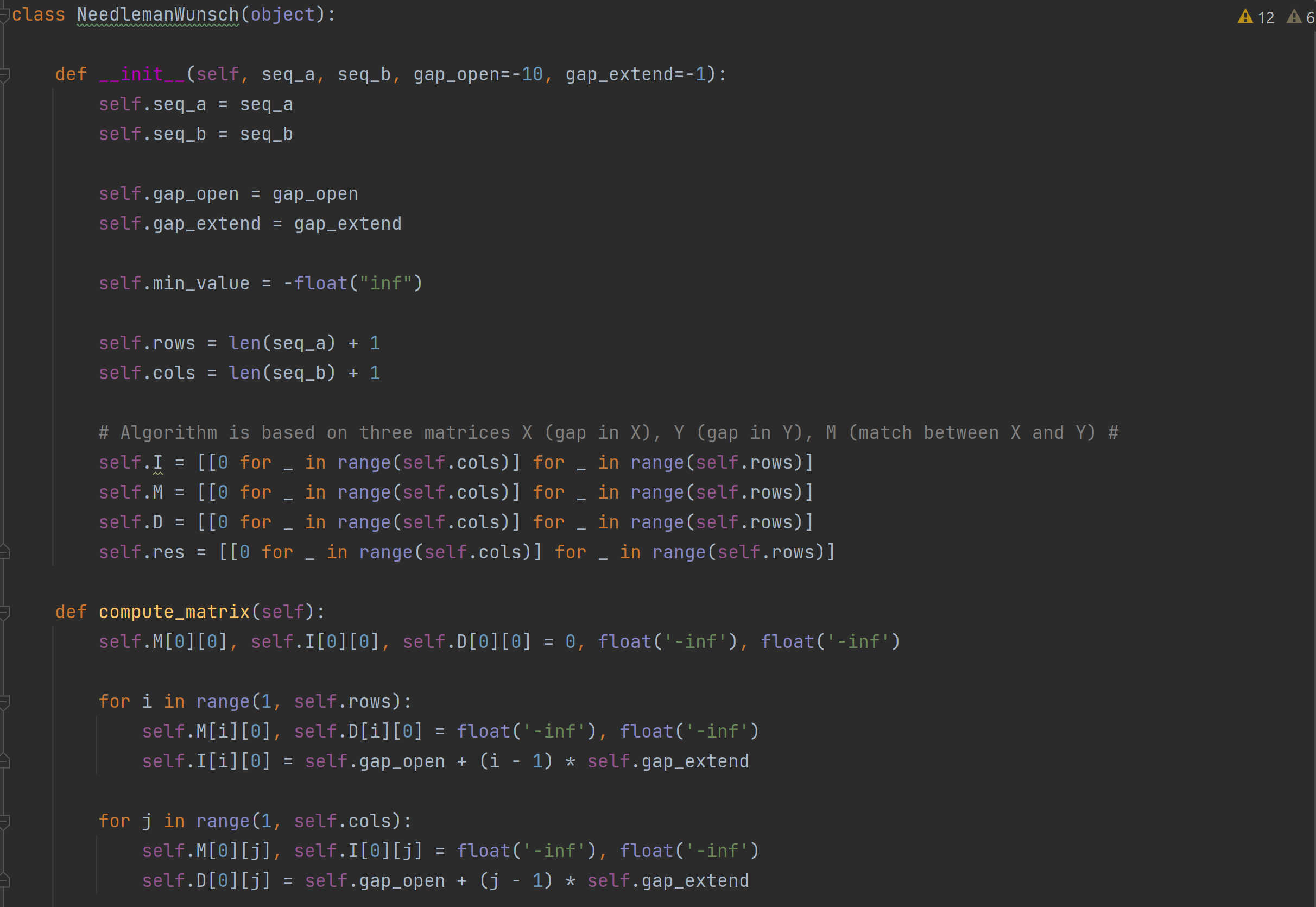
Также для реализации используются 3 матрицы вместо одной:

