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

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Объектно-ориентированное программирование»

Тема: **Создание классов, конструкторов и методов классов.**

Студент гр. 1384 _____ Алиев Д. А.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург
2022

Цель работы.

Изучить основы объектно-ориентированного программирования, научиться реализовывать простые классы и связывать их между собой.

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.

Примечание:

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

Выполнение работы.

Для выполнения лабораторной работы были созданы классы, отвечающие за игрока, создание клетки поля, создания поля, их вывод и взаимодействие пользователя с программой. Было решено создать классы для игровой логики – Field, Cell, Player. Чтобы нарисовать объекты этих классов необходимо было создать классы, отвечающие за из внешний вид: FieldView, CellView. Считывание и исполнение команд пользователя были реализованы в классах CommandReader и Controller, сообщение между которыми было осуществлено через класс Mediator, что позволяет им общаться не знаю друг о друге.

Описание классов:

1. Field – класс поля, хранящий в себе размер поля, само поле, представляющее из себя двумерный вектор, элементами которого являются клетки и позицию игрока. Для поля реализованы конструкторы и операторы копирования и перемещения, а также конструктор по умолчанию.
2. Cell – класс клетки, объёкт которого хранит в себе проходимость и событие. Является составной частью класса Field.

3. Entity – класс родитель для всех существ, задающий общие базовые характеристики, такие как : здоровье, вес, урон, максимальное здоровье, максимальный вес.

4. Player – класс игрока, наследник класса Entity, имеет конструктор, задающий ему начальные характеристики.

5. CellView – класс отрисовки клетки, определяющий её символ в зависимости от характеристики (например проходимость).

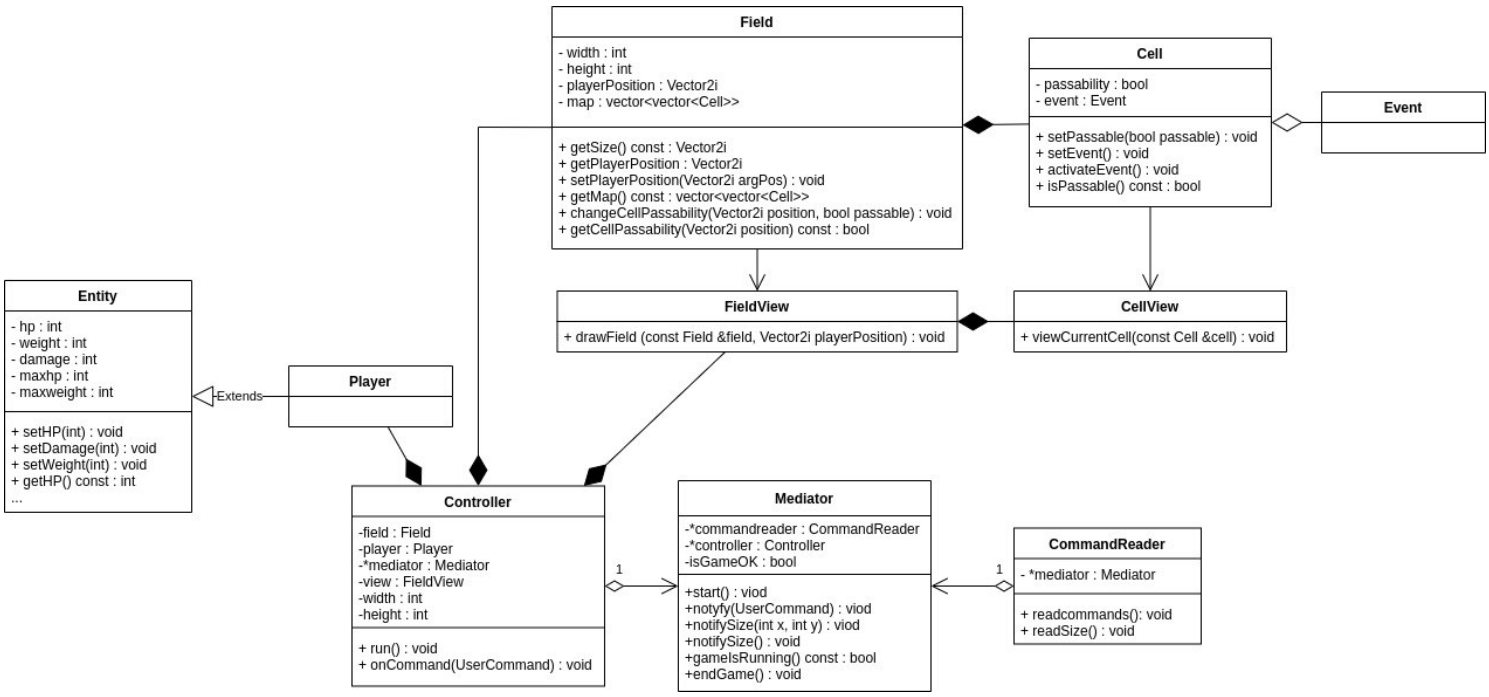
6. FieldView – класс отрисовки поля и игрока (хранит его кординаты). Отрисовка клеток осуществляется посредством вызова CellView.

