МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное образовательное учреждение высшего образования «Санкт-петербургский государственный морской технический университет»

ФАКУЛЬТЕТ ЦИФРОВЫХ ПРОМЫШЛЕННЫХ ТЕХНОЛОГИЙ

Кафедра Киберфизических систем

ЛАБОРАТОРНАЯ РАБОТА

По дисциплине «Программирование»

Выполнил: Мамашев Олег Игоревич

Проверил: Поделенюк Павел Петрович

Санкт-Петербург

2024

Оглавление

[1. Цели и формулировка задачи 3](#_Toc187676633)

[2. Результаты работы 4](#_Toc187676634)

[2.1. Реализация программы с использованием функционального программирования языка Python 4](#_Toc187676635)

[2.1.1. Ход работы 4](#_Toc187676636)

[2.1.2. Демонстрация работы программы 4](#_Toc187676637)

[2.1.3. Листинг кода 4](#_Toc187676638)

[ЗАКЛЮЧЕНИЕ 5](#_Toc187676639)

[Список использованных источников 6](#_Toc187676640)

# Цели и формулировка задачи

Цели работы – разобрать вопросы:

1. Работа с текстовыми файлами,
2. Работа с локальными и глобальными переменными,
3. Работа с несколькими функциями, а не только с main,
4. Нисходящий метод разработки программ,
5. Анализ ходов.

Формулировка задания:

Дана квадратная шахматная доска размером N x N. На доске уже размещено K фигур. Фигуры размещены так, что находятся не под боем друг друга.

Необходимо расставить на доске еще L фигур так, чтобы никакая из фигур на доске не находилась под боем любой другой фигуры. Необходимо найти все возможные решения.

Вы работаете с фигурой: \*(Задание по вариантам)

Входные данные в файле input.txt. На первой строке файла записаны три числа: N L K (через пробел). Далее следует K строк, содержащих числа x и y (через пробел) - координаты уже стоящей на доске фигуры (фигуры стоят правильно). Координаты отсчитываются от 0 до N-1. 1 <= N <= 20.

Ваш алгоритм должен быть оптимизирован, для работы с доской 20!!

Выходные данные в файл output.txt. На каждое найденное решение необходимо записать в файл одну строку. Строка состоит из пар (x,y) - координаты фигур на доске. В решение следует вывести координаты всех фигур, находящихся на доске. Каждую фигуру необходимо записать в виде пары координат, разделенных запятой и обрамленных скобками. Координаты отсчитываются от 0 до N-1. Порядок, в котором фигуры перечислены в решении, не имеет значения. Если не было найдено ни одного решения, в файл необходимо записать no solutions.

Выходные данные на консоль — это доска N\*N, где фигура обозначается #, ее ходы обозначаются \*, а пустые клетки обозначаются 0.

# Результаты работы

## **Реализация программы с использованием функционального программирования языка Python**

Программа ставит уже расставленные фигуры, а затем оставшиеся в рекурсивной функции с учётом их координат возможного движения (с помощью функции для проверки этих движений). При первой расстановке запускается функция, которая выводит игровое поле в консоль.

## Ход работы

1. Считывание данных с текстового файла input.txt:

* Размер поля (N\*N),
* Количество фигур, которые нужно расставить,
* Количество уже расставленных фигур,
* Координаты расставленных фигур расположить в список в виде кортежей.

1. Реализация функции movements, которая принимает координаты фигуры и возвращает координаты её возможного движения в виде кортежей в списке.
2. Реализация функции figures, которая ставит уже расставленные фигуры на поле и запускает рекурсивную функцию counting. При нахождении первой комбинации запускается функция print\_first\_combination, которая выводит игровое поле в консоль. Если решений программы нет – то в output.txt записывается «No solutions».

