МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «ООП»

ТЕМА: СОЗДАНИЕ КЛАССОВ, КОНСТРУКТОРОВ И МЕТОДОВ

Студентка гр. 1383 Седова Э.А

Преподаватель Жангиров Т.Р.

Санкт-Петербург

2022

Цель работы.

Создать игру на C++, состоящую из нескольких классов, понять, как взаимодействуют классы друг с другом.

Задание.

Реализовать прямоугольное игровое поле, состоящее из клеток. Клетка - элемент поля, которая может быть проходима или нет (определяет, куда может стать игрок), а также содержит какое-либо событие, которое срабатывает, когда игрок становится на клетку. Для игрового поля при создании должна быть возможность установить размер (количество клеток по вертикали и горизонтали). Игровое поле должно быть зациклено по вертикали и горизонтали, то есть если игрок находится на правой границе и идет вправо, то он оказывается на левой границе (аналогично для всех краев поля).

Реализовать класс игрока. Игрок - сущность контролируемая пользователем. Игрок должен иметь свой набор характеристик и различный набор действий (например, разные способы перемещения, попытка избежать событие, и так далее).

Требования:

- Реализован класс игрового поля
- Для игрового поля реализован конструктор с возможностью задать размер и конструктор по умолчанию (то есть конструктор, который можно вызвать без аргументов)
- Реализован класс интерфейс события (в данной лабораторной это может быть пустой абстрактный класс)
- Реализован класс клетки с конструктором, позволяющим задать ей начальные параметры.
- Для клетки реализованы методы реагирования на то, что игрок перешел на клетку.

- Для клетки реализованы методы, позволяющие заменять событие. (То есть клетка в ходе игры может динамически меняться)
- Реализованы конструкторы копирования и перемещения, и соответствующие им операторы присваивания для игрового поля и при необходимости клетки
- Реализован класс игрока минимум с 3 характеристиками. И соответствующие ему конструкторы.
- Реализовано перемещение игрока по полю с проверкой допустимости на переход по клеткам.

Примечания:

- При написании конструкторов учитывайте, что события должны храниться по указателю для соблюдения полиморфизма
- Для управления игроком можно использовать медиатор, команду, цепочку обязанностей

Выполнение работы.

В классе *Cell* содержится необходимые поля для логики клетки: *playerHere* (находится ли игрок здесь), *available* (может ли игрок сюда перейти), *event* (для хранения событий). Также реализованы методы для того, чтобы получить или задать эти значения.

Класс *CommandReader* имеет метод *readCommand* для считывания слова из консоли.

Класс *Field* содержит в себе двумерный массив с помощью вектора (*vector*), хранит width и height – ширина и высота, и положение игрока - *posX* и *posY*. Есть соответствующие геттеры и сеттеры, а также возможность обновить информацию о конкретной клетке – методы *updateAvailable* и *updatePlayerPos*.

Класс *FieldView* содержит единственный метод *printField* для рисования поля.

Класс *Player* содержит такие поля, как количество жизней (*healthPoint*), урон (*damage*), защита (*armor*) и имя (*name*), и к ним – соответствующие геттеры и сеттеры.

Класс *Controller* создает взаимодействие с классами *Player*, *Field*, *FieldView*. Он содержит метод для вывода поля на консоль (*printField*), метод для перемещения (*move*) и метод для проверки на возможность перехода на следующее поле (*canMove*).

Класс *Mediator* связывает *Controller* и *CommandReader*, вызывая нужные методы класса *Controller* на основе данных, которые ввел пользователь.

Тестирование программы.

