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

КУРСОВАЯ РАБОТА

По дисциплине «Программирование»

Выполнил:

Халилев Владислав Виленович

Проверил:

Санкт-Петербург

2025

## Оглавление

[Оглавление 1](#_Toc199116916)

[3. Цели и формулировка задачи 2](#_Toc1502061571)

[4. Разработка классов и UML диаграмма 3](#_Toc1973487047)

[4.1. Ход работы 3](#_Toc109931564)

[4.2. Демонстрация работы программы 4](#_Toc1240884150)

[5. Заключение 4](#_Toc1148186670)

[6. Список используемых источников 4](#_Toc616955594)

## 3. Цели и формулировка задачи

Цель: Анализ ходов фигуры на шахматной доске При работе над данным лабораторным проектом будут затронуты следующие вопросы: 1. Работа с основами функционального программирования языка Python 2. Работы с основами ООП языка Python 3. Разработка классов и UML диаграмма 4. Работы с пакетами Python 5. Создание GUI приложения с использованием PySide6

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. Требуется найти одно решение для визуализации и все возможные решения для вывода в файл. Если решение не найдено, то необходимо вывести соответствующее сообщение. 1) Необходимо создать UML диаграмму взаимодействия классов. При необходимости можно делать несколько UML диаграмм. Для диаграммы создается табличка взаимодействия (Имя класса :: Имя базового класса :: Описание). Также для каждого класса создается таблица методов и атрибутов (Методы и атрибуты :: Описание ). При описании методов обязательно прописывать тип параметров, а также выходной тип данных. При описании атрибутов необходимо прописывать его тип. 2) Необходимо составить программу с использованием функционального программирования языка Python, где:  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.  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-  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. 3) Необходимо составить программу с использованием объектноориентированного программирования (ООП) на языке Python. Каждый класс должен чётко обозначать своё предназначение. Также необходимо разработать интерфейс с использованием пакета PySide6.

## 4. Разработка классов и UML диаграмма

Ui\_MainWindow - Главный интерфейс программы с шахматной доской, полями ввода и кнопками управления.

MainWindow - Обертка главного окна, запускающая и отображающая интерфейс Ui\_MainWindow.

Ui\_place\_user\_figure\_dialog - Диалоговое окно для ручного выбора позиций фигур на доске.

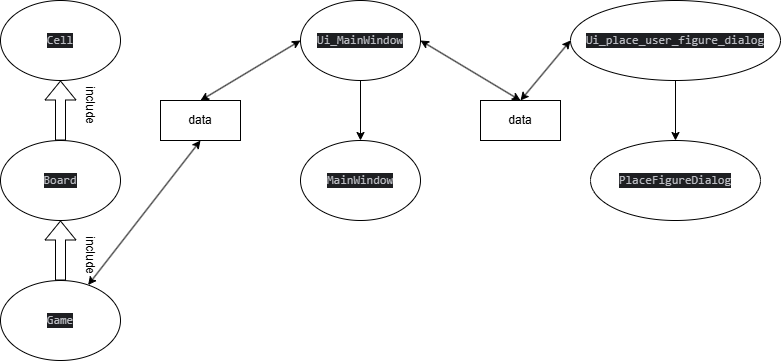
PlaceFigureDialog - Обертка диалога ручной расстановки, передающая выбранные координаты в программу.

Cell - Класс, представляющий одну клетку доски с состояниями "свободна" и "под атакой".

Board - Шахматная доска, состоящая из сетки клеток класса Cell.

Game - Класс с алгоритмом расстановки фигур, исключающим их взаимное атакование.

data - Глобальный словарь для хранения и обмена данными между классами программы.



## 4.1. Ход работы

* Запуск:

Открывается главное окно с шахматной доской, полями для ввода (N - размер доски, L - кол-во фигур) и кнопками:

Старт - запускает расстановку фигур.

Расставить фигуры - открывает окно, где можно вручную выбрать позиции фигур (левая кнопка - добавить, правая - убрать).

Выход - закрывает программу.

Ввод данных:

Пользователь вводит N и L.

Если нужно, добавляет статические фигуры с помощью диалогового окна.

* Расстановка фигур:

При нажатии Старт программа:

Проверяет, что N и L заданы.

Размещает фигуры так, чтобы они не били друг друга.

Сохраняет все варианты в файл output.txt.

* Результат:

Пользователь видит одно из решений на доске.

4.2. Демонстрация работы программы

[https://github.com/Cvif223/20121\_Vlad\_Khalilev/tree/main/2%20семестр/Chess\_For\_Exam](https://github.com/Cvif223/20121_Vlad_Khalilev/tree/main/2%20%D1%81%D0%B5%D0%BC%D0%B5%D1%81%D1%82%D1%80/Chess_For_Exam)

## 5. Заключение

В ходе работы была разработана программа для решения задачи расстановки фигур на шахматной доске по правилам "Альфин-Рыбка". Программа предоставляет удобный графический интерфейс, позволяющий пользователю задавать размер доски, количество фигур, а также вручную размещать дополнительные фигуры.

## 6. Список используемых источников

Конспекты лекций

https://doc.qt.io/qtforpython-6/gettingstarted.html