7. CommandReader – класс считывания команд с клавиатуры. Считывает размер, заданный пользователем и команды данные игровому персонажу.

8. Controller – класс, отвечающий за инициализацию игрового поля, игрока и класса отрисовки. Обрабатывает команды полученные из CommandReader и вызывает FieldView.

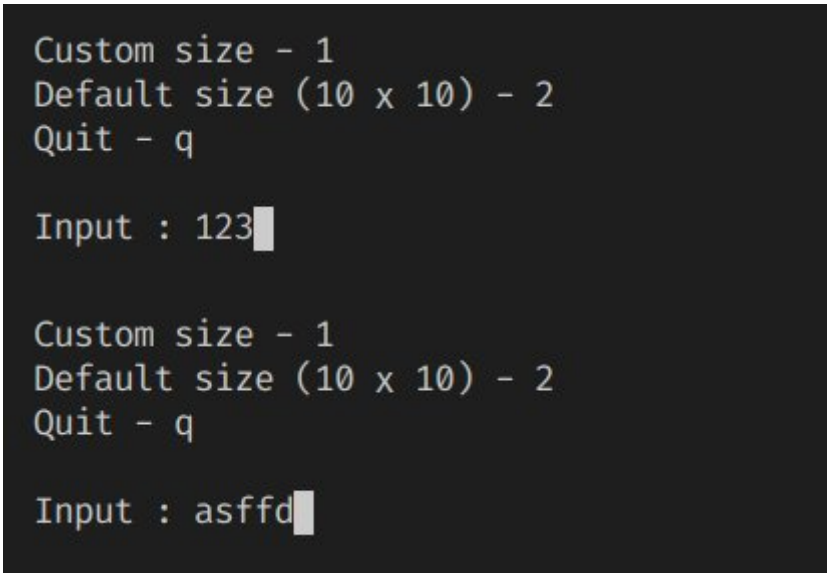
9. Mediator – класс посредник, организующий общение между классами CommandReader и Controller, который также отвечает за отслеживанием состояния игры (окончена или нет).

UML-диаграмма межклассовых отношений:

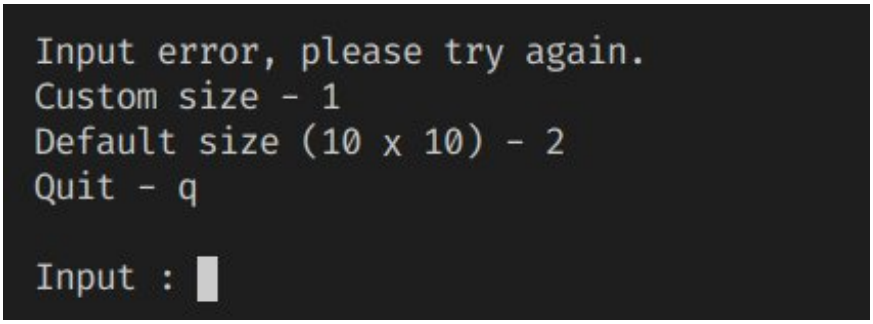


Тестирование программы :