## Демонстрация работы программы

## 

Рисунок 1 - input.txt

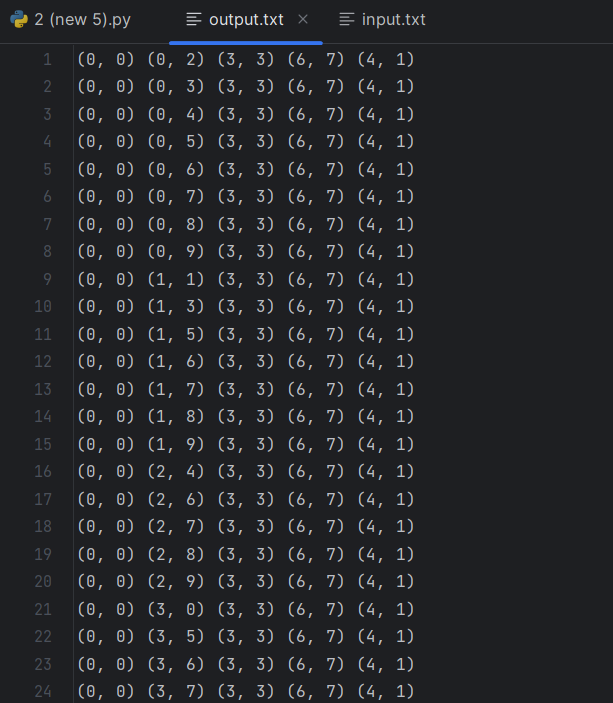


Рисунок 2 - output.txt

