**Министерство науки и высшего образования Российской Федерации**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**(Университет ИТМО)**

Факультет **Инфокоммуникационных технологий**

Образовательная программа **Мобильные и сетевые технологии**

Направление подготовки(специальность) **09.03.03 Прикладная информатика**

**ЛАБОРАТОРНАЯ РАБОТА №2**

**По дисциплине «Программирование»**

**Тема: Судоку.**

**Выполнил** Грицкевич К.О.; К3141.

**Проверил** Терещенко В.В.

**Дата**

**Санкт-Петербург 2024**

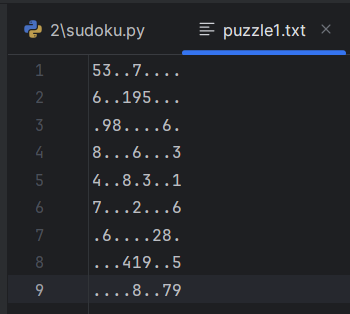
**Цель работы**

Написание программы для решения и генерации пазлов, такой игры как Судоку.

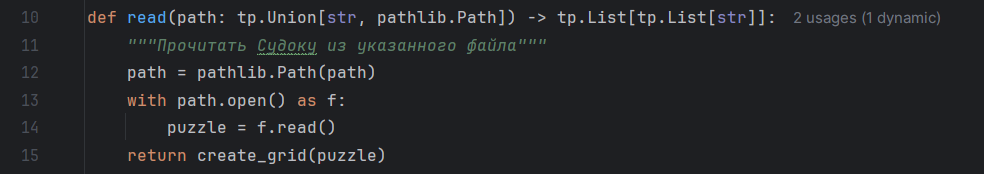
**Ход работы**

*Шаг 1 Чтение и разбивка на строки, колонки и блоки*

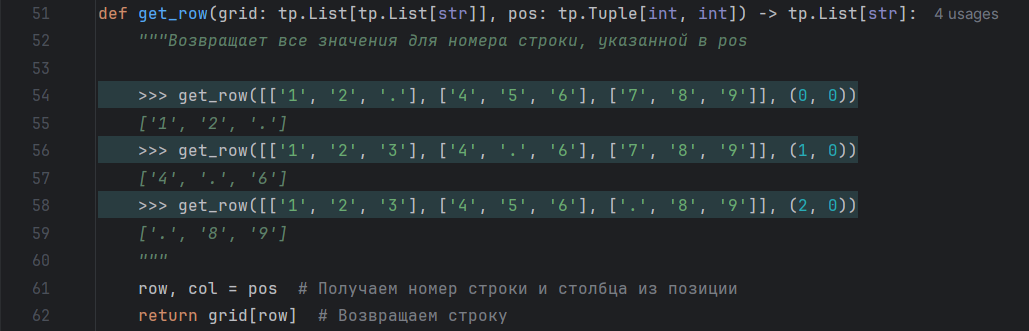
В первой части задания нам предстоит написать код для чтения и разбиения Судоку. Для начала разберемся, как хранится пазл. Он представляет из себя матрицу 9 на 9, записанную в 9 строках в текстовом файле:



Функция, читающая файл, выглядит следующим образом:



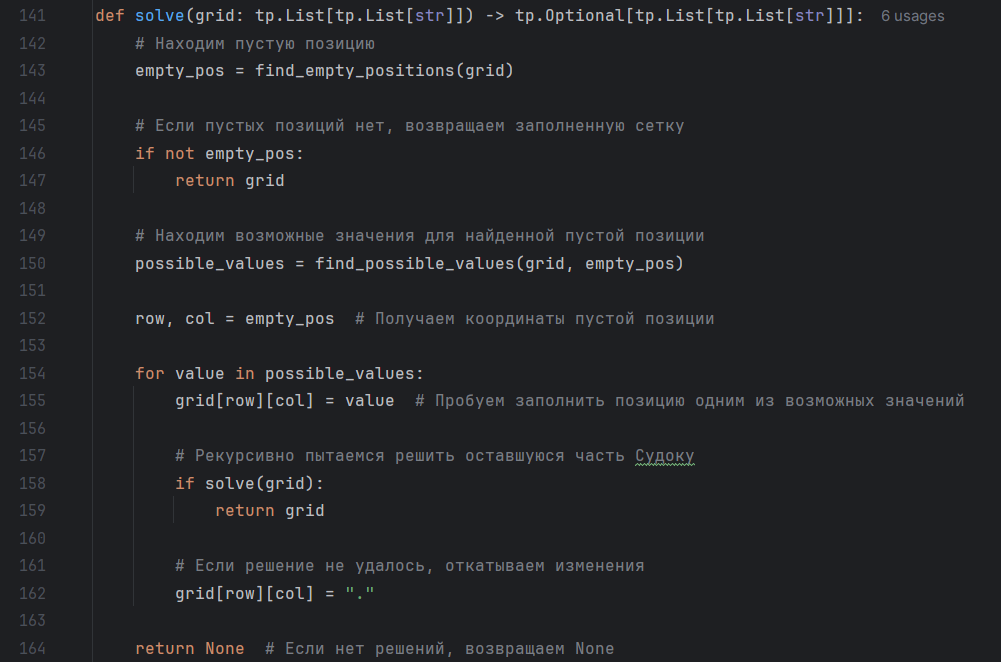
Затем, для дальнейшего решения Судоку, требуется разбивать пазл на строки, колонки и блоки. В каждом из них, должны быть 9 разных цифр. Код для разбиения на строки, выглядит таким образом:



