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*** |  | |  | | **Пояснительная записка**  на лабораторную работу №9  по дисциплине «Алгоритмы и структуры данных»  **Тема** «Разработка компьютерной игры Крестики Нолики» | | | | | | |
|  | | |  | |  | | | |
|  | | Исполнитель  студент гр. ИСТбд-21  Вражкин.Н.А.  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2024 г. | | | |
|  | | 2024 | | | | | | |

**Введение**

**Разрабатываемое приложение представляет собой классическую игру "Крестики-нолики", созданную с использованием языка Python и библиотеки Tkinter для создания графического интерфейса. Игра позволяет пользователю сражаться с компьютерным противником, используя алгоритм минимакс для искусственного интеллекта. Разработка демонстрирует навыки работы с графическими интерфейсами, реализации алгоритмов и разработки интерактивных приложений.**

**Основания для разработки:**

**Лабораторная работа по учебному плану направления "Информационные системы и технологии".**

**Постановка задачи**

**Создание игры "Крестики-нолики" с реализацией следующих возможностей:**

1. **Игра против компьютера.**
2. **Удобный графический интерфейс для отображения игрового процесса.**
3. **Реализация искусственного интеллекта на основе алгоритма минимакс.**
4. **Отображение статуса игры (победа, ничья) и возможность перезапуска.**

**Структура программы**

**Основные функции программы:**

1. **check\_winner(board, player): Проверяет, выиграл ли игрок ('X' или 'O').**
2. **is\_full(board): Определяет, заполнено ли поле, что указывает на ничью.**
3. **minimax(board, depth, is\_maximizing): Реализация алгоритма минимакс для выбора оптимального хода компьютера.**
4. **computer\_move(): Логика хода компьютера, основанного на минимакс.**
5. **player\_move(row, col): Обработка хода игрока.**
6. **disable\_all\_buttons(): Блокирует все кнопки при завершении игры.**
7. **reset\_game(): Сбрасывает игру до начального состояния.**

**Функциональное назначение:**

1. **Создание игрового поля 3x3, отображаемого в виде кнопок.**
2. **Реализация алгоритма минимакс для игры компьютера.**
3. **Отображение результатов (победа, ничья) в графическом интерфейсе.**
4. **Возможность перезапуска игры.**

**Проектная часть**

**1. Архитектура и алгоритмы**

**1.1 Архитектура программы:**

* **Программа состоит из следующих компонентов:**
  + **Игровое поле (board): Двумерный список 3x3, отражающий текущее состояние игры.**
  + **Кнопки интерфейса (buttons): Двумерный список кнопок tkinter, связанный с игровым полем.**
  + **Статус игры: Информационные элементы интерфейса для вывода результата или статуса (победа, ничья, ход компьютера/игрока).**

**1.2 Алгоритмы:**

* **Алгоритм минимакс (minimax):**
  + **Просчитывает все возможные ходы.**
  + **Оценивает ходы: победа компьютера = 1, победа игрока = -1, ничья = 0.**
  + **Возвращает оптимальный ход для компьютера.**
* **Проверка победы (check\_winner):**
  + **Проверяет строки, столбцы и диагонали на наличие выигрышной комбинации.**
* **Проверка ничьей (is\_full):**
  + **Проверяет, заполнено ли всё игровое поле.**

**2. Стадии и этапы разработки**

1. **Определение требований к функционалу игры.**
2. **Проектирование структуры кода, функций и интерфейса.**
3. **Реализация игровой логики и ИИ.**
4. **Тестирование корректности работы всех компонентов.**
5. **Оптимизация пользовательского интерфейса.**
6. **Документирование кода.**

**Тестирование**

**Цель тестирования:**

**Проверка корректной работы игровой логики, взаимодействия с пользователем и функционирования искусственного интеллекта.**

**Методика тестирования:**

**Тестирование проводилось методом "черного ящика", где проверялась работа приложения в различных сценариях.**

**Проведённые тесты:**

1. **Проверка работы алгоритма минимакс (компьютер всегда выбирает оптимальные ходы).**
2. **Тестирование корректного определения победителя.**
3. **Проверка сценариев ничьей (поле полностью заполнено).**
4. **Тестирование взаимодействия кнопок интерфейса.**
5. **Проверка возможности перезапуска игры.**
6. **Проверка блокировки кнопок после завершения игры.**
7. **Интеграционное тестирование взаимодействия всех компонентов.**

**Результаты тестирования:**

1. **Обнаруженные ошибки в алгоритме проверки победы были устранены.**
2. **Оптимизирована работа кнопок интерфейса.**
3. **Приложение корректно работает во всех заявленных режимах.**

**Заключение**

**Разработанное приложение "Крестики-нолики" соответствует поставленным задачам и обладает следующими преимуществами:**

* **Полноценная игровая логика для игры против компьютера.**
* **Простота и удобство пользовательского интерфейса.**
* **Использование алгоритма минимакс для эффективной реализации искусственного интеллекта. Программа является отличным примером реализации интерактивного приложения с использованием Python и Tkinter.**

**Источники, использованные при разработке**

1. **Лутц М. *Изучаем Python*. – СПб.: Символ-Плюс, 2011.**
2. [**Документация Tkinter**](https://docs.python.org/3/library/tkinter.html)**.**
3. **Рассел С., Норвиг П. *Искусственный интеллект: современный подход*. – М.: Вильямс, 2006.**
4. **ГОСТ 19.701-90. *Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения*. – М.: Издательство стандартов, 1990.**
5. **Саммерфилд М. *Python на практике*. – М.: ДМК Пресс, 2014.**