МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Объектно-ориентированное программирование»

Студент гр. 4384 Кочкин Д. М.

Преподаватель Жангиров Т. Р.

Санкт-Петербург

2025

Цель работы.

Изучение принципов объектно-ориентированного проектирования и реализация игровой логики на языке C++. В ходе работы создаются классы для представления сущностей игрового поля (игрока, врагов, зданий, клеток и самого поля) и реализуются их взаимодействия.

Задание.

Создать интерфейс карточки заклинания. Заклинание должно применяться игроком. На использование заклинания игрок тратит один ход.

Создать класс “руки” игрока, которая содержит все карточки заклинаний, которые игрок может применить в свой ход. Изначально рука игрока содержит только одно случайное заклинание. Реализовать возможность получать новые заклинание игроком, например, тратить очки на покупку или после уничтожения определенного кол-ва врагов. Размер “руки” должен быть ограничен и задается через конструктор.

Реализовать интерфейс заклинанием прямого урона. Это заклинание при использовании должно наносить урон врагу или вражескому зданию, если они находятся в достижимом радиусе. Если в качестве цели не выбран враг или вражеское здание, то заклинание не используется.

Реализовать интерфейс заклинания урона по площади. Это заклинание при использовании в допустимом радиусе наносит урон по области 2 на 2 клетки. Заклинание используется, даже если там нет никого.

Реализовать интерфейс заклинания ловушки. Заклинание размещает на поле ловушку, если враг наступает на клетку с ловушкой, то ему наносится урон, и ловушка пропадает.

Создать класс вражеской башни. Вражеская башня размещается на поле, и если в радиусе ее атаки появляется игрок, то применяет ослабленную версию заклинания прямого урона. Не может применять заклинание несколько ходов подряд.

Реализовать интерфейс заклинания призыва. Заклинание создает союзника рядом с игроком, который перемещается самостоятельно.

Реализовать интерфейс заклинание улучшения. Заклинание улучшает следующее используемое заклинание:

Заклинание прямого урона - увеличивает радиус применения

Заклинание урона по площади - увеличивает площадь

Заклинание ловушки - увеличивает урон

Заклинание призыва - призывает больше союзников

Заклинание улучшение - накапливает усиление, то есть при применении следующего заклинания отличного от улучшения, все улучшения применяются сразу

Описание архитектуры программы.

Программа построена по принципам объектно-ориентированного программирования с чётким разделением ответственности между классами. Центральным элементом является класс Field, отвечающий за состояние

игрового поля, размещение и обновление сущностей.

Основные классы:

• Entity — базовый абстрактный класс, определяющий общие свойства всех сущностей (идентификатор, здоровье, урон).

• Player — класс игрока, наследник Entity. Реализует перемещение, режимы боя (ближний/дальний), взаимодействие с замедляющими клетками и врагами.

• Enemy — противник, движущийся к игроку. Если враг пытается шагнуть на клетку с игроком, он не перемещается, но наносит урон.

• Building — структура, периодически создающая врагов.

• Field — класс, содержащий сетку клеток, список сущностей и их координаты. Реализует механику игры: обновление, движение, спавн врагов и проверку конца игры.

• Cell — элемент поля, хранящий тип содержимого (пустая, непроходимая, замедляющая и т.д.).

Дополнительные классы:

Tower — специализированный тип здания, наследник Building. В отличие от обычных зданий не создает врагов, а атакует игрока в определенном радиусе, нанося уменьшенный урон.

Ally — дружественная сущность, наследник Entity. Автономно перемещается по полю и атакует врагов.

Trap — статическая сущность, наследник Entity. При попадании врага на клетку с ловушкой наносит урон и исчезает.

Система заклинаний:

ISpell — интерфейс, определяющий базовое поведение всех заклинаний (применение, клонирование, получение имени).

DirectDamageSpell — заклинание прямого урона по одной цели в указанном радиусе.

AreaDamageSpell — заклинание урона по области размером 2x2 клетки.

TrapSpell — заклинание, создающее ловушку на поле.

SummonSpell — заклинание призыва союзника.

BuffSpell — заклинание, временно усиливающее характеристики других заклинаний.

Вспомогательные компоненты:

Hand — класс, представляющий "руку" игрока с заклинаниями. Управляет коллекцией заклинаний, их добавлением и удалением.