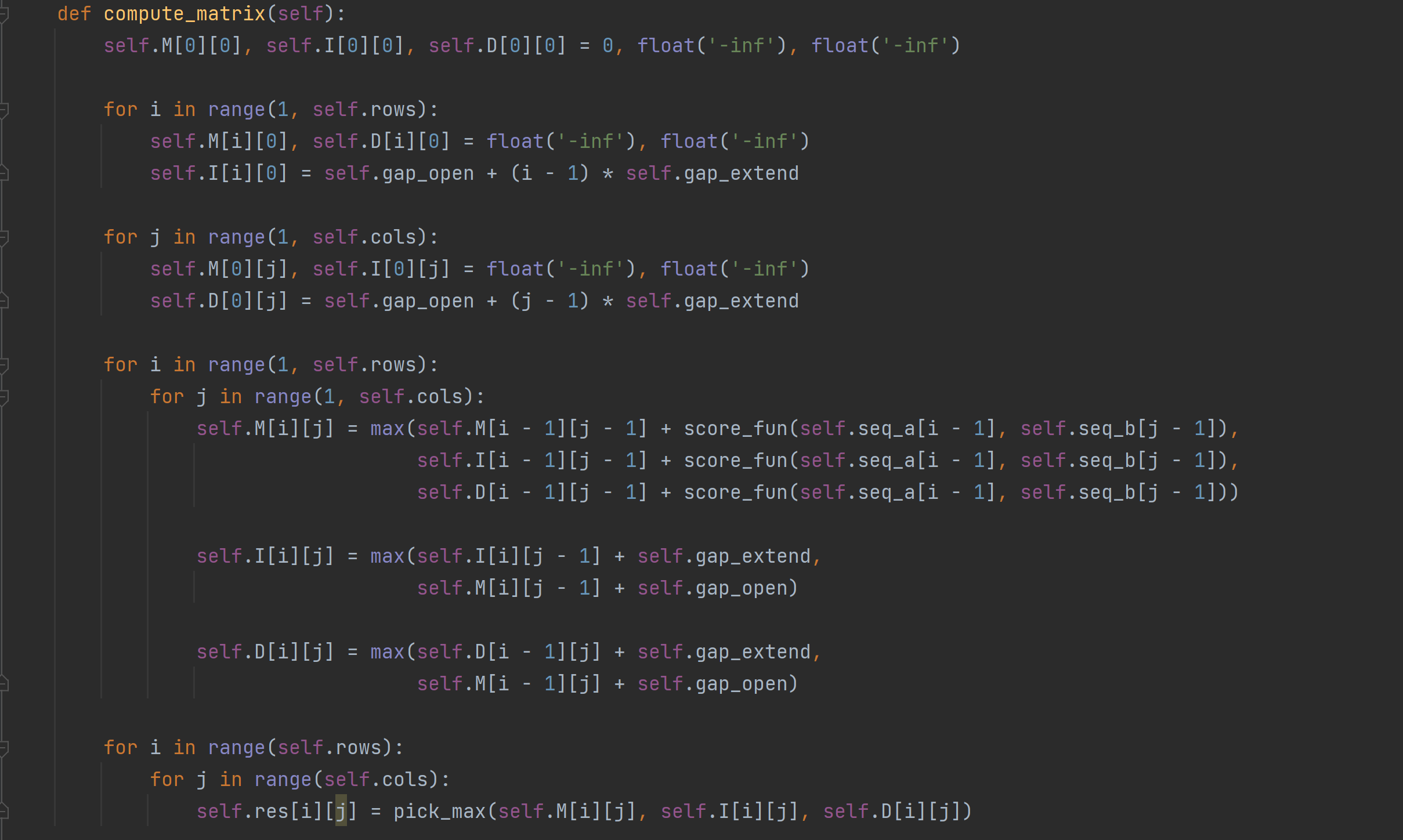
1. *М(i, j)* – Match/Mismatch
2. I(i, j) – Insertion (буква и гэп)
3. *D(i, j)* – Deletion (гэп и буква)
4. *Result(i, j) = max(M(i, j), I(i, j), D(i, j))*

