МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Измерительно-вычислительные комплексы»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | |  | |  |  |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | |  | | Модель приложения  на лабораторную работу №9  по дисциплине «Алгоритмы и структуры данных»  Тема «Игра крестики-нолики» | | | | | | |
|  | | |  | |  | | | |
|  | | Исполнитель  студент гр. ИСТбд-21 Мишин А.В.  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2024 г. | | | |
|  | | 2024 | | | | | | |

**Введение**

Программа "Крестики-нолики" предназначена для реализации классической игры с графическим интерфейсом. Игра поддерживает два режима: игра против другого игрока и игра против компьютера. Программа реализована с использованием языка Python и библиотеки Tkinter для построения графического интерфейса.

**1. Проектная часть**

**1.1 Структуры данных**

Основными структурами данных в программе являются:

**Игровое поле:** Массив (список) размером 3x3, представляющий состояние поля (пустые клетки или занятые крестиками и ноликами).

Пример:

*board = ["", "", "", "", "", "", "", "", ""]*

**Текущий игрок:** Переменная, которая хранит информацию о текущем игроке (крестики или нолики).

Пример:

*current\_player = "❌" # или "⚫"*

**Режим игры:** Переменная, которая определяет, играет ли пользователь против другого игрока или против компьютера.

Пример:

*play\_vs\_computer = False # Если False — игра против другого игрока*

**1.2 Алгоритмы**

* Алгоритм проверки победителя: Метод, проверяющий выигрышные комбинации на поле.
* Алгоритм Minimax: Используется для реализации логики игры против компьютера. Это рекурсивный алгоритм, который вычисляет все возможные ходы и выбирает наилучший ход для компьютера.
* Алгоритм переключения хода: Переключает ход между игроками после каждого клика.
* Алгоритм обновления времени: Отслеживает время игры и обновляет таймер на экране.

**1.3 Интерфейс**

Программа использует tkinter для построения графического интерфейса. Основные элементы интерфейса:

**Главное меню:**

* Кнопки для выбора режима игры ("Играть против компьютера", "Играть против игрока").
* Кнопка выхода из игры.

**Игровое поле:**

* Кнопки, которые представляют клетки 3x3 поля.
* Отображение текущего игрока и времени игры.

**Контрольные кнопки:**

* Кнопки для перезапуска игры и возврата в главное меню.

**1.4 Дополнительные элементы**

Программа включает следующие дополнительные возможности:

* Сохранение и загрузка игры: Сохранение текущего состояния игры в файл и возможность загрузки ранее сохраненной игры.
* Валидация ввода: Проверка правильности введенных данных и состояния игры.
* Таймер: Отображение времени игры в реальном времени.

**1.5 Тестирование**

Цель тестирования — убедиться в корректной работе всех функций программы:

1. Проверка правильности работы алгоритмов (проверка победителя, алгоритм Minimax).
2. Проверка правильности отображения состояния игры.
3. Тестирование переключения хода.
4. Проверка функциональности кнопок.
5. Тестирование подсчета времени и перезапуска игры.

**1.6 Структура файлов приложения**

1. tic\_tac\_toe.py – основной файл программы, содержащий логику игры и интерфейс.
2. game\_state.json – файл для хранения текущего состояния игры (если реализована функция сохранения).
3. config.json – файл для хранения настроек (например, сложности игры с компьютером).