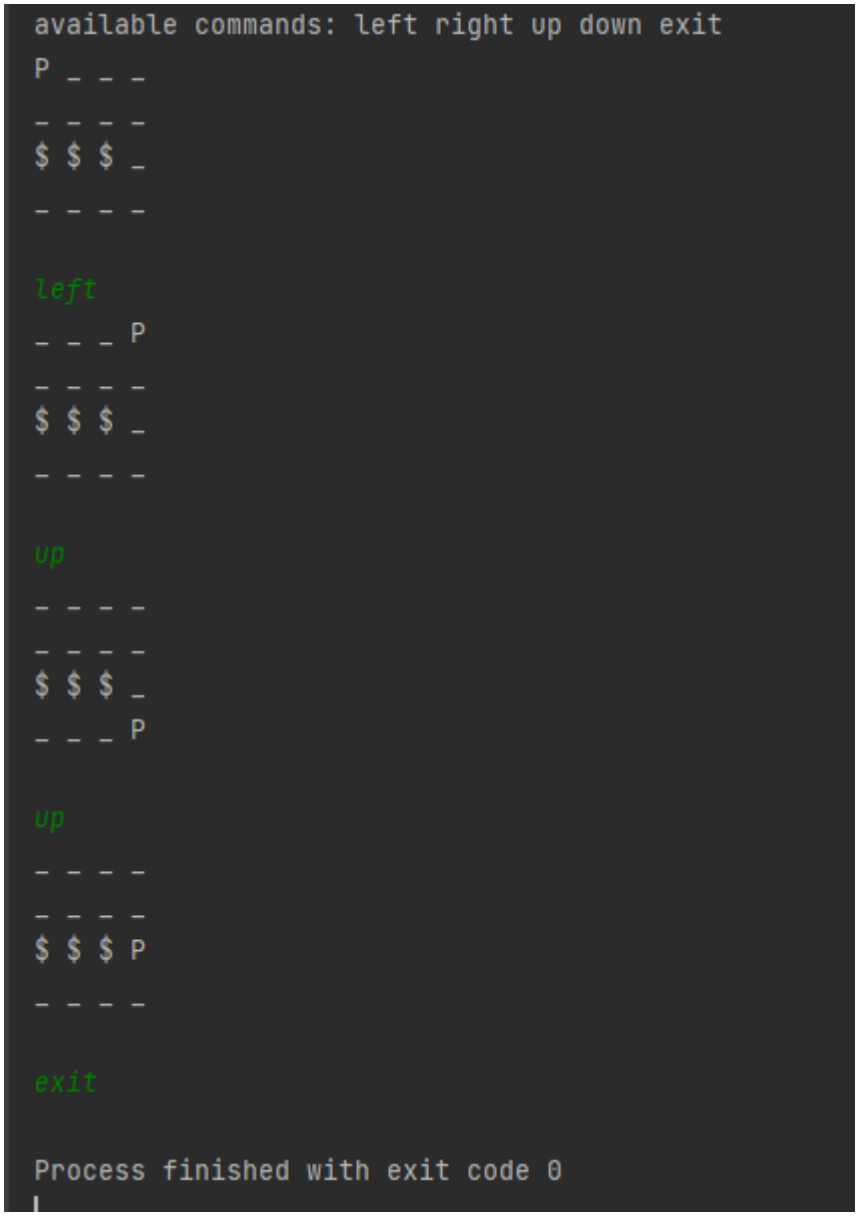

Рисунок 1. – Тестирование программы

UML-диаграмма межклассовых отношений.

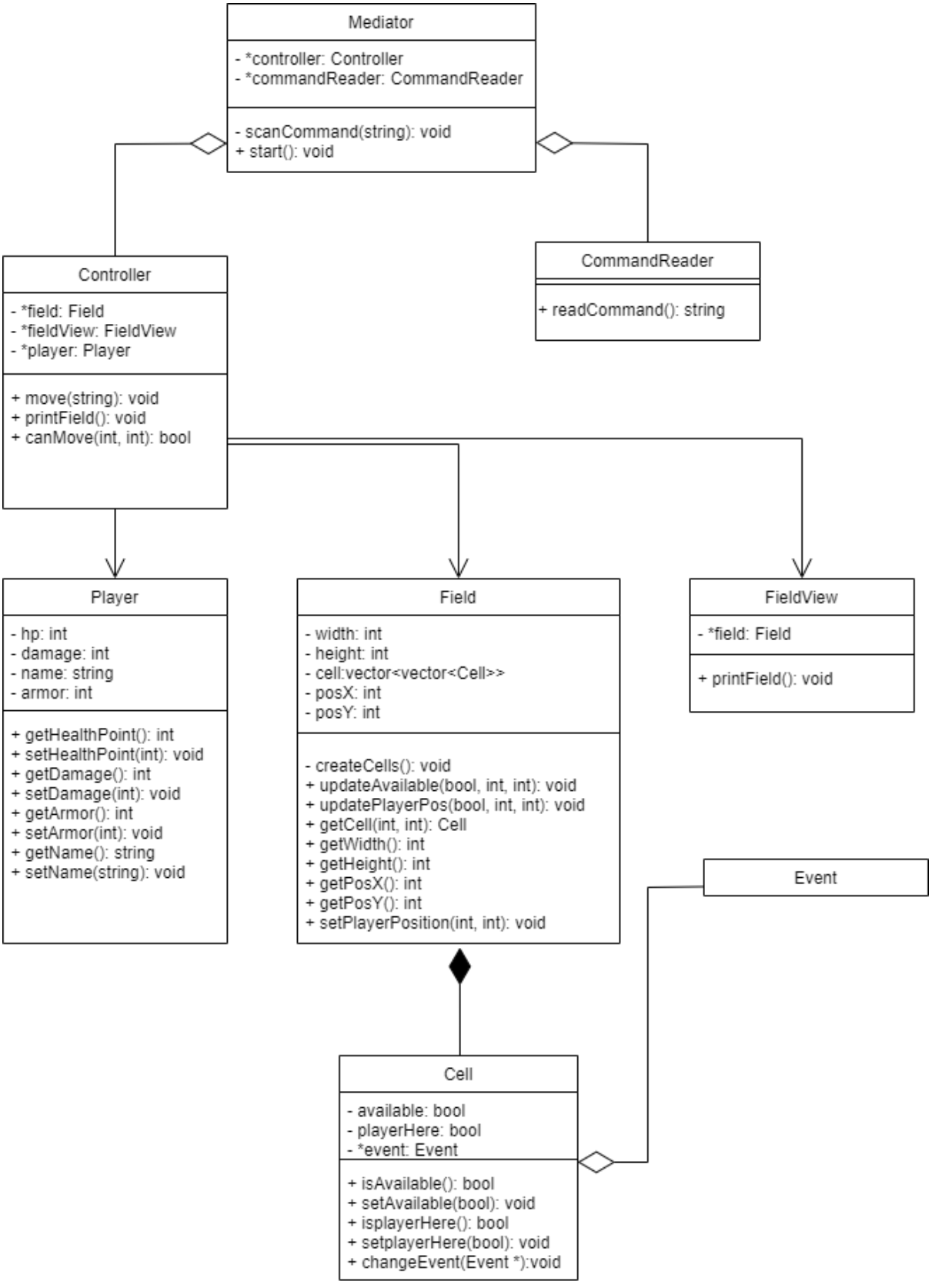

Рисунок 2. – UML Диаграмма классов

Выводы.

В ходе выполнения работы была создана игра, в которой было сделано взаимодействие между классами.