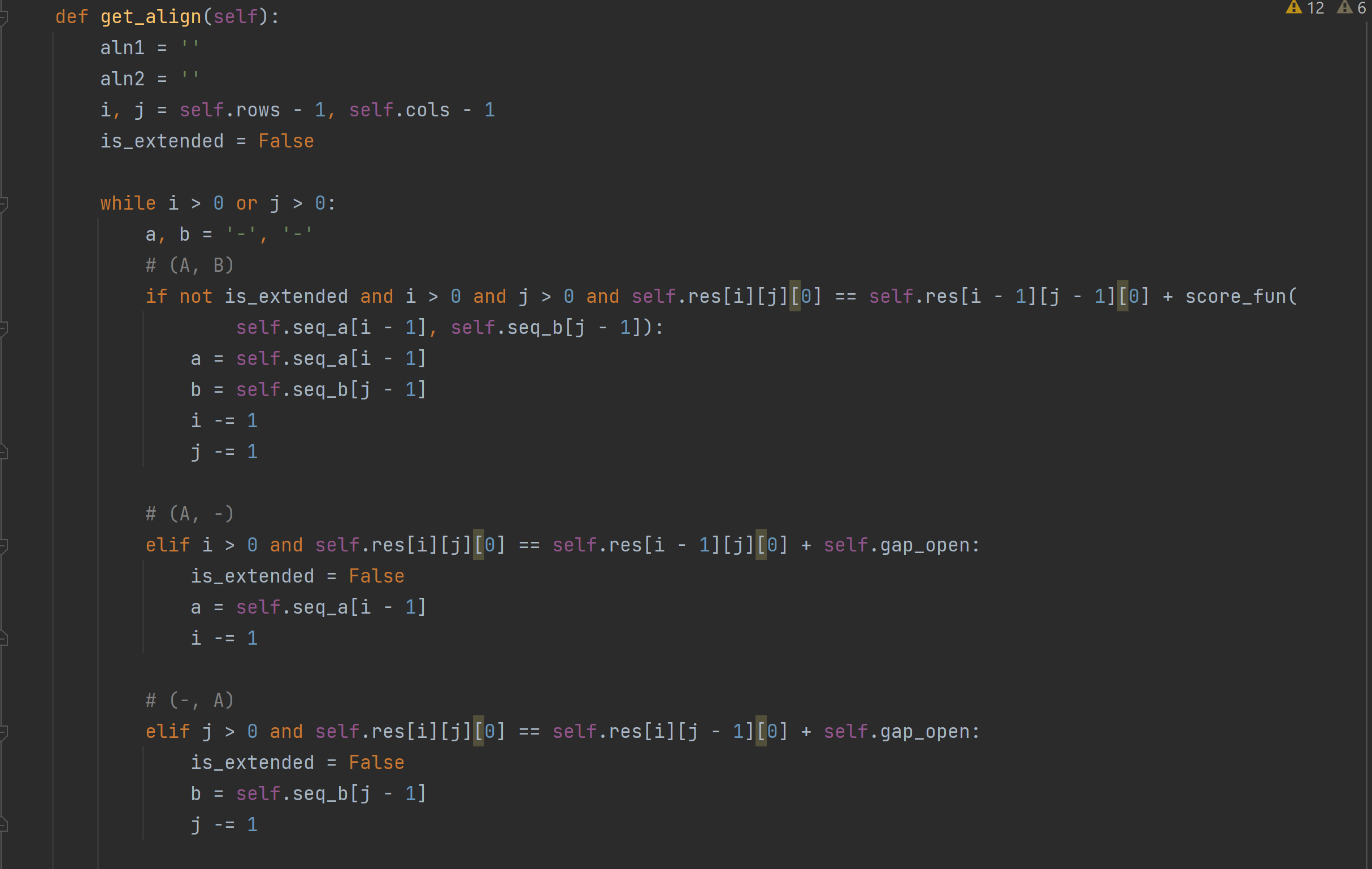
# Реализация

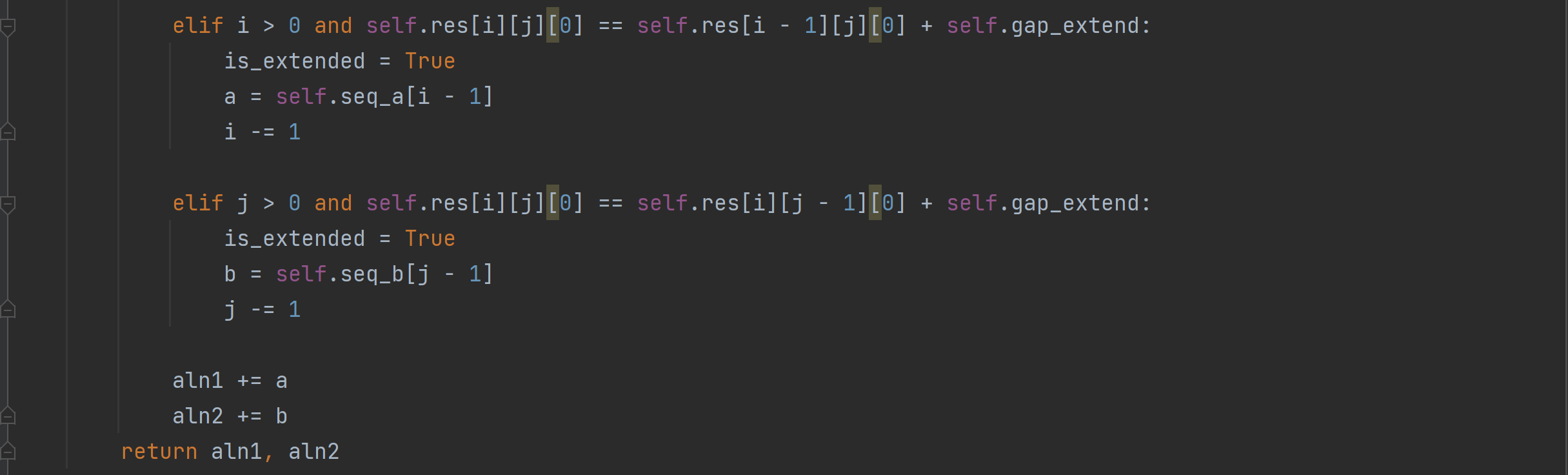
В качестве языка реализации был выбран Python:

