Разработать графическое приложение с использованием технологий WPF или Windows Forms на выбор разработчика с последующим обоснованием выбора используемой технологии. Использование других технологий, совместимых с языком C# возможно по согласованию с преподавателем. Приложение должно реализовывать все описанные в варианте возможности и обладать достаточной производительностью для комфортного использования и демонстрации на доступной в учебных аудиториях технике.

**1. Модели**

* **Character.cs**: Класс для персонажей (игроков и противников), содержащий атрибуты, методы для атаки, защиты и использования способностей.
* **Player.cs**: наследует от Character, добавляет специфические для игрока атрибуты и методы (например, инвентарь, прокачка).
* **Enemy.cs**: наследует от Character, добавляет специфические для противников атрибуты и методы (например, способности).
* **Item.cs**: Класс для предметов, содержащий атрибуты и методы для использования.
* **Level.cs**: Класс для уровней, содержащий информацию о противниках, предметах и комнатах.
* **Cat.cs**: Класс для кошек, содержащий их характеристики и способности.

**2. Логика игры**

* **GameManager.cs**: Класс для управления основными процессами игры (инициализация, переход между уровнями, управление состоянием игры).
* **BattleSystem.cs**: Класс для обработки боевой механики (атаки, уклонения, использование способностей).
* **InventoryManager.cs**: Класс для управления инвентарем игрока (добавление, удаление предметов, использование).
* **LevelGenerator.cs**: Класс для генерации уровней и размещения противников и предметов.

**3. Пользовательский интерфейс**

* **MainForm.cs**: Основная форма приложения, содержащая элементы управления для взаимодействия с игроком.
* **BattleForm.cs**: Форма для отображения боя, включая информацию о противниках и действия игрока.
* **InventoryForm.cs**: Форма для отображения инвентаря игрока и управления предметами.
* **CatFeedingForm.cs**: Форма для взаимодействия с кошками и выбора предметов для кормления.

**4. Данные**

* **ItemsData.cs**: Файл или класс, содержащий информацию о всех предметах (например, их эффекты, стоимость).
* **EnemiesData.cs**: Файл или класс, содержащий информацию о всех противниках (например, здоровье, сила, способности).
* **LevelsData.cs**: Файл или класс, содержащий информацию о уровнях (например, расположение комнат, типы противников).

**5. Утилиты**

* **Utils.cs**: Вспомогательные функции, которые могут использоваться в разных частях игры (например, генерация случайных чисел, форматирование строк).
* **Constants.cs**: Константы, используемые в игре (например, максимальное здоровье, базовые значения атрибутов).

**6. Ресурсы**

* **Images/**: Папка для хранения изображений (персонажей, предметов, фонов).
* **Sounds/**: Папка для хранения звуковых эффектов и музыки.

**7. Тестирование**

* **Tests/**: Папка для хранения тестов (например, юнит-тесты для классов и методов).

**Последовательность разработки**

1. **Определение классов**: Начните с создания классов для персонажей, предметов и уровней.
2. **Реализация логики игры**: Создайте классы для управления процессами игры, боевой механики и инвентаря.
3. **Создание пользовательского интерфейса**: Разработайте формы для взаимодействия с игроком, отображения боя и управления инвентарем.
4. **Добавление графики**: начните добавлять графику, включая изображения персонажей, фонов и элементов интерфейса.
5. **Тестирование и отладка**: проведите тестирование, чтобы убедиться, что все работает корректно, и исправьте ошибки.
6. **Расширение функционала**: после реализации основных механик добавьте дополнительные функции, такие как сохранение прогресса и улучшение графики.