Автономная некоммерческая организация профессионального образовательного учреждения

"Московский международный колледж цифровых технологий Академия ТОП"

Специальность: 09.02.07 "Информационные системы и программирование"

---

## ИНДИВИДУАЛЬНЫЙ ПРОЕКТ

"Разработка игры Ultimate Tic Tac Toe"

---

\*\*Руководитель:\*\*

\_\_\_\_\_\_\_С. Б. Татаринова

\*\*Выполнил:\*\*

\_\_\_\_\_\_\_[Ваше имя]

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024

---

## ВВЕДЕНИЕ

С развитием компьютерных технологий классические настольные игры переходят в цифровой формат, приобретая новые возможности и улучшенный пользовательский интерфейс. Одной из таких игр являются "Крестики-нолики", которая получила новое развитие в виде "Ultimate Tic Tac Toe" - усложнённой версии с дополнительными правилами и стратегическими возможностями.

Целью данного проекта является создание современной компьютерной реализации игры "Ultimate Tic Tac Toe" с интуитивно понятным интерфейсом, анимационными эффектами и возможностью настройки параметров игры. Программа разработана на языке Python с использованием библиотеки Pygame для графического интерфейса.

Основные особенности реализации:

- Многоуровневая игровая доска (9 больших клеток, каждая из которых содержит 9 малых)

- Плавные анимации и визуальные эффекты

- Гибкая система настроек (язык интерфейса, тема оформления)

- Система отладки и мониторинга состояния игры

- Возможность отмены ходов

Разработанное приложение может использоваться как для развлечения, так и для развития стратегического мышления, поскольку игра требует более глубокого анализа ситуации по сравнению с классическими крестиками-ноликами.

---

## ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

### Правила игры Ultimate Tic Tac Toe

Ultimate Tic Tac Toe (UTTT) - это усложнённая версия классических крестиков-ноликов. Основные отличия:

1. Игровое поле состоит из 9 больших клеток, каждая из которых представляет собой поле для обычных крестиков-ноликов 3×3.

2. Первый игрок может поставить свой знак в любую клетку любого большого поля.

3. Следующий игрок должен сделать ход в маленькой клетке, соответствующей позиции, куда предыдущий игрок поставил свой знак в маленькой клетке.

4. Если игрок выигрывает маленькое поле (по правилам обычных крестиков-ноликов), всё поле отмечается как выигранное этим игроком.

5. Цель игры - выиграть 3 больших поля в ряд (по горизонтали, вертикали или диагонали).

### Технологии разработки

Для реализации проекта были выбраны следующие технологии:

1. \*\*Python\*\* - интерпретируемый язык программирования высокого уровня, обладающий простым синтаксисом и богатой экосистемой библиотек.

2. \*\*Pygame\*\* - библиотека для создания игр и мультимедийных приложений, предоставляющая:

- Управление графикой и анимацией

- Обработку пользовательского ввода

- Звуковые эффекты

3. \*\*JSON\*\* - формат для хранения настроек и состояния игры.

Преимущества выбранного стека технологий:

- Кроссплатформенность (приложение может работать на Windows, Linux, macOS)

- Быстрая разработка прототипов

- Богатые возможности для создания 2D-графики

- Простота модификации и расширения функционала

---

## СПЕЦИАЛЬНАЯ ЧАСТЬ

### Архитектура проекта

Проект имеет модульную структуру:

1. \*\*UTTT.py\*\* - основной файл, содержащий класс Main и точку входа в программу

2. \*\*function/game.py\*\* - логика игры (правила, состояние, обработка ходов)

3. \*\*function/Reader.py\*\* - работа с файлами (загрузка и сохранение настроек)

4. \*\*display/main.py\*\* - основной класс отображения, управление интерфейсом

5. \*\*display/game.py\*\* - визуализация игрового процесса

### Основные классы и их функции

#### Класс Main (UTTT.py)

- Инициализирует приложение

- Управляет основным циклом игры

- Обрабатывает пользовательский ввод (клавиатура, мышь)

- Управляет состоянием программы (загрузка, игра, завершение)

#### Класс Game (function/game.py)

- Хранит состояние игрового поля (self.cells)

- Обрабатывает логику ходов (методы selectbigcell, selectsmallcell)

- Проверяет условия победы (wincheck)

- Реализует систему отмены ходов (loadsave)

- Управляет таймером игры

#### Класс Display (display/main.py)

- Отрисовывает интерфейс игры

- Управляет анимациями

- Реализует режим отладки (F3)

- Обрабатывает цветовые схемы

### Ключевые алгоритмы

1. \*\*Определение выбранной клетки\*\*:

```python

def selectbigcell(self,m):

for x in range(3):

for y in range(3):

if self.x < -100+200\*x and self.x > -300+200\*x and self.y < 300-200\*y and self.y > 100-200\*y:

return 3\*y+x

return None

```

2. \*\*Проверка победы в малом поле\*\*:

```python

def capture(self,m):

capture = False

for p in range(2):

for i in range(3):

if self.cells[self.selected\_cell][0+3\*i] == p and self.cells[self.selected\_cell][1+3\*i] == p and self.cells[self.selected\_cell][2+3\*i] == p:

self.cells[self.selected\_cell] = [p,p,p,p,p,p,p,p,p]

capture = True

break

```

3. \*\*Система анимаций\*\*:

```python

def select(self,m):

z = (self.offset[11]+self.selectsize)\*m.config['zoom']

x = m.width//2+m.Disp.offset[0] + self.selectpos[0]\*m.config['zoom'] + self.selectoffset[0]\*m.config['zoom']

y = m.height//2-m.Disp.offset[1] - self.selectpos[1]\*m.config['zoom'] - self.selectoffset[1]\*m.config['zoom']

...