CombatSystem — статический класс, отвечающий за боевую механику:

Обработка нанесения урона

Обработка смерти сущностей

Подсчет убийств и выдача наград игроку

Связи классов:

Иерархия сущностей (Entity):

Player, Enemy, Building, Ally, Trap наследуют базовый класс Entity

Tower наследует Building

Иерархия заклинаний (ISpell):

DirectDamageSpell, AreaDamageSpell, TrapSpell, SummonSpell, BuffSpell реализуют интерфейс ISpell

Композиция:

Game содержит Field

Field содержит коллекции Cell и Entity

Player содержит Hand

Hand содержит коллекцию ISpell

Зависимости:

CombatSystem использует Field и Player для реализации боевой механики

Spell классы используют Field для применения эффектов

Tower использует Field для атаки игрока

UML-диаграмма классов.

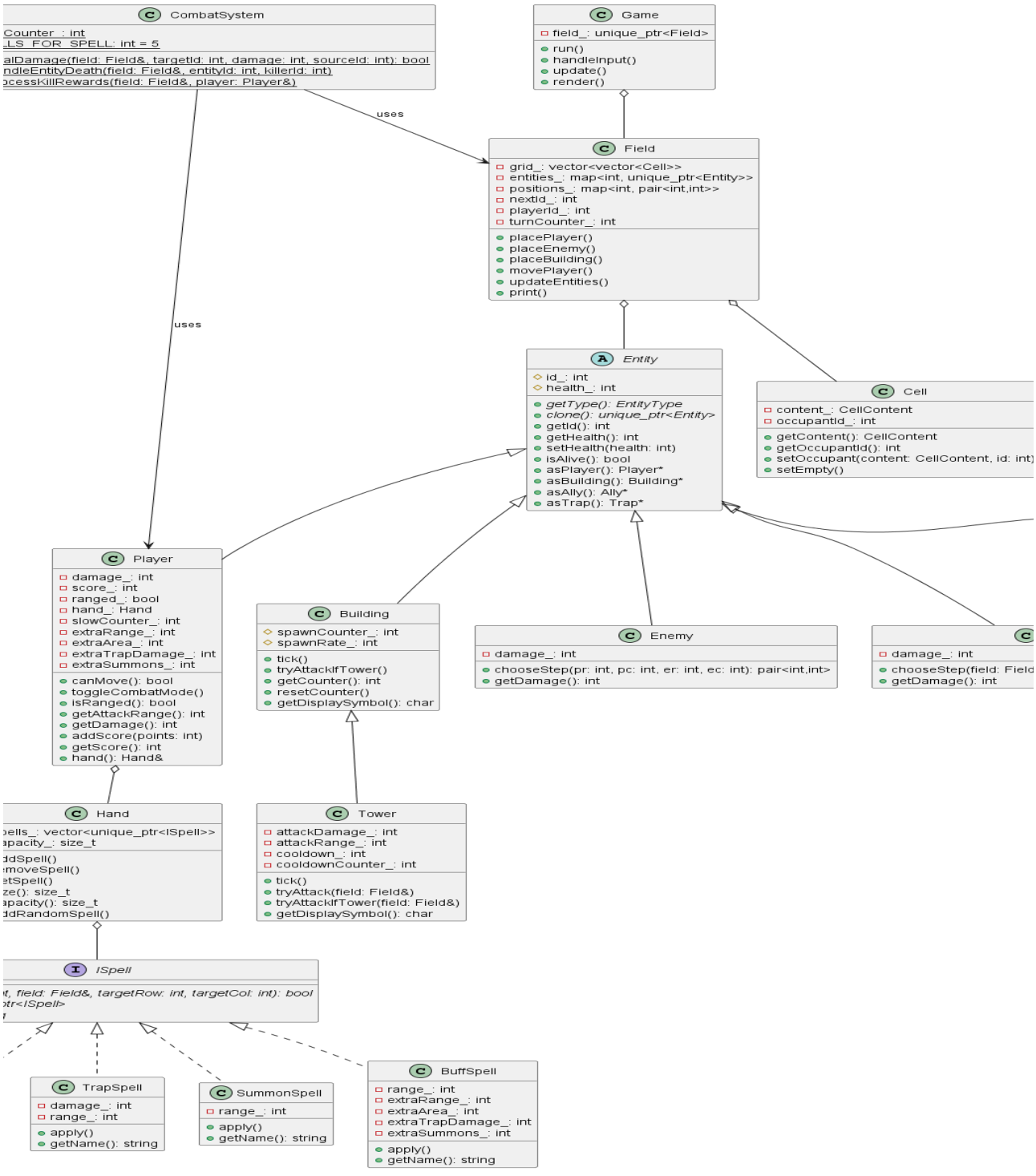

Рисунок 1 - UML-диаграмма

Проверка работы программы.

