**Руководство программиста.**

**Руководство программиста на разработку программы "Крестики-нолики".**

**1. Общее описание**

Данное руководство предназначено для программистов, работающих с кодом игры "Крестики-нолики", реализованной на языке Python и написанный на базе библиотеки Tkinter. Руководство содержит информацию о ключевых функциях, структуре программы и алгоритмах, используемых в ней.

**2. Структура программы**

Программа состоит из одного файла, в котором содержится класс TicTacToe, который реализует логику игры и графический интерфейс.

* Класс TicTacToe:

1. init(self, root): Инициализирует графический интерфейс игры, создавая игровое поле, кнопки для хода.
2. Создает матрицу self.board 3x3, которая хранит состояние игрового поля.
3. Инициализирует переменные self.current\_player (текущий игрок), self.game\_over (флаг окончания игры).
4. Создает матрицу self.buttons 3x3, где каждый элемент - это кнопка Tkinter, представляющая ячейку на игровом поле.
5. Функция click\_button(self, row, col): Обрабатывает ход пользователя.

А) Проверяет, не закончилась ли игра.

Б) Проверяет, пуста ли выбранная ячейка.

В) Ставит знак текущего игрока на выбранную ячейку.

Г) Обновляет текст кнопки, соответствующей ячейке.

Д) Проверяет победу текущего игрока.

Е) Проверяет ничью.

Ж) Переключает ход на другого игрока.

З) Вызывает функцию computer\_move(), когда ход перешел к компьютеру.

* + computer\_move (self): Реализует ход компьютера с использованием алгоритма best\_move.

1. Ищет оптимальный ход на основе рекурсивного алгоритма best\_move, который оценивает все возможные ходы и выбирает наиболее выгодный для компьютера.
2. Вызывает функцию button\_click(), чтобы сделать выбранный ход на игровом поле.
   * Best\_move(self, board, is\_player\_move): Рекурсивная функция best\_move, которая находит наиболее выгодный ход для компьютера.
3. Принимает на вход состояние игрового поля board и флаг is\_player\_move, указывающий, является ли текущий ход максимизирующим (компьютер) или минимизирующим (игрок).
4. Рекурсивно просматривает все возможные ходы, оценивая их с помощью функции win().
5. Возвращает оценку хода, которая позволяет определить, насколько он выгодный для компьютера.
   * win(self, player): Проверяет, победил ли пользователь player.
6. Проверяет все возможные линии (горизонтальные, вертикальные и диагональные) на наличие трех знаков в ряд у игрока.
7. Возвращает True, если игрок победил, иначе False.
   * draw(self): Проверяет, закончилась ли игра ничьей.
8. Проверяет, есть ли свободные ячейки на игровом поле.
9. Возвращает True, если все ячейки заняты, иначе False.
   * show\_winner(self, winner): Отображает сообщение о победителе или ничьей с помощью диалогового окна.
   * restart(self): Сбрасывает игру, очищая игровое поле и устанавливая начальные значения переменных.

**3. Алгоритмы**

Алгоритм best\_move:

* + Алгоритм best\_move используется для принятия решения о ходе компьютера.
  + Алгоритм рекурсивно просматривает все возможные ходы на игровом поле, оценивая их с точки зрения компьютера.
  + Каждый ход оценивается по тому, насколько он приближает компьютера к победе или насколько он ухудшает положение игрока.
  + Алгоритм выбирает ход с наибольшей оценкой для компьютера, то есть наиболее выгодный для него.

**4. Рекомендации программисту**

* 1. Программа хорошо структурирована и документирована, что упрощает ее модификацию и отладку.
  2. Алгоритм best\_move реализован с использованием рекурсии, что обеспечивает его эффективность.
  3. Графический интерфейс простой и интуитивно понятный, что делает программу удобной в использовании.

**Заключение**

Данная программа предоставляет пользователям возможность сыграть в классическую игру "Крестики-нолики", и благодаря использованию алгоритма best\_move, компьютер будет соперником на уровне с человеком. Программа может быть расширена дополнительными функциями.