МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2

по дисциплине «ООП»

Тема: Интерфейсы, динамический полиморфизм

Студентка гр. 1383 Седова Э.А

Преподаватель Жангиров Т.Р.

Санкт-Петербург

2022

Цель работы.

Создать класс интерфейс, задающий события, влияющие на игрока и поле.

Задание.

Реализовать систему событий. Событие - сущность, которая срабатывает при взаимодействии с игроком. Должен быть разработан класс интерфейс общий для всех событий, поддерживающий взаимодействие с игроком. Необходимо создать несколько групп разных событий реализуя унаследованные от интерфейса события (например, враг, который проверяет условие, будет ли воздействовать на игрока или нет; ловушка, которая безусловно воздействует на игрока; событие , которое меняет карту; и.т.д.). Для каждой группы реализовать конкретные события, которые по разному воздействуют на игрока (например, какое-то событие заставляет передвинуться игрока в определенную сторону, а другое меняет характеристики игрока). Также, необходимо предусмотреть событие “Победа/Выход”, которое срабатывает при соблюдении определенного набора условий.

Реализовать ситуацию проигрыша (например, потери всего здоровья игрока) и выигрыша игрока (добрался и активировал событие “Победа/Выход”)

Выполнение работы.

Создан класс интерфейс Event, содержащий одну виртуальную функцию make_event(Player& player, Cell& cell). Его наследуют два класса: EventPlayer, EventField. Первый отвечает за события, влияющие на игрока, второй – на игровое поле.

Класс EventPlayer наследуют классы: Heal, fire, Level_exit, Key.

- В классе Heal повышается здоровье (поле игрока healtgPoint) на 50 единиц (но не больше ста единиц).
- В классе fire сначала разрушается броня (поле игрока Armor) на 50 единиц (но не менее нуля единиц). Затем, когда броня разрушается полностью, начинает понижаться здоровье игрока на 50 единиц (но не менее нуля единиц). Когда healthPoint становится меньше или равным нулю игра завершается проигрышем.

- В классе Key меняется поле игрока Key. Его значение меняется с false на true.
- В классе Level_exit проверяется значение поля игрока Key, с помощью метода getKey() класса Player. Если его значение false, то игрок не может выйти с уровня. Если его значение true, то игрок выходит с уровня и игра заканчивается победой.

Класс EventField наследуют классы: Clear_cell, Wall.

- Экземпляр класс Clear_cell ничего не меняет. Он записывается в пустые клетки.
- В классе Wall реализован метод “разрушения стен”. Он меняет непроходимые клетки на проходимые. Эта манипуляция производится с помощью метода setAvailable() класса Cell, который меняет поле клетки available.

Во всех классах событий (кроме Level_exit) после метода, make_event вызывается метод clear() класса Cell, который “зачищает” поле event клетки после срабатывания события и присваивает ему событие clear_cell. Также в нем меняется значение поля number класса Cell. По значению number задаётся определённое событие в конкретную клетку. Каждому числу соответствует класс события: Wall(0), fire(1), Heal(2), Clear_cell(3), Key(4), Level_exit(5).

Тестирование программы.

