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  Саранцев С.А.  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2024 г. | | | |
|  | | 2024 | | | | | | |

**Введение**

Разрабатываемое приложение представляет собой классическую игру "Крестики-нолики", реализованную с использованием языка программирования Python и библиотеки Tkinter для создания графического интерфейса. Целью приложения является предоставление пользователям возможности играть в "Крестики-нолики" как против другого игрока, так и против компьютера. Программа демонстрирует навыки программирования, создания пользовательских интерфейсов и реализации игровой логики.

**Основания для разработки:**

Задание для лабораторной работы по учебному плану направления «Информационные системы и технологии».

**Постановка задачи**

Цель разработки заключается в создании игры "Крестики-нолики" с возможностью игры против компьютера или другого игрока, реализацией игровой логики и удобным пользовательским интерфейсом.

**Структура программы**

**Основные функции:**

1. 1. initialize\_ui - инициализация пользовательского интерфейса.

2. initialize\_board - создание игрового поля.

3. set\_first\_move - установка первого хода.

4. player\_vs\_player - режим игры против игрока.

5. player\_vs\_ai - режим игры против компьютера.

6. on\_click - обработка нажатия на игровое поле.

7. make\_move - выполнение хода.

8. ai\_move - ход компьютера.

9. minimax - алгоритм минимакс для ИИ.

10. check\_win - проверка победы.

11. check\_draw - проверка ничьей.

12. reset\_game - сброс игры.

**Функциональное назначение:**

1. Создание игрового поля 3x3.

2. Реализация игры против другого игрока.

3. Реализация игры против компьютера с использованием алгоритма минимакс.

4. Отображение текущего хода и статуса игры.

5. Возможность перезапуска игры и выбора режима игры.

6. Выбор первого хода в каждом режиме.

**Проектная часть**

**1.1 Математические методы**

В приложении используется алгоритм минимакс для реализации искусственного интеллекта компьютерного игрока. Этот алгоритм позволяет компьютеру выбирать оптимальные ходы путем анализа всех возможных вариантов развития игры и выбора хода, максимизирую его шансы на победу компьютера и минимизирующего шансы на победу игрока.

**1.2 Архитектура и алгоритмы**

**1.2.1 Архитектура**

Программа реализована в виде класса TicTacToe, который содержит:

• Игровое поле (self.board).

• Кнопки интерфейса (self.buttons).

• Режим игры (self.play\_with\_ai).

• Текущий игрок (self.current\_player).

• Состояние игры (self.game\_over).

**1.2.2 Алгоритмы**

• Алгоритм минимакс для ИИ: рекурсивно просчитывает все возможные ходы и их последствия.

• Алгоритм проверки победы: проверяет все возможные выигрышные комбинации.

• Алгоритм проверки ничьей: проверяет заполненность всего поля.

**Стадии и этапы разработки**

1. Определение требований: постановка задач и определение функциональности игры.

2. Проектирование: разработка структуры функций и пользовательского интерфейса.

3. Реализация: написание кода программы.

4. Тестирование: проверка работоспособности всех функций и режимов игры.

5. Оптимизация: улучшение производительности и пользовательского интерфейса.

6. Документирование: создание пояснительной записки и комментариев к коду.

**Тестирование**

**Цель тестирования**

Цель тестирования заключается в проверке корректности работы игровой логики, взаимодействия с пользователем и функционирования искусственного интеллекта компьютерного игрока.

**Методика тестирования**

Тестирование проводилось методом "черного ящика", где проверялись функциональные возможности игры при различных сценариях.

**Тестирование**

1. Модульное тестирование: проверка работы отдельных функций.

2. Интеграционное тестирование: проверка взаимодействия между различными компонентами программы.

3. Функциональное тестирование: проверка соответствия игры заданным требованиям.

4. Тестирование пользовательского интерфейса: проверка удобства использования и отзывчивости интерфейса.

5. Проверка корректности определения победителя.

6. Проверка работы алгоритма минимакс в игре против компьютера.

7. Проверка корректности переключения ходов между игроками.

8. Проверка функциональности кнопок управления (перезапуск, выбор режима).

**Результаты тестирования**

Были выявлены и исправлены ошибки в логике определения победителя и работе алгоритма минимакс. Улучшен пользовательский интерфейс для более удобного взаимодействия с игрой.

**Заключение**

Разработанная игра "Крестики-нолики" полностью соответствует поставленным задачам и обеспечивает возможность игры как против другого игрока, так и против компьютера. Программа имеет удобный пользовательский интерфейс и реализует все необходимые функции классической игры "Крестики-нолики".

**Источники, использованные при разработке**

1. Лутц М. Изучаем Python. – СПб.: Символ-Плюс, 2011.

2. "Документация Tkinter" [Электронный ресурс] // Официальная документация Python: https://docs.python.org/3/library/tkinter.html – Режим доступа: свободный.

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.