Функции для разбиения на группы и колонки, написаны похожим образом.

*Шаг 2 Алгоритм решения и его проверка*

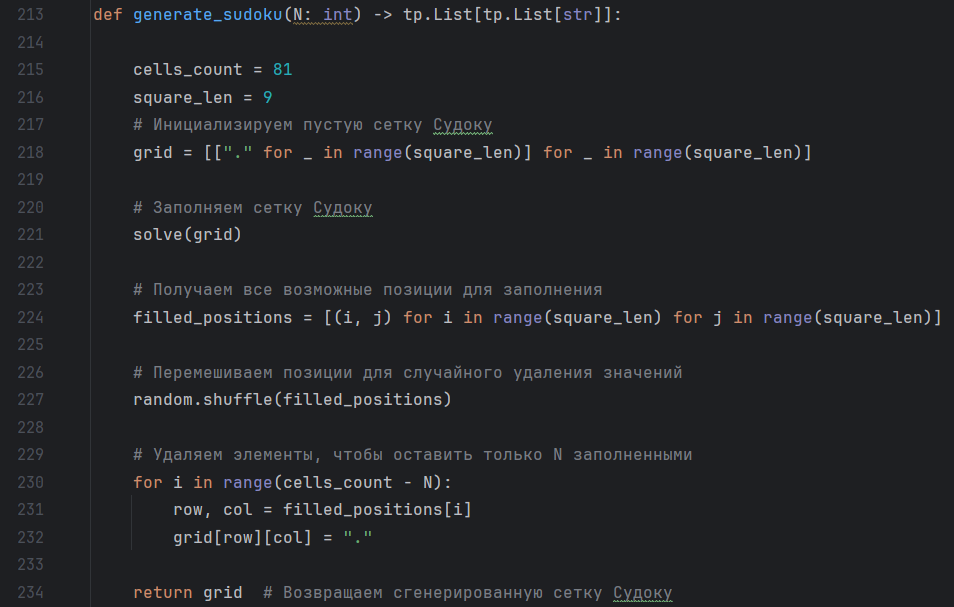
Умея читать Судоку, и разбивать его на различные группы, для определения свободных цифр, мы можем перейти к написанию алгоритма для решения пазлов. Выполнять это мы будем через рекурсию: подставлять одну из доступных цифр и переходить к следующей пустой клетке. В случае, если свободных цифр не останется, мы возвращаемся на шаг назад и выбираем другую цифру.



Проверка будет осуществляться через перебор каждой клетки пазла и проверки его в группах (строчках, колонках и блоках).

*Шаг 3 Генерация новых пазлов*

Одной из задач в данной лабораторной работе, является генерация новых пазлов Судоку. На вход в функцию будет подаваться количество заполненных клеток (0-81). За «случайность» будет отвечать библиотека random.



*Шаг 4 Многопоточность*

Дабы оптимизировать решение Судоку, мы подключим многопоточность, для этого мы будем использовать 2 библиотеки: threading и multiprocessing. Первая позволяет запускать разные задачи на разных ядрах, а вторая – решать задачи параллельно.

**Вывод**

В результате выполнения лабораторной работы, мы написали код для решения и генерации пазлов Судоку, а также поработали с библиотеками, для подключения многопоточности.

**Источники**

[https://ru.wikipedia.org/wiki/Судоку](https://ru.wikipedia.org/wiki/Шифр_Виженера)

<https://docs.python.org/3/library/threading.html>

<https://docs.python.org/3/library/multiprocessing.html>