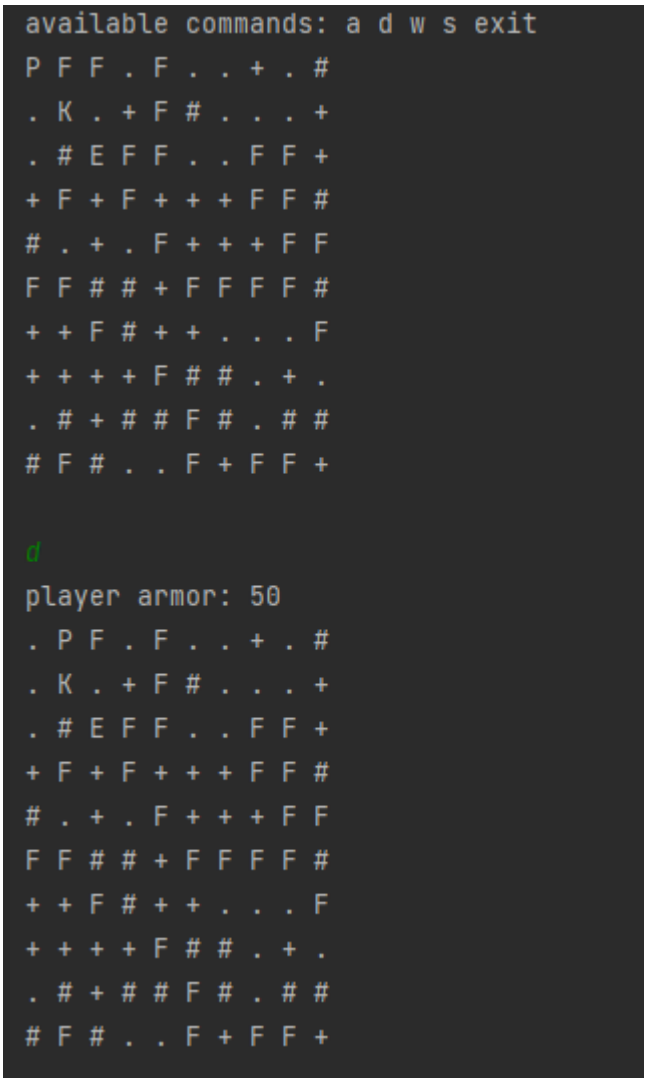

Рисунок 1. – Тестирование программы

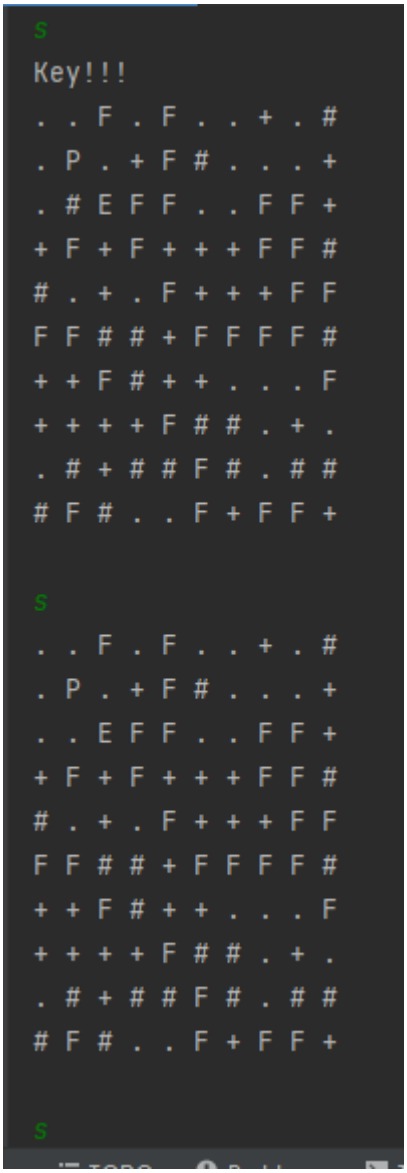

Рисунок 2. – Тестирование программы

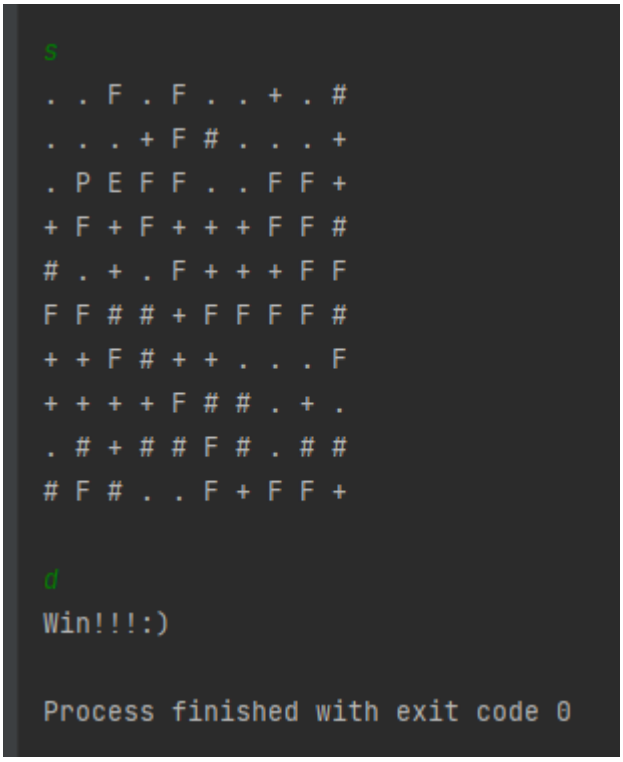

Рисунок 3. – Тестирование программы

UML-диаграмма межклассовых отношений.