Проверка корректности ввода :

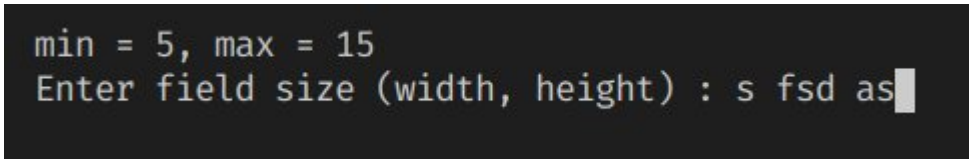


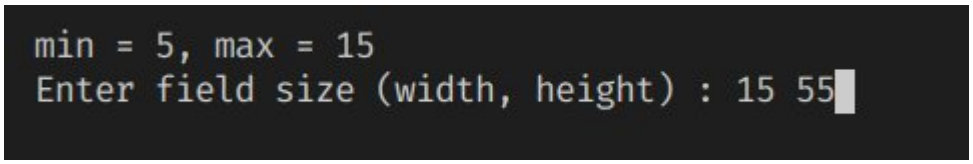
– не пройдут



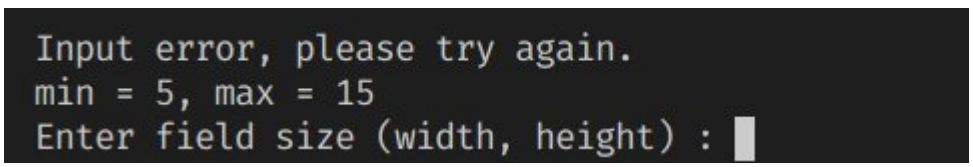
– ошибка, повторный ввод

Custom size :





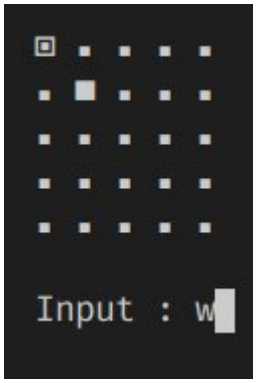
– не пройдут



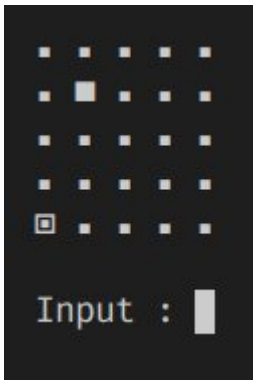
– ошибка, повторный ввод

Цикличность :

До

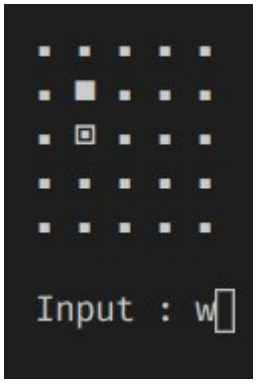


После

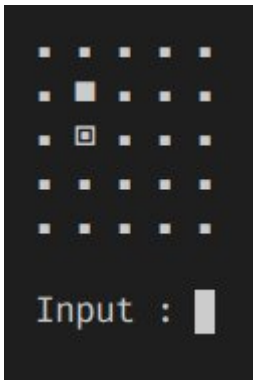


Проходимость :

До



После



Вывод:

По результатам лабораторной работы были изучены основные принципы объектно-ориентированного программирования; были реализованы классы со своими методами и полями, отвечающие определенному логическому модулю и в целом образующие единую систему.