МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Измерительно-вычислительные комплексы»

**РУКОВОДСТВО ПРОГРАММИСТА**

на лабораторную работу №10

по дисциплине «Алгоритмы и структуры данных»

|  |  |
| --- | --- |
| **Подп.и** |  |
| **Инв** |  |
| **Вза** |  |
| **Подп.и** |  |
| **Инв** |  |

**Компьютерная игра «крестики-нолики».**

Р.02069337. №23/737-Вариант 22 РП-

Листов <11>

Исполнитель:

студент гр. ИСТбд-23

Селезнев К.В.

«\_\_» \_\_\_\_\_\_\_\_ 2025 г.

2025

1. **Назначение и условия применения программы**
   1. Назначение и функции, выполняемые приложением

Мобильное приложение "Крестики-нолики" позволяет пользователям играть в классическую игру в цифровом формате. Оно рассчитано на двух игроков, которые по очереди делают ходы на поле 3×3. Приложение предлагает удобный и понятный интерфейс и режим игры против компьютера с интеллектуальным алгоритмом (минимакс).

*Краткий свод правил игры:*

Игроки поочередно ставят "X" или "O" на свободные клетки поля. Побеждает тот, кто первым соберет три своих символа в ряд — по горизонтали, вертикали или диагонали. Если поле заполняется, но ни у кого нет выигрышной комбинации, игра завершается ничьей.

*Приложение предоставляет следующие возможности:*

**Графический интерфейс:** Игровое поле выполнено в виде интерактивной сетки из кнопок. Пользователи делают ходы, нажимая на свободные клетки.

**Ход компьютера:** Для расчета хода компьютера применяется алгоритм минимакса, обеспечивающий оптимальную стратегию.

**Проверка выигрыша:** После каждого хода система проверяет, достигнута ли победа одного из игроков или ничья. В таком случае выводится соответствующее уведомление.

**Перезапуск игры:** В любой момент пользователь может начать новую партию, нажав кнопку "Новая игра".

* 1. Условия, необходимые для использования приложения

*Операционная система*

* Windows (версия 7 и выше)
* macOS (Catalina и выше)
* Linux (Ubuntu 18.04 LTS и выше, Debian 9 и выше, CentOS 7 и выше)

*Платформа*

* Поддерживаемые платформы зависят от выбранной среды разработки и языка программирования. Для Python-приложений обычно достаточно стандартного интерпретатора Python.

*Инструментальная среда*

* Python (версия 3.6 и выше) — основной язык программирования.
* pip — менеджер пакетов для установки зависимостей.

*Библиотеки*

* tkinter — библиотека для создания графического интерфейса.
* messagebox — часть библиотеки tkinter, позволяющая создавать простые диалоговые окна для вывода сообщений и запросов.

1. **Характеристики программы**
   1. Характеристики приложения

Значимые строки кода: ~150

Структуры данных:

* Двумерные списки для представления игрового поля.
* Переменные для хранения состояния игры (текущего игрока, результатов и т.д.).

Алгоритмы:

* Алгоритм минимакса для расчета оптимальных ходов компьютера.
* Простые циклы и условия для проверки состояния игры и обработки ходов.

Используемые библиотеки

1. tkinter – стандартная библиотека Python для создания графического интерфейса пользователя (GUI). Используется для создания окон, кнопок, меток и других элементов интерфейса.
2. messagebox – часть библиотеки tkinter, позволяющая создавать простые диалоговые окна для вывода сообщений и запросов.

Работа приложения

При запуске начинается игра. Игровое поле состоит из 9 кнопок (3x3), каждая из которых соответствует одной клетке на поле.

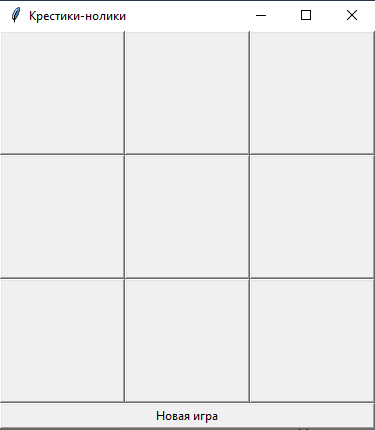
Игроки, начиная с человека, по очереди делают ходы, нажимая на свободные клетки. Программа проверяет допустимость хода и обновляет состояние игрового поля.