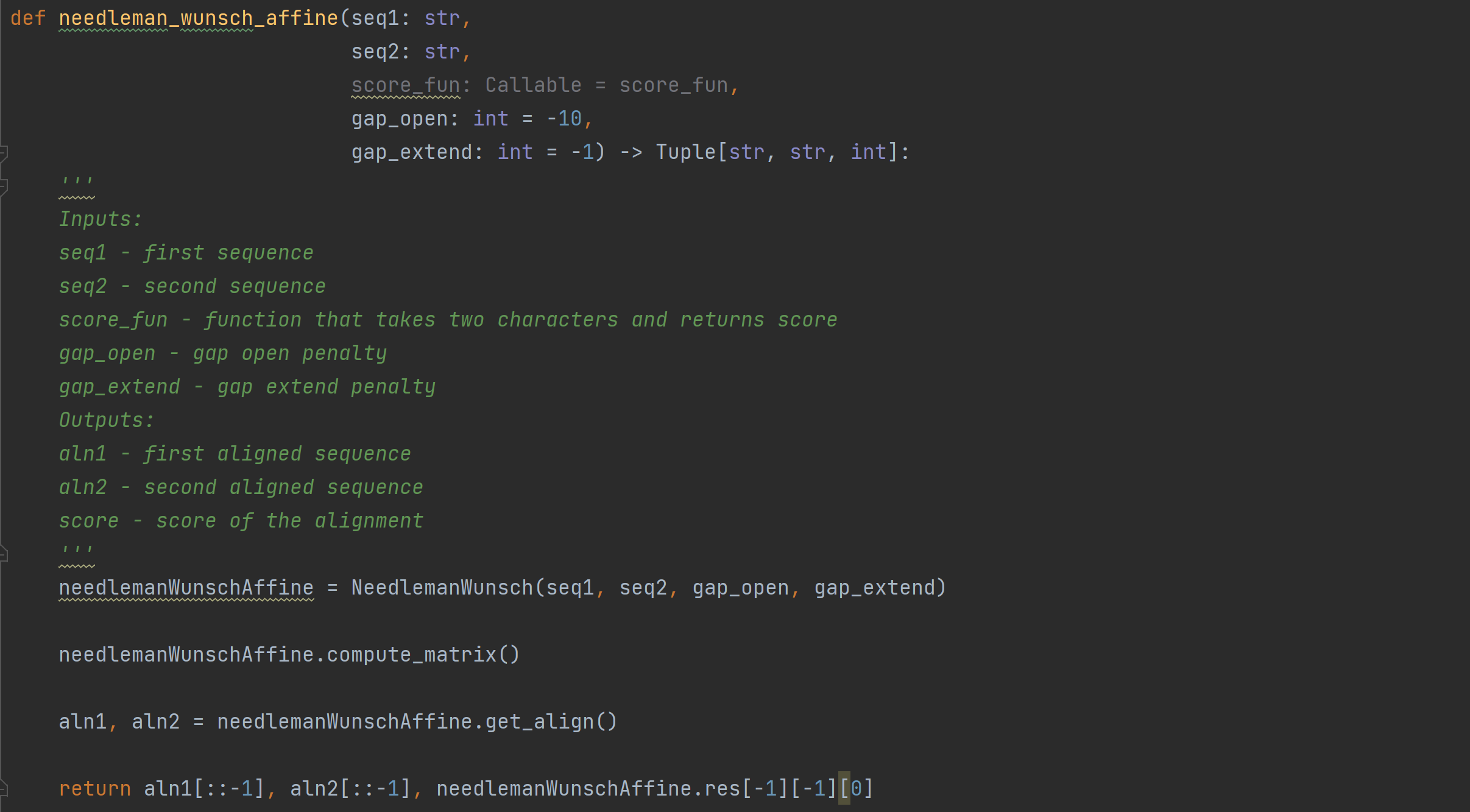












# Тестирование

Для тестирования использовался выложенный на гитхаб файл test.py

При выполнении всех тестов из test.py ни одно предупреждение не появляется в результате => тесты проходят успешно:

