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

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Объектно-ориентированное программирование»
Тема: Добавление игрока и элементов для поля

Студентка гр. 9381 _____ Москаленко Е.М.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2020

Цель работы.

Создать иерархию классов объекта игрока и трёх разных элемента поля, унаследованных от базового абстрактного класса (интерфейса) и по-разному взаимодействующих с игроком.

Задание.

Создан класс игрока, которым управляет пользователь. Объект класса игрока может перемещаться по полю, а также взаимодействовать с элементами поля. Для элементов поля должен быть создан общий интерфейс и должны быть реализованы 3 разных класса элементов, которые по разному взаимодействуют с игроком. Для взаимодействия игрока с элементом должен использоваться перегруженный оператор *(Например, оператор +)*. Элементы поля могут добавлять очки игроку/замедлять передвижения/и.т.д.

Обязательные требования:

Реализован класс игрока

Реализованы три класса элементов поля

Объект класса игрока появляется на клетке со входом

Уровень считается пройденным, когда объект класса игрока оказывается на клетке с выходом (и при определенных условиях: например, набрано необходимое кол-во очков)

Взаимодействие с элементами происходит через общий интерфейс

Взаимодействие игрока с элементами происходит через перегруженный оператор

Дополнительные требования:

Для создания элементов используется паттерн **Фабричный метод/Абстрактная фабрика**

Реализовано динамическое изменение взаимодействия игрока с элементами через паттерн **Стратегия**. Например, при взаимодействии с определенным количеством элементов, игрок не может больше с ними взаимодействовать

Ход работы.

Написание работы происходило в среде разработки QtCreator с использованием фреймворка Qt.

Для создания объектов игрового поля был реализован класс Player, приватными полями которого являются coins – количество монет, win – сведения о победе, coords – координаты игрока на данный момент и lives – количество оставшихся жизней. Помимо функций, изменяющих и возвращающих значения полей объекта, был перегружен оператор +=, аргументом которого является любой объект поля. Цель оператора – выполнить взаимодействие игрока и объекта.

В программе содержится 4 различных класса объектов, унаследованных от абстрактного класса CellObject:

CoinObject – монетка, которые игрок должен собирать. Необходимо собрать минимум 2 монетки для окончания игры.

ExitObject – объект выхода. Игрок не сможет выйти, пока не соберет нужное количество монеток и не сохранит жизни.

WallObject – стена или же камень. Игрок теряет жизни, если врезается в нее.

VortexObject – воронка, переносящая игрока в рандомную клетку поля.

Объекты создаются в конкретных клетках при инициализации самого поля.

Управление игроком происходит с помощью обработки нажатия клавиш на клавиатуре. Для этого в классе графического интерфейса MainWindow перегружен метод keyPressEvent(). Обязательна латинская раскладка.

W - вверх

A - влево

S - вниз

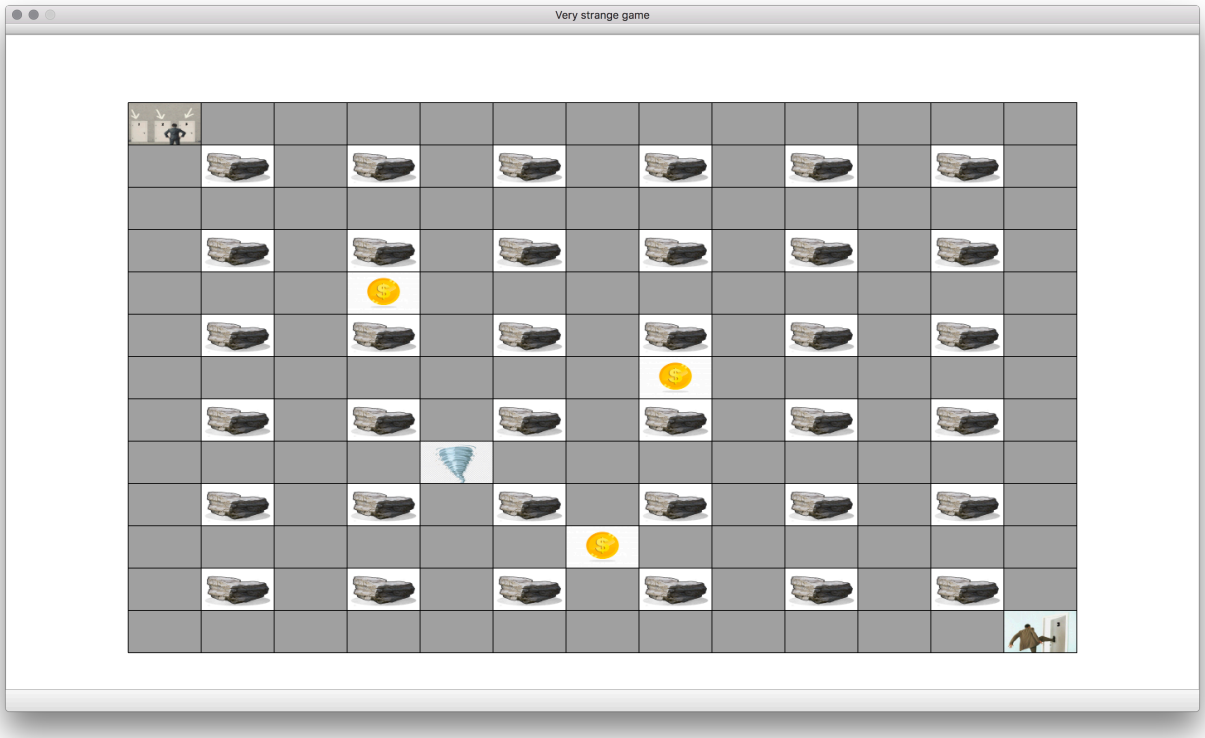
D – вправо

Связь между MainWindow и Player осуществляет объект класса Controller. Контроллер содержит игрока и поле как приватные поля. При обработки нажатий клавиш MainWindow подает команду в Controller, который вызывает соответствующие методы объекта класса Player и возвращает в MainWindow новые координаты игрока.

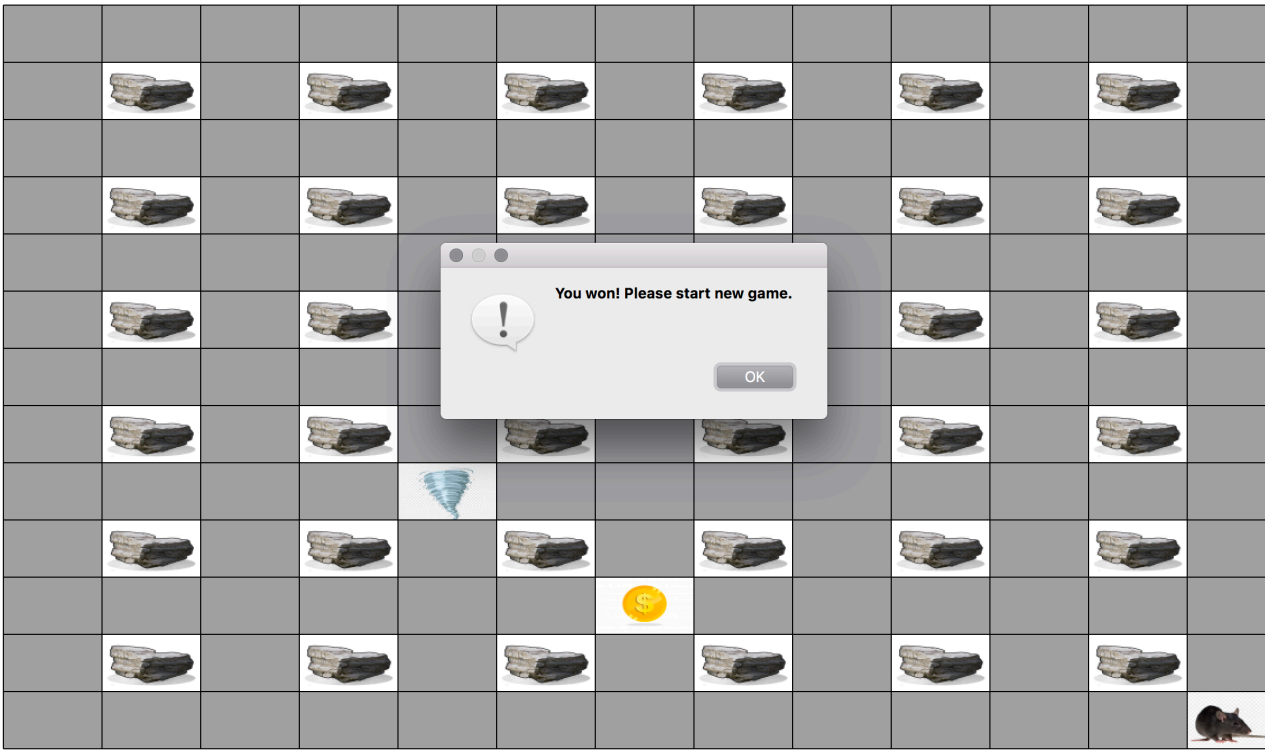
Также был реализован класс итератора Field::Iterator для обхода поля.

Тестирование.

Программа успешно запускается.



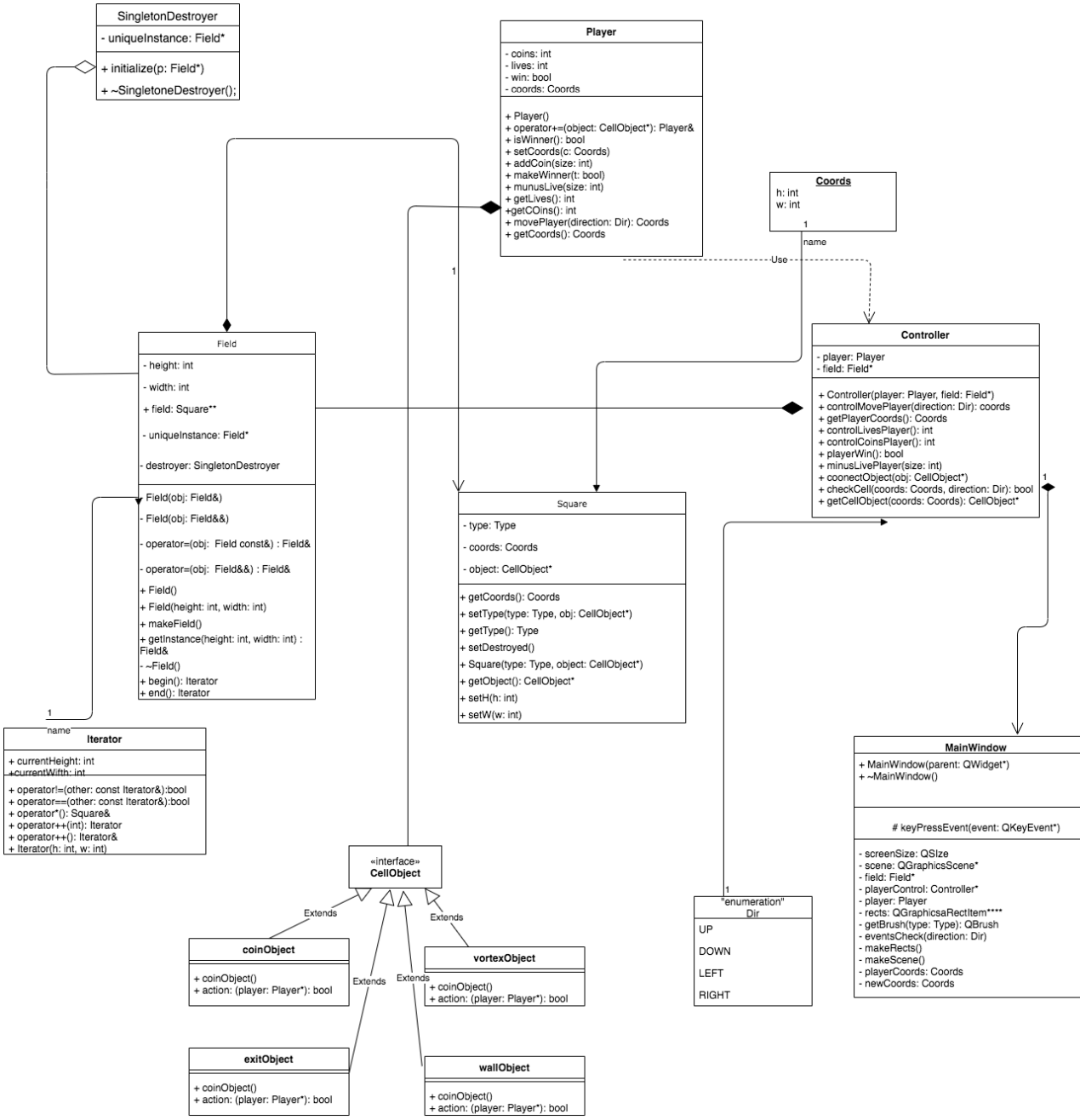
Взаимодействие с клеткой выхода.



Взаимодействие со стеной.

Взаимодействие с монеткой.

Uml-диаграмма.



Выводы.

Был реализован класс игрока, базовый абстрактный класс объекта игры и 4 унаследованных от него класса объектов. Для всех классов созданы все необходимые конструкторы, операторы и методы. Реализован GUI с помощью фреймворка Qt, связь между которым с бизнес-логикой программы реализует класс контроллера.