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: Добавления игрока и элементов для поля

Студент гр. 9381 _____ Камакин Д.В.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2020

Цель работы.

Изучить синтаксис языка программирования C++, ознакомиться с основными принципами объектно-ориентированного программирования. Реализовать класс игрока и элементов поля, также написать GUI.

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.

Выполнение работы.

Написание работы производилось на базе операционной системы Ubuntu 20.04 в среде разработки QtCreator с использованием фреймворка Qt.

Для игрового поля был реализован класс GameField. Написаны конструкторы копирования и перемещения, а также операторы присваивания и перемещения. Класс содержит двумерный массив клеток собственного класса GameCell, а также переменную и соответствующие функции для реализации паттерна «Одиночка».

Класс GameCell служит для реализации элемента на игровом поле. Каждая клетка содержит поля для хранения её типа, информацию о положении и занятости объектом. Реализованы функции для получения текущего состояния клетки.

Для создания GUI был использован фреймворк Qt, написан класс MainWindow, объект которого отображается в функции main(). В menuBar был добавлен пункт информации об авторе программы. Поле отображается при помощи QGraphicsView, сцена QGraphicsScene которого содержит клетки QGraphicsRectItem, внешний вид которых зависит от типа элемента на игровом поле GameField. Размер игрового поля 20X20.

Кроме того, для удобства работы с координатами был написан класс Point2D, в котором определены функции для получения и установки координат.

Обработка нажатий реализована в классе MainWindow. Поддерживается только **латинская** раскладка.

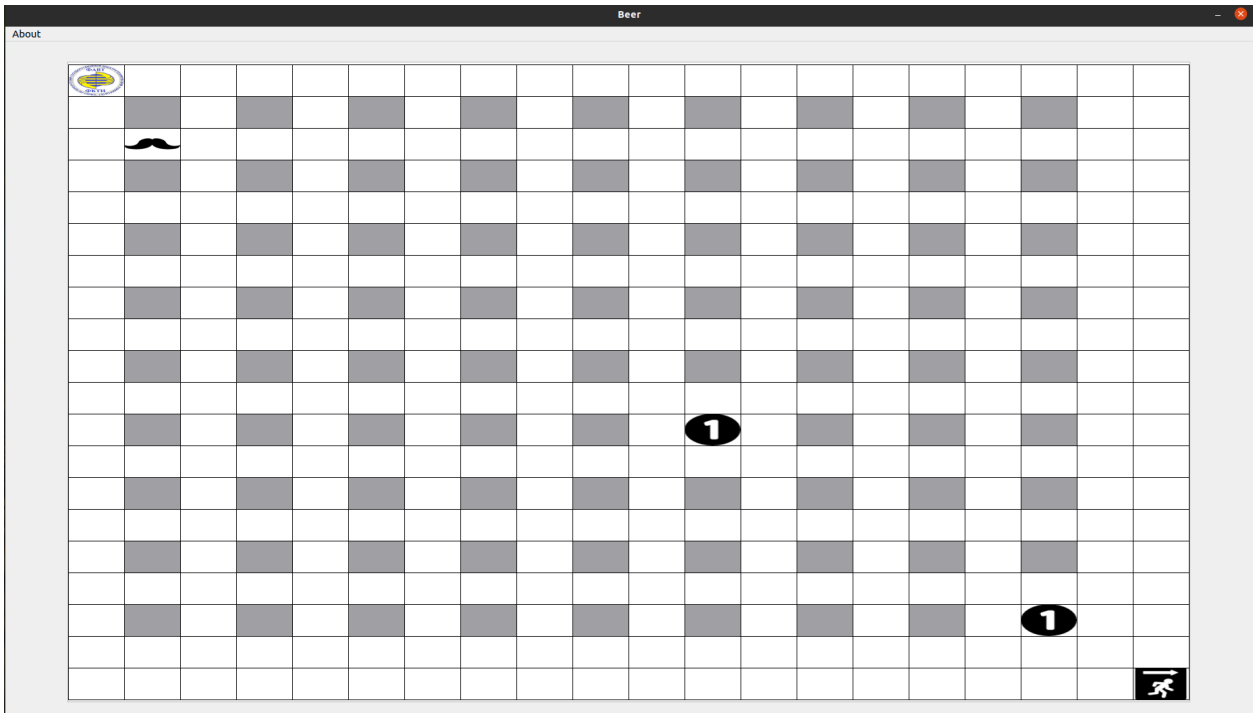
После получения команды от пользователя, направление движение отправляется в класс GameController, который перемещает игрока по полю, возвращая новую координату расположения персонажа.

Объекты на поле наследуются от интерфейса GameObject, в котором объявлена виртуальная функция action. Взаимодействие героя с объектами реализовано при помощи перегруженного оператора «+»