Если ходит компьютер, он рассчитывает оптимальный ход с помощью алгоритма минимакса и делает его.

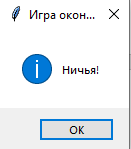
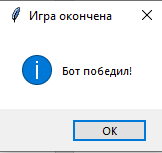
После каждого хода проверяется, есть ли победитель или ничья. Если игра заканчивается, выводится соответствующее сообщение.

После окончания игры пользователю предлагается начать новую игру.

Игровое поле:



Вывод результата(ничья): Вывод результата (выиграл компьютер):

Обеспечение корректности работы программы

1. Проверка допустимости ходов:

* В методе on\_click() выполняется проверка, свободна ли выбранная клетка (if self.board[row \* 3 + col] == " ").
* Ход регистрируется только если клетка пуста, иначе действие игнорируется.

1. Контроль очерёдности ходов:

* Ход игрока (X) обрабатывается только если текущий ход принадлежит ему (self.current\_player == "X").
* Ход бота (O) выполняется автоматически после хода игрока, если игра не завершена.

1. Определение завершения игры:

* После каждого хода вызывается метод check\_game\_over(), который проверяет:
  + Победную комбинацию (check\_winner())
  + Заполнение поля (ничью)
* В случае завершения игры:
  + Выводится информационное сообщение (messagebox.showinfo)
  + Все кнопки блокируются (disable\_all\_buttons())

1. Защита от некорректных действий:

* При нажатии на занятую клетку ход не выполняется, и очередь не переходит к противнику.
* После завершения игры новые ходы невозможны до перезапуска.

1. Корректная работа бота:

* Алгоритм minimax гарантирует, что бот всегда выбирает оптимальный ход.
* Исключены ситуации, когда бот может сделать ход в занятую клетку.
  1. Особенности реализации приложения

В программе используются следующие структуры данных:

1. Представление игрового поля

* Используется одномерный список self.board длиной 9 элементов, где каждый элемент соответствует клетке поля:
  + "X" — ход игрока
  + "O" — ход бота
  + " " — пустая клетка
* Преимущества выбранного подхода:
  + Простота индексации (позиция клетки вычисляется как row \* 3 + col).
  + Удобство для алгоритма minimax, который работает с линейным списком.

Альтернативные варианты:

* Двумерный список (матрица 3×3)
  + Более наглядно отражает геометрию поля.
  + Усложняет некоторые операции (например, перебор всех клеток в minimax).
* Словарь с координатами
  + Например, {(0,0): "X", (0,1): "O", ...}.
  + Повышает читаемость, но требует больше памяти и усложняет проверку победы.

Обоснование выбора:

1. Двумерный список для игрового поля (self.board)

* Преимущества выбранной реализации:
  + Прямое соответствие между логической структурой данных и визуальным представлением поля (3 строки × 3 столбца).
  + Упрощённый доступ к элементам по координатам (например, self.board[row][col]).
  + Наглядность при проверке победных комбинаций (по строкам, столбцам и диагоналям).

Альтернативные варианты и их недостатки:

* Одномерный список:
  + Требует пересчёта индексов (row \* 3 + col), что усложняет чтение кода.
  + Менее интуитивен для операций, связанных с геометрией поля (например, проверка диагоналей).
* Словарь с координатами:
  + Например, {(0,0): "X", (0,1): "O", ...}.
  + Увеличивает объем кода для базовых операций (итерация, проверка победы).

Обоснование выбора:

Двумерный список оптимален для задач, где важна визуальная согласованность и простота отладки. Например, проверка победы реализуется в 3 циклах (по строкам, столбцам и диагоналям), что соответствует правилам игры.

1. Целочисленные переменные для хранения состояния игры

В программе применяются следующие целочисленные переменные:

* self.current\_player

Хранит идентификатор текущего игрока:

* + "X" — ход пользователя
  + "O" — ход компьютера
* self.turn\_count
  + Отслеживает количество сделанных ходов для определения ничьи (максимум 9).