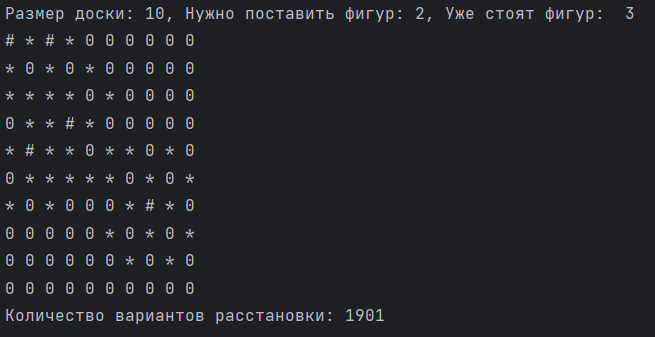
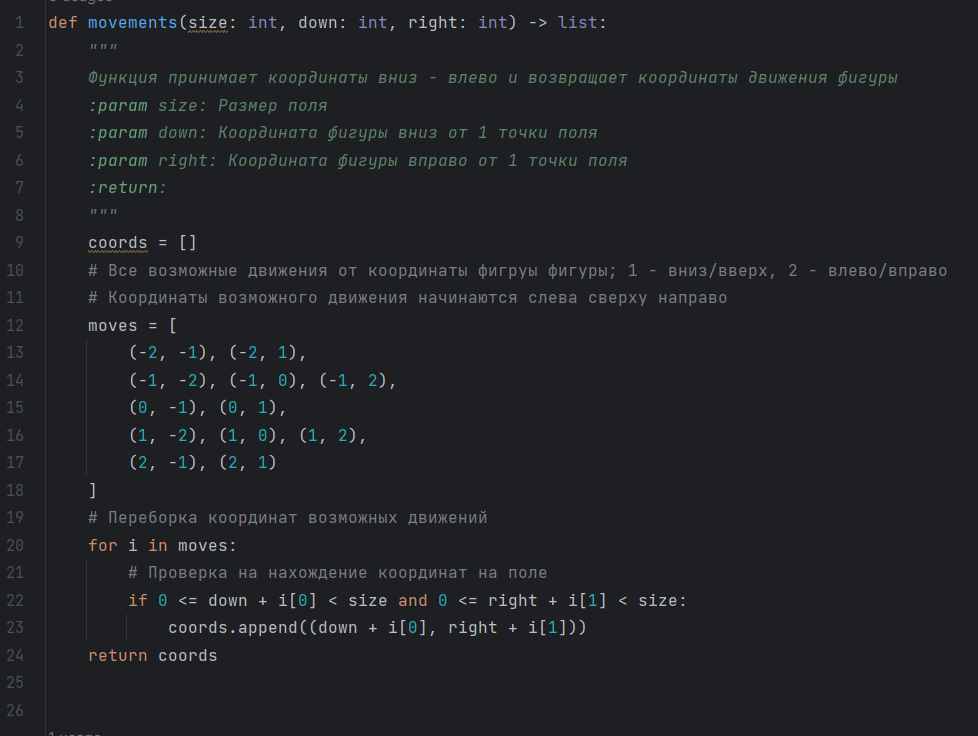
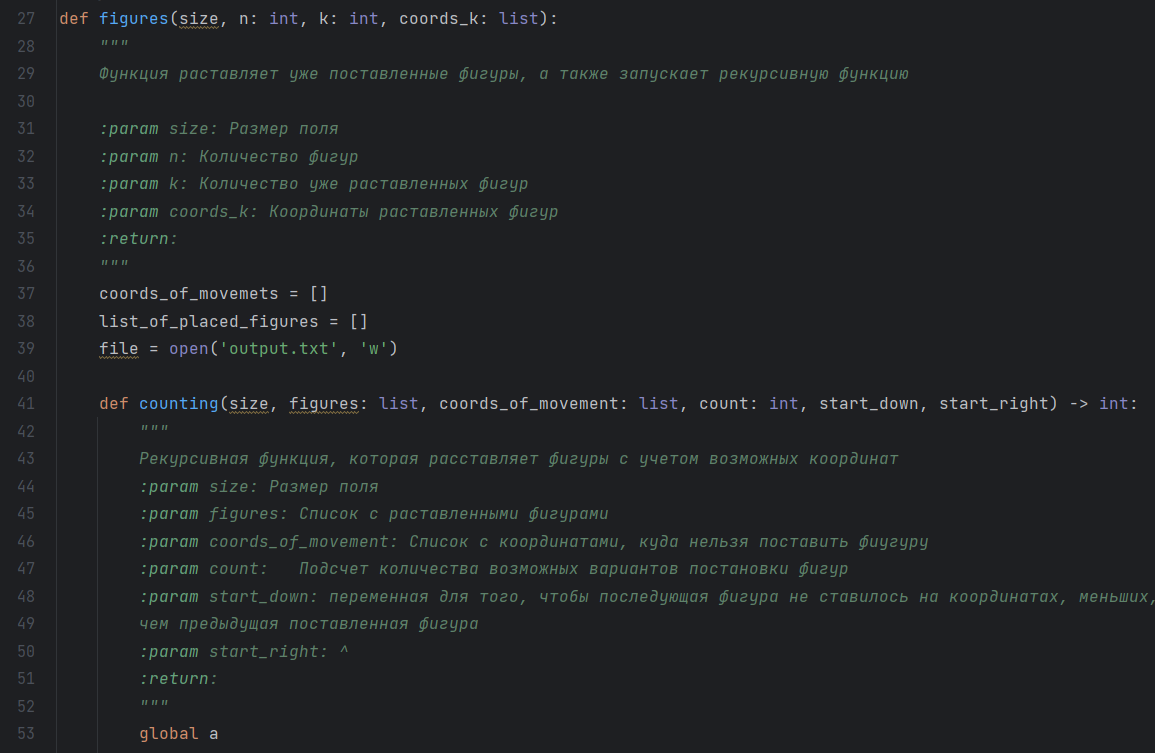
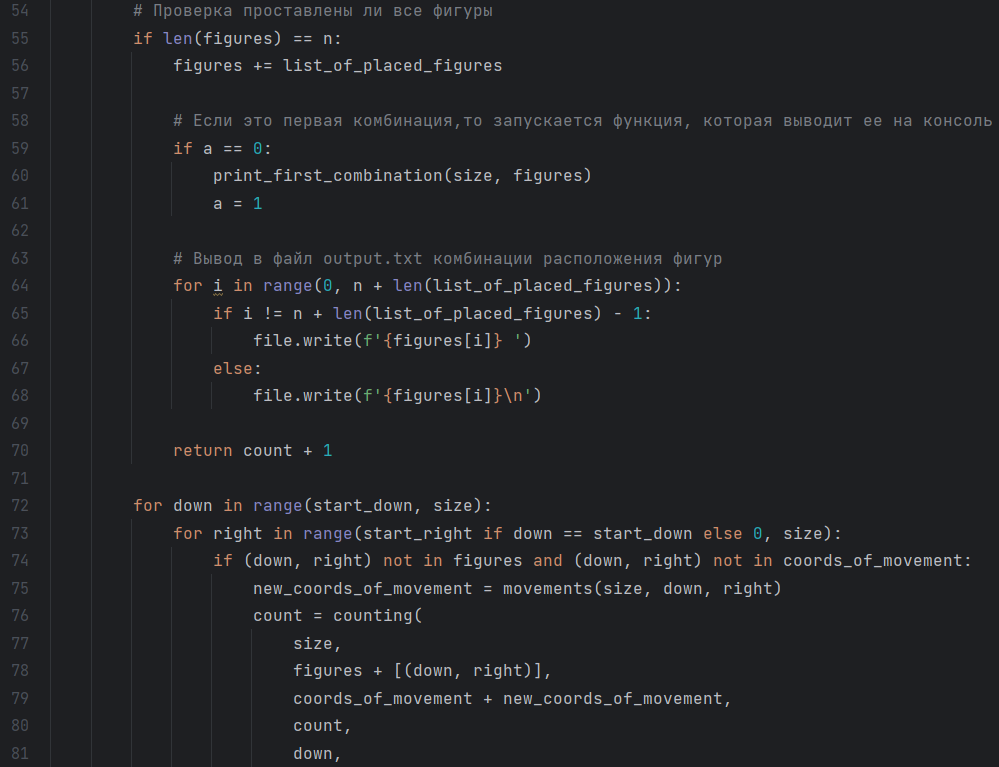


