# ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №1

## Тема: “Разработка консольной игры ‘Крестики-нолики’ на Flutter/Dart”

### ИНФОРМАЦИЯ О СТУДЕНТЕ

* **ФИО**: Кононыхин Евгений Евгеньевич
* **Группа**: п50-7-22
* **Курс**: 4 курс
* **Дисциплина**: Flutter/Dart
* **Дата выполнения**: 15 сентября 2024

### ЦЕЛЬ РАБОТЫ

Разработать консольную игру “Крестики-нолики” с различными режимами игры, демонстрирующую навыки программирования на языке Dart и работу с Flutter SDK.

### ТЕХНИЧЕСКОЕ ЗАДАНИЕ

#### Требования на оценку 3:

* Приложение запрашивает у пользователя размер игрового поля
* Игроки поочередно делают ходы (начиная с X)
* Определение победителя или патовой ситуации

#### Требования на оценку 4:

* Случайный выбор первого игрока в каждой игре
* Возможность запустить новую игру без выхода из приложения

#### Требования на оценку 5:

* Режим игры друг против друга
* Режим игры против робота
* Реализован ИИ для робота

### РЕАЛИЗОВАННЫЙ ФУНКЦИОНАЛ

#### 1. Основные возможности:

* **Размер поля**: от 3x3 до 10x10 (настраивается пользователем)
* **Режимы игры**:
  + Игра против друга (два игрока)
  + Игра против робота (игрок vs ИИ)
* **Интерфейс**: консольный с красивым отображением и эмодзи
* **Валидация**: проверка корректности ввода координат

#### 2. Искусственный интеллект:

* **Стратегия робота**:
  1. Попытка выиграть (поставить свой символ для победы)
  2. Блокировка выигрышного хода игрока
  3. Случайный ход в свободную клетку
* **Уровень сложности**: средний (можно выиграть, но нужно думать)

#### 3. Дополнительные возможности:

* Случайный выбор первого игрока
* Красивое отображение игрового поля с координатами
* Обработка ошибок ввода
* Меню выбора действий после игры
* Возможность новой игры без перезапуска

### СТРУКТУРА ПРОЕКТА

Практическая работа 1/  
├── lib/  
│ ├── main.dart # Точка входа в приложение  
│ └── game\_logic.dart # Основная логика игры  
├── pubspec.yaml # Конфигурация Flutter проекта  
├── README.md # Документация проекта  
├── ИНСТРУКЦИЯ.md # Инструкция по запуску  
├── demo.dart # Демонстрация возможностей  
├── run.sh # Скрипт запуска (macOS/Linux)  
├── run.bat # Скрипт запуска (Windows)  
└── .gitignore # Игнорируемые файлы Git

### ТЕХНИЧЕСКИЕ ДЕТАЛИ

#### Используемые технологии:

* **Язык программирования**: Dart 3.0.0+
* **Фреймворк**: Flutter 3.10.0+
* **Тип приложения**: Консольное приложение
* **Система контроля версий**: Git

#### Ключевые классы и методы:

* **Класс TicTacToeGame**: основной класс игры
* **Метод startGame()**: запуск игры и главное меню
* **Метод setupGame()**: настройка игры и инициализация поля
* **Метод playGame()**: основной игровой цикл
* **Метод makeRobotMove()**: логика ИИ робота
* **Метод checkWinner()**: проверка условий победы

#### Особенности реализации:

* Использование late переменных для отложенной инициализации
* Обработка пользовательского ввода с валидацией
* Модульная архитектура с разделением логики
* Кроссплатформенность (macOS, Linux, Windows)

### ИНСТРУКЦИЯ ПО ЗАПУСКУ

#### Способ 1 - Через скрипт (рекомендуется):

# На macOS/Linux:  
./run.sh  
  
# На Windows:  
run.bat

#### Способ 2 - Напрямую через Dart:

dart run lib/main.dart

#### Способ 3 - Демонстрация возможностей:

dart run demo.dart

### ТЕСТИРОВАНИЕ

#### Проведенные тесты:

1. **Тест базового функционала**: игра 3x3 против друга
2. **Тест ИИ**: игра против робота на поле 3x3
3. **Тест больших полей**: игра на поле 5x5
4. **Тест валидации**: проверка обработки некорректного ввода
5. **Тест граничных случаев**: ничья, быстрая победа

#### Результаты тестирования:

* Все режимы игры работают корректно
* ИИ демонстрирует разумное поведение
* Валидация ввода работает правильно
* Интерфейс отображается корректно
* Обработка ошибок функционирует

### АНАЛИЗ ВЫПОЛНЕНИЯ ТРЕБОВАНИЙ

| Требование | Статус | Описание реализации |
| --- | --- | --- |
| Запрос размера поля | ✅ Выполнено | Пользователь вводит размер от 3 до 10 |
| Поочередные ходы | ✅ Выполнено | Игроки X и O ходят по очереди |
| Определение победителя | ✅ Выполнено | Проверка строк, столбцов, диагоналей |
| Случайный первый игрок | ✅ Выполнено | Используется Random.nextBool() |
| Новая игра | ✅ Выполнено | Меню выбора после завершения |
| Режим против друга | ✅ Выполнено | Два игрока поочередно делают ходы |
| Режим против робота | ✅ Выполнено | Игрок против ИИ |
| ИИ для робота | ✅ Выполнено | Стратегический ИИ с блокировкой |

### ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ

#### Реализованные улучшения:

1. **Красивый интерфейс**: использование эмодзи и форматирования
2. **Подробная документация**: README, инструкции, демо
3. **Кроссплатформенность**: скрипты для разных ОС
4. **Обработка ошибок**: валидация всех пользовательских вводов
5. **Модульность**: разделение кода на логические блоки
6. **Расширяемость**: легко добавить новые режимы игры

### ЗАКЛЮЧЕНИЕ

#### Достигнутые результаты:

* ✅ Все требования технического задания выполнены на 100%
* ✅ Игра демонстрирует высокий уровень программирования на Dart
* ✅ Реализован функциональный ИИ с разумной стратегией
* ✅ Код написан с соблюдением принципов чистого кода
* ✅ Проект готов к демонстрации и сдаче

#### Оценка сложности:

* **Базовый функционал**: средняя сложность
* **ИИ робота**: высокая сложность
* **Интерфейс и UX**: средняя сложность
* **Общая оценка**: соответствует требованиям на оценку 5

#### Рекомендации для развития:

1. Добавить сохранение статистики игр
2. Реализовать различные уровни сложности ИИ
3. Добавить звуковые эффекты
4. Создать графический интерфейс