```

### Особенности реализации

1. \*\*Гибкая система координат\*\*:

- Использование относительных координат с учётом масштабирования (zoom)

- Центрирование игрового поля с возможностью смещения (offset)

2. \*\*Плавные переходы\*\*:

- Анимация выбора клетки (изменение размера и цвета)

- Плавное появление элементов интерфейса

3. \*\*Система настроек\*\*:

- Хранение конфигурации в JSON-файле

- Возможность изменения темы оформления

- Настройки языка интерфейса

4. \*\*Режим отладки (F3)\*\*:

- Отображение технической информации

- Мониторинг состояния игры

- Визуализация внутренних структур данных

---

## ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

Разработанное приложение соответствует следующим требованиям информационной безопасности:

1. \*\*Защита данных пользователя\*\*:

- Все настройки хранятся локально в файле config.json

- Отсутствие сбора персональных данных

- Нет необходимости в подключении к интернету

2. \*\*Устойчивость к ошибкам\*\*:

- Обработка исключений при работе с файлами

- Проверка корректности ввода

- Восстановление состояния после сбоев

3. \*\*Защита от несанкционированного доступа\*\*:

- Отсутствие уязвимостей инъекций кода

- Ограниченный набор системных вызовов

- Изолированное выполнение

4. \*\*Методы обеспечения безопасности\*\*:

- Валидация входных данных

- Ограниченные права доступа к файловой системе

- Шифрование критически важных данных (если потребуется в будущем)

---

## СПИСОК ЛИТЕРАТУРЫ

1. Python Documentation. — URL: https://docs.python.org/3/

2. Pygame Documentation. — URL: https://www.pygame.org/docs/

3. Марк Лутц. Изучаем Python. — СПб.: Символ-Плюс, 2021.

4. Уилл Макгоуэн. Разработка игр на Python с Pygame. — М.: ДМК Пресс, 2020.

5. ГОСТ 19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем.

6. ГОСТ 34.201-89 Виды, комплектность и обозначения документов при создании автоматизированных систем.

---

## ПРИЛОЖЕНИЯ

### Приложение 1. Техническое задание

\*\*1. Назначение разработки\*\*

Создание компьютерной реализации игры "Ultimate Tic Tac Toe" с расширенным функционалом.

\*\*2. Требования к функциональным характеристикам\*\*

- Реализация правил UTTT

- Система отмены ходов (Ctrl+Z)

- Настройки графического интерфейса

- Таймер игры

- Анимационные эффекты

\*\*3. Требования к надежности\*\*

- Стабильная работа при продолжительном использовании

- Восстановление после ошибок ввода

- Сохранение настроек между сеансами

\*\*4. Условия эксплуатации\*\*

- ОС: Windows 7/10/11, Linux, macOS

- Python 3.8+

- Библиотека Pygame

---

### Приложение 2. Руководство пользователя

\*\*1. Запуск игры\*\*

Выполнить файл UTTT.py

\*\*2. Управление\*\*

- ЛКМ - выбор клетки

- ПКМ + движение - смещение поля

- Ctrl + колёсико - масштабирование

- F3 - режим отладки

- F11 - полноэкранный режим

- Ctrl+Z - отмена хода

\*\*3. Настройки\*\*

Хранятся в файле function/config.json:

- "lang": "en"/"ru" - язык интерфейса

- "them": "dark" - тема оформления

- "zoom": 1.0 - масштаб

- "fullscreen": false - полноэкранный режим

---

### Приложение 3. Примеры экранов

\*\*Рис. 1.\*\* Начальный экран с анимацией

![Начальный экран](screenshots/start.png)

\*\*Рис. 2.\*\* Игровой процесс

![Игровой процесс](screenshots/game.png)

\*\*Рис. 3.\*\* Режим отладки (F3)

![Режим отладки](screenshots/debug.png)

---

### Приложение 4. Текст программы

Основные модули (см. исходный код в п.1 задания):

1. UTTT.py - основной файл

2. function/game.py - логика игры

3. display/game.py - отображение игрового процесса

Пример ключевого кода:

```python

class Game:

def \_\_init\_\_(self,m):

self.player = 0

self.selected\_cell = None

self.cells = []

for i in range(9):

self.cells.append([])

for j in range(9):

self.cells[i].append(None)

self.memory = []

self.timer = {'tick':0,'seconds':0,'minutes':0}

self.win = None

```

---

### Приложение 5. Паспорт программы

\*\*1. Наименование\*\*

Ultimate Tic Tac Toe 2.8.11 DEV

\*\*2. Назначение\*\*

Компьютерная реализация усложнённой версии крестиков-ноликов

\*\*3. Системные требования\*\*

- Python 3.8+

- Pygame 2.0+

- 100 Мб свободного места

\*\*4. Автор\*\*

[Ваше имя], студент группы [номер группы]

\*\*5. Дата создания\*\*

[Дата]

\*\*6. Версия\*\*

2.8.11 DEV