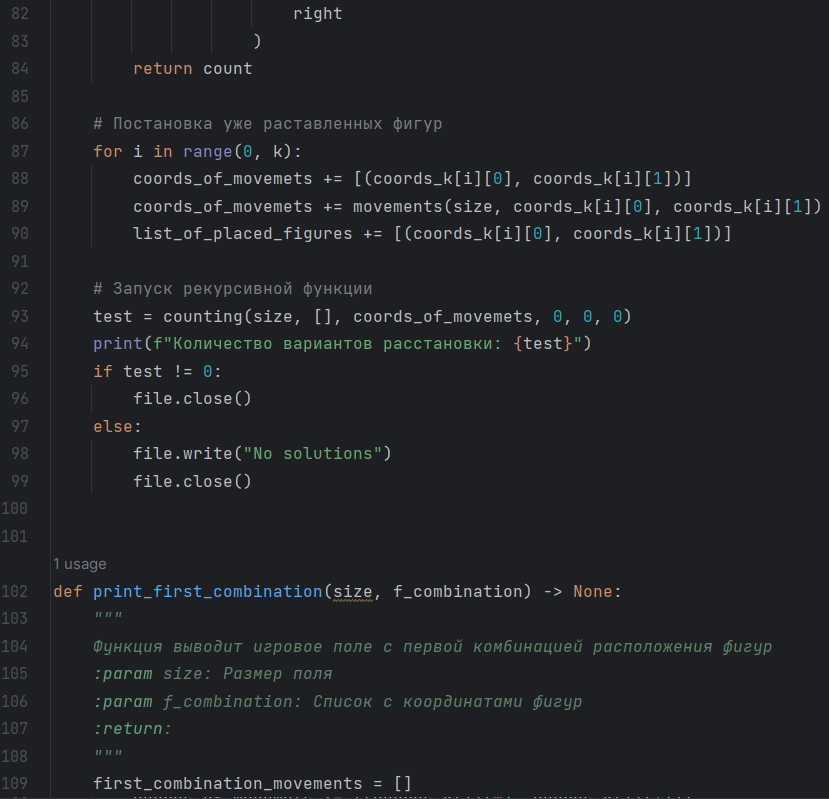
Рисунок 3 - Вывод в консоли

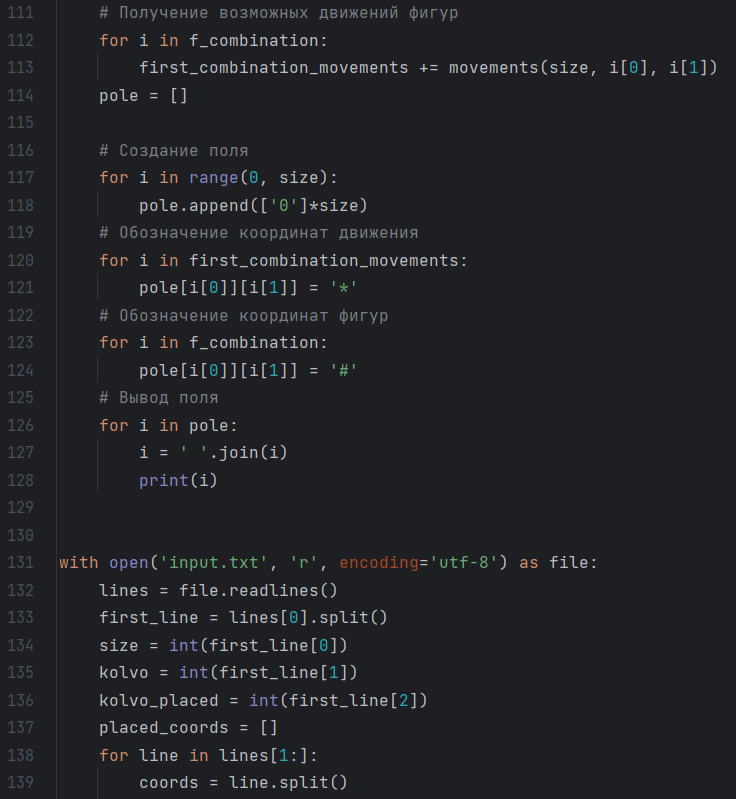
## Листинг кода

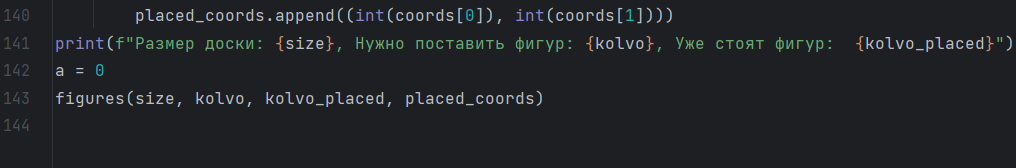












# ЗАКЛЮЧЕНИЕ

В ходе работы была реализована программа, которая получает все возможные комбинации расстановки фигур на поле и записывает их координаты в текстовый файл.

# Список использованных источников