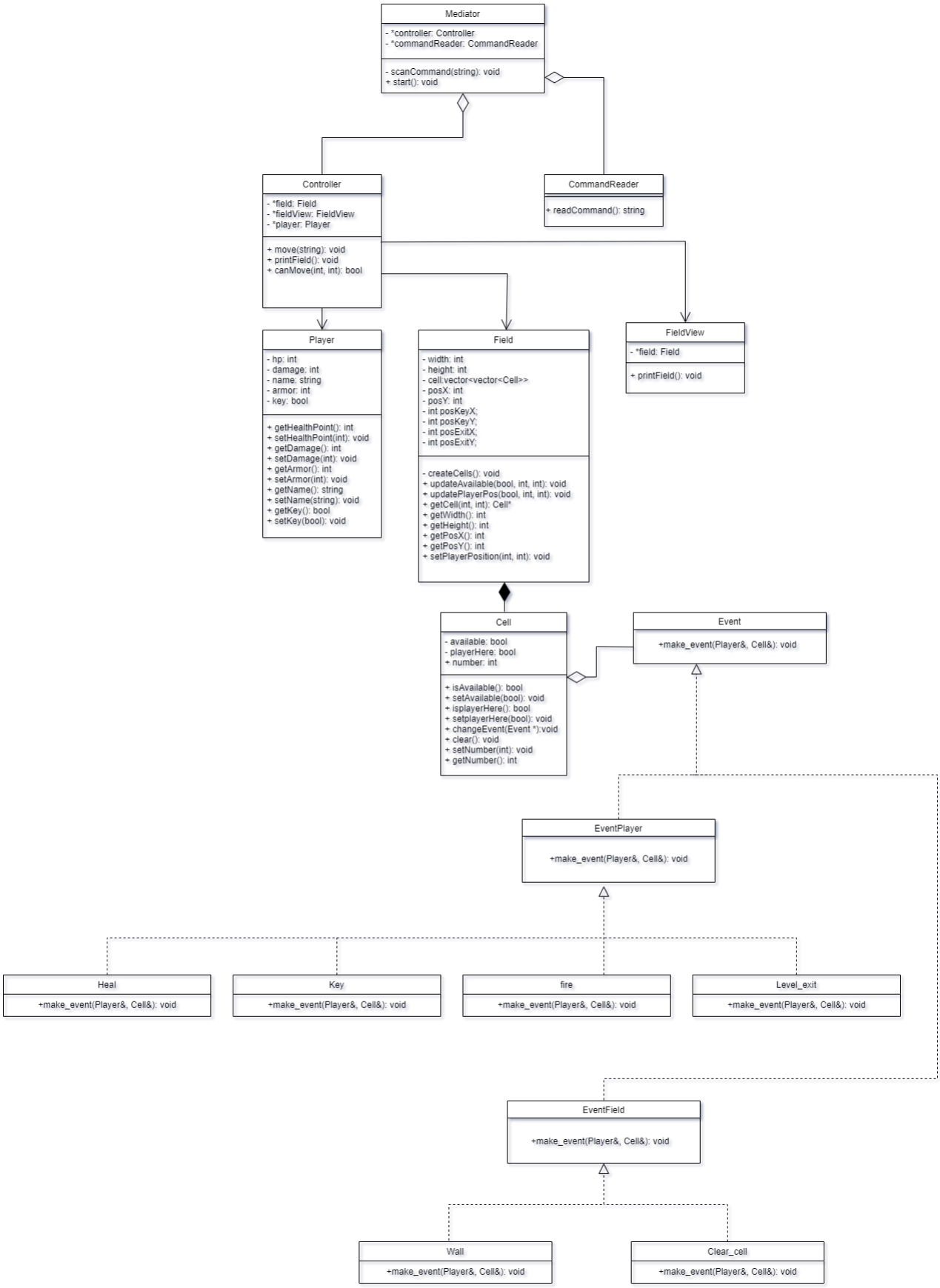

Рисунок 4. – UML Диаграмма классов

Выводы.

В ходе выполнения работы был создан интерфейс, задающий события, влияющие на игрока и поле.