Альтернативные варианты:

* Булевы флаги
  + Например, is\_human\_turn: bool.
  + Подходит только для игр с двумя участниками.
  + Не масштабируется для сложных режимов (например, мультиплеер).
* Строковые константы
  + Как в текущей реализации ("X" и "O").
  + Плюсы:
  + Прямая связь с отображаемыми символами на поле.
  + Минусы:
  + Риск опечаток (например, "x" вместо "X").

Обоснование выбора:

Использование простых целочисленных переменных для хранения состояния игры — это минимальный и эффективный подход. Он позволяет легко манипулировать этими значениями и не требует дополнительных библиотек или классов.

**3. Обращение к программе**

**Методы и алгоритмы:**

**1.** **Метод \_\_init\_\_**

Описание: Инициализирует игровое поле, настраивает интерфейс Tkinter, создает кнопки 3×3 и кнопку перезапуска. Также определяет текущего игрока (X — человек, O — бот) и инициализирует пустое игровое поле.

Назначение: Подготовка игрового интерфейса и начального состояния игры. Если бот ходит первым (current\_player = "O"), сразу вызывает bot\_move().

**2. Метод on\_click**

Параметры:

* row: Индекс строки (0–2).
* col: Индекс столбца (0–2).

Описание: Обрабатывает клик игрока по кнопке. Проверяет, что клетка свободна и ход принадлежит игроку (X). После хода проверяет окончание игры. Если игра продолжается, передает ход боту (current\_player = "O") и вызывает bot\_move().

Назначение: Обеспечивает взаимодействие игрока с полем и переход хода к боту.

**3. Метод bot\_move**

Описание: Реализует ход бота с использованием алгоритма Minimax. Перебирает все свободные клетки, оценивает каждый возможный ход и выбирает оптимальный (с максимальным счетом). После выбора делает ход (make\_move) и передает очередь игроку (current\_player = "X").

Назначение: Автоматический "умный" ход компьютера, гарантирующий либо победу, либо ничью.

**4. Метод minimax**

Параметры:

* board: Текущее состояние доски.
* depth: Глубина рекурсии (учитывается для оптимизации, но не используется).
* is\_maximizing: Флаг, указывающий, максимизирует ли бот счет (True для "O") или минимизирует (False для "X").

Описание: Рекурсивно оценивает все возможные исходы игры, возвращая:

* 1, если побеждает "O",
* -1, если побеждает "X",
* 0 в случае ничьи.

Назначение: Определение оптимального хода для бота на основе полного дерева возможных комбинаций.

**5. Метод make\_move**

Параметры:

* row, col: Координаты клетки.
* player: Символ игрока (X или O).

Описание: Обновляет игровое поле (board) и отключает нажатую кнопку, устанавливая на ней символ player.

Назначение: Фиксация хода на поле и в интерфейсе.

**6. Метод check\_game\_over**

Описание: Проверяет условия окончания игры:

* Победа X или O (через check\_winner),
* Ничья (все клетки заполнены).

Выводит сообщение о результате и блокирует кнопки (disable\_all\_buttons).

Назначение: Завершение игры с выводом итога.

**7. Метод check\_winner**

Параметры:

* board: Опционально — состояние доски для анализа (по умолчанию проверяется текущее).

Описание: Проверяет все возможные линии (горизонтали, вертикали, диагонали) на наличие трех одинаковых символов (X или O). Возвращает None, если победителя нет.

Назначение: Определение победителя или отсутствия выигрышной комбинации.

**8. Метод restart**

Описание: Сбрасывает игровое поле, активирует все кнопки и устанавливает current\_player = "X". Если бот должен ходить первым, вызывает bot\_move().

Назначение: Перезапуск игры без перезагрузки окна.

**Используемые библиотеки:**

1. tkinter

*Описания:* Стандартная библиотека Python для создания графического интерфейса пользователя (GUI). Включает в себя множество виджетов, таких как окна, кнопки, метки и рамки.

*Применение:* Используется для создания главного окна приложения, кнопок игрового поля, меток и других элементов интерфейса.