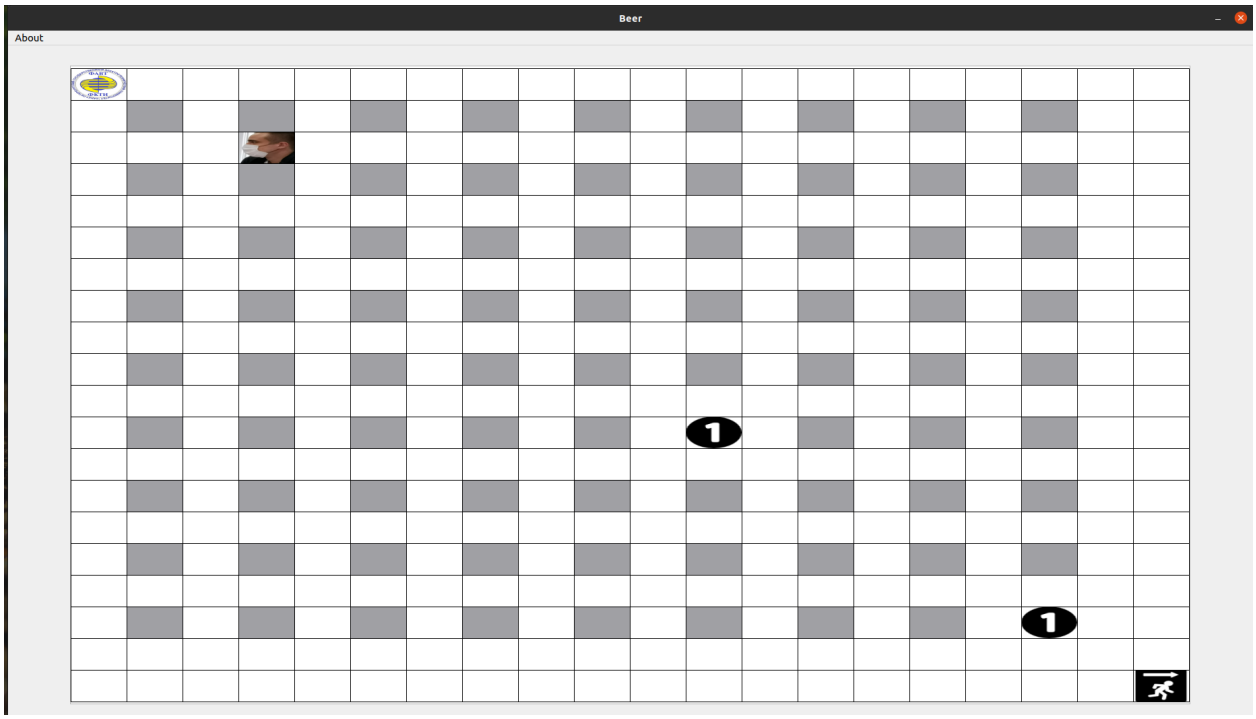
UML диаграмма представлена в файле uml.png.

Тестирование.

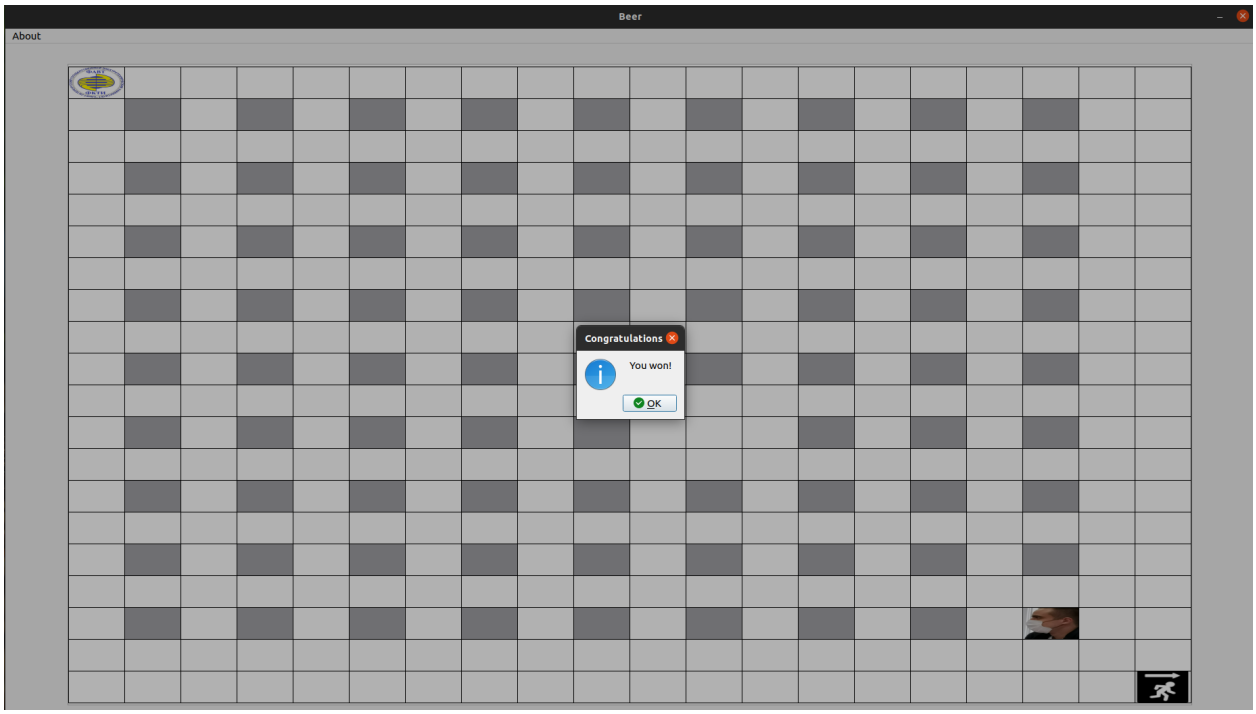
1. Программа успешно запускается, поле корректно отображается.



2. Персонаж перемещается по полю, взаимодействуя с объектами.



3. При сборе всех монет игра завершается



Выводы.

Изучен синтаксис языка программирования C++, реализован класс игрового поля, все необходимые конструкторы и операторы, для каждого элемента создан класс клетки. Также, при помощи фреймворка Qt, был написан GUI для отображения информации об игровом поле.