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

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Объектно-ориентированное программирование»

Тема: Создание классов, конструкторов и методов

Студент гр. 1304 ___ Байков Е.С.

Преподаватель ___ Жангиров Т.Р.

Санкт-Петербург

2022

Цель работы.

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 перешелна клетку.

Для клетки реализованы методы, позволяющие заменять событие. (Тоесть 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.

Описание архитектурных решений и классов.

Работа написана с помощью классов для облегчения взаимодействия многих компонентов, составляющих игру. Помимо этого, код, написанный в стиле ООП легче обслуживать и модифицировать.

Была использована библиотека SFML для визуализации игры.

Рассмотрим краткое содержание классов, то, как они связаны между собой.

1. Класс *Game*: в конструкторе происходит инициализация полей класса и запуск методов, составляющих игру — игровое поле *Field*, которое состоит из клеток *Cell*, игрок *Player*, отрисовщик *Drawer*. Создание соответствующих полей класса *Game* происходит по данным, введенным пользователем. В методе *update* происходит запуск окна и с помощью полей класса происходит обновление поля и передвижение персонажа.

2. Класс *Cell*: описывает игровую клетку с присущим ей типом. Тип клетки определяет событие, которое в ней происходит или не происходит, а также является ли клетка проходимой или не проходимой.

3. Класс *CellView*: хранит в себе форму клетки и способ ее отрисовки с помощью библиотеки SFML. Цвет клетки зависит от ее типа.

4. Класс *Player*: реализует игрока с пятью характеристиками — *damage, mana, health, playerClass, playerRace*, две из которых задаются пользователем, остальные зависят от пользовательских данных и инициализируются в конструкторе.

5. Класс *PlayerView*: хранит в себе спрайт персонажа, а также задает его параметры.

6. Класс *Field*: описывает игровое поле, состоящее из клеток *Cell*. Перед его реализацией можно указать размер игрового поля, либо ввести параметры для поля по умолчанию. Также реализует движение персонажа и обновление клеток (изменение события после ухода персонажа с клетки).

7. Класс *Drawer*: является классом отрисовки объектов игры – поля и персонажа, включает в себе соответствующие методы.

8. Интерфейс *Event*: реализует событие для клетки.

9. Класс *CellEvent*: наследуется от *Event* и переопределяет его метод, под определенную клетку.

10. Класс *RandomNumberGenerator*: реализует в себе метод генерации псевдослучайного числа, необходимого для изменения игрового поля в процессе игры.

11. Класс *Controller*: является классом который осуществляет передвижение игрока.

12. Класс *Listener*: является классом считывания кнопок, которые вводит пользователь перед игрой.

Демонстрация работы программы и тестирование.

При запуске программа в консоли обращается к пользователю с предложением ввести данные об игроке — о его классе и расе. На рисунке 1 показан пример вывода программы.

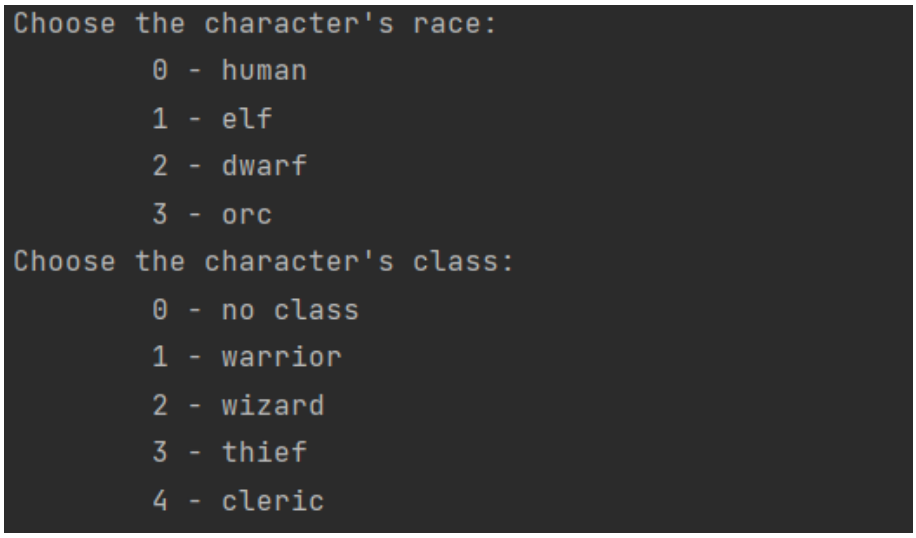

Рисунок 1 — Создание персонажа.

Затем программа обращается к пользователю с предложением ввести размер игрового поля. Пример работы программы можно увидеть на рисунке 2.

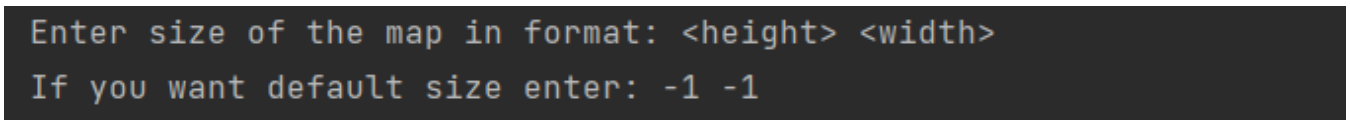

Рисунок 2 — Создание игрового поля.

При корректном вводе программа создает поле с заданным размером и игрока в случайной клетке поля. Клетка под игроком окрашивается в желтый цвет при условии, что игрок стоит на свободной клетке, если клетка не свободна, то под игроком будет бледно-зеленый цвет. Поле и отображение игрока показано на рисунке 3.

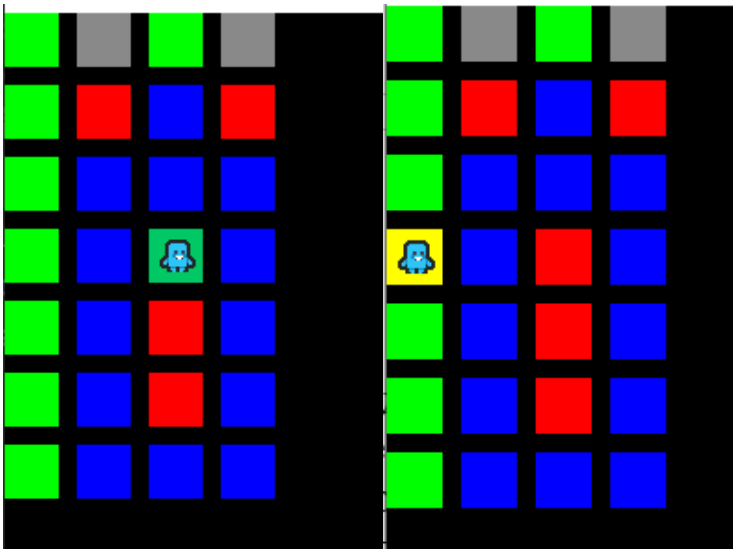

Рисунок 3 — Отображение игрока и реакция клеток.

При нажатии стрелочек (вверх, вниз, вправо, влево) игрок перемещается по полю. Также при попытке зайти на серую клетку (закрытую) игрок будет стоять на месте. Реализована «зацикленность» поля. Пример продемонстрирован на рисунке 4.

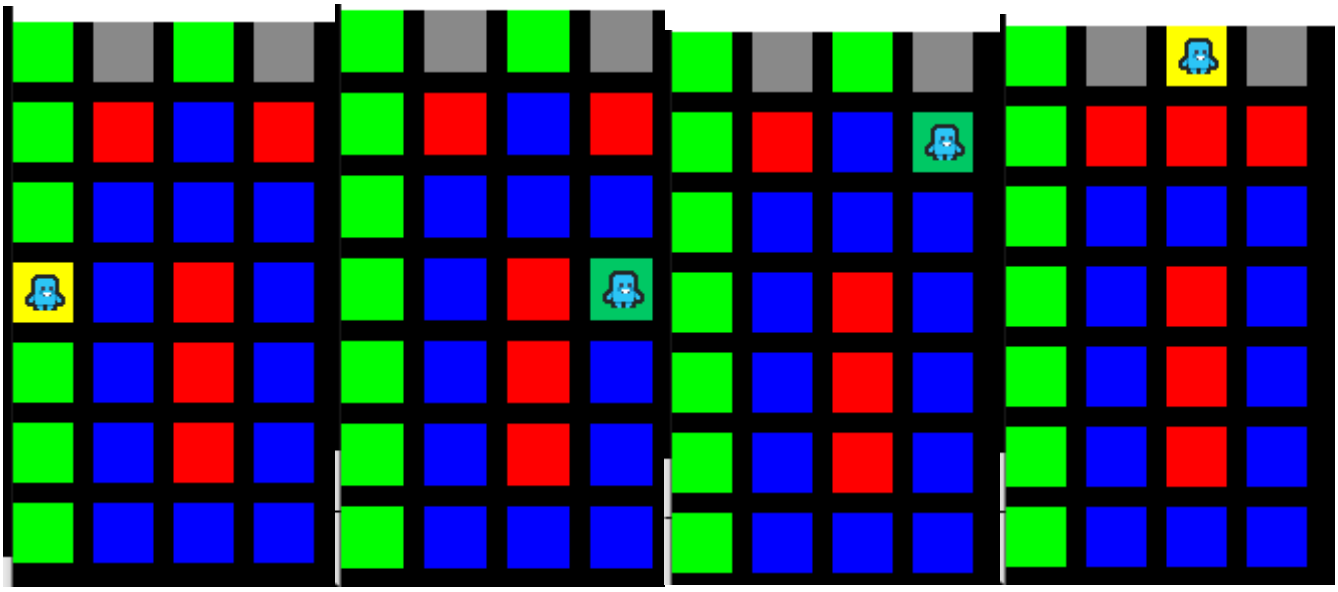

Рисунок 4 — Способность реагирования и перемещения в разные стороны на разные клетки.