Была проведена проверка работы игры. Игрок может перемещаться по полю, сталкиваться с врагами и зданиями, а также переключаться между ближним и дальним боем. При переходе на клетку с врагом он атакует противника, но теряет возможность хода на следующий раунд. При переходе на замедляющую клетку — пропускает ход. Враги корректно двигаются к игроку и атакуют при попытке занять его клетку. Создана механика заклинаний. Реализована возможность возможность добавить союзника, атакующего врагов. Добавлен класс башни, которая применяет заклинания и наносит урон игроку, который стоит рядом с ней.

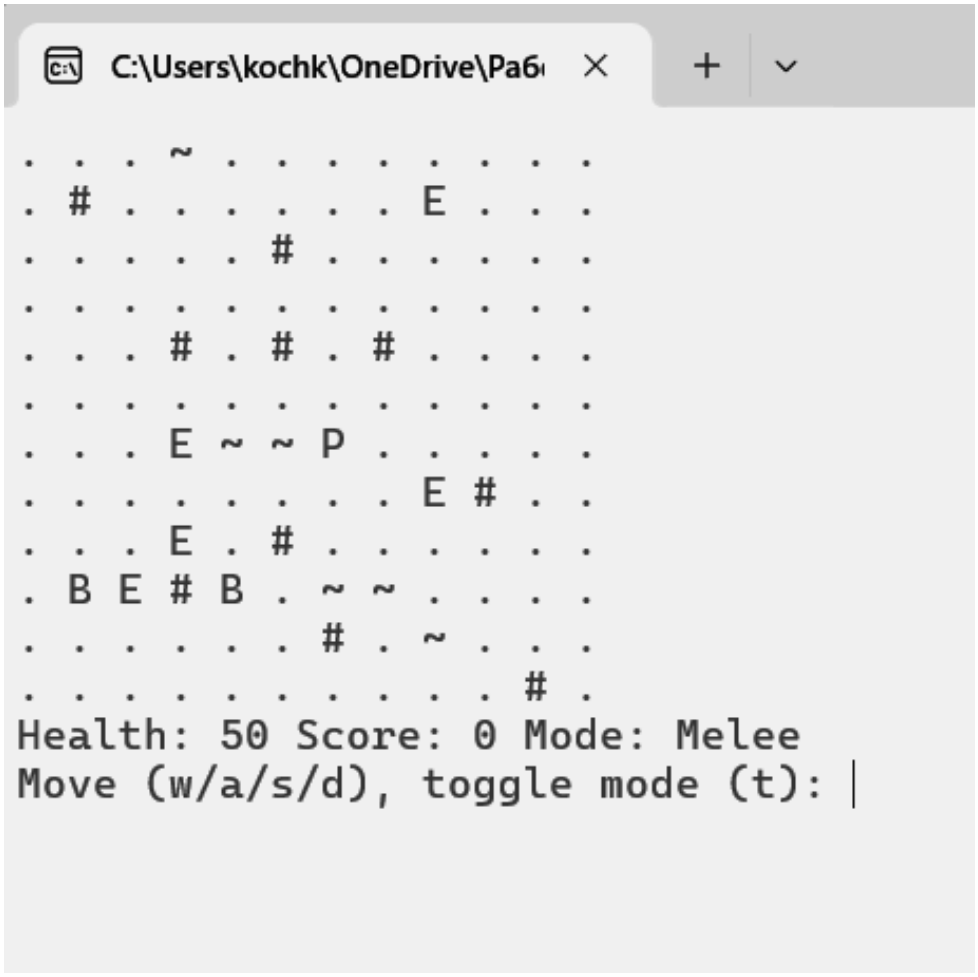

Рисунок 2 - Игровое поле

Выводы.

В результате выполнения лабораторной работы были изучены принципы ООП: наследование, инкапсуляция и полиморфизм. Разработана архитектура игры с несколькими типами сущностей и взаимодействиями между ними. Программа успешно проходит тестирование и демонстрирует корректное поведение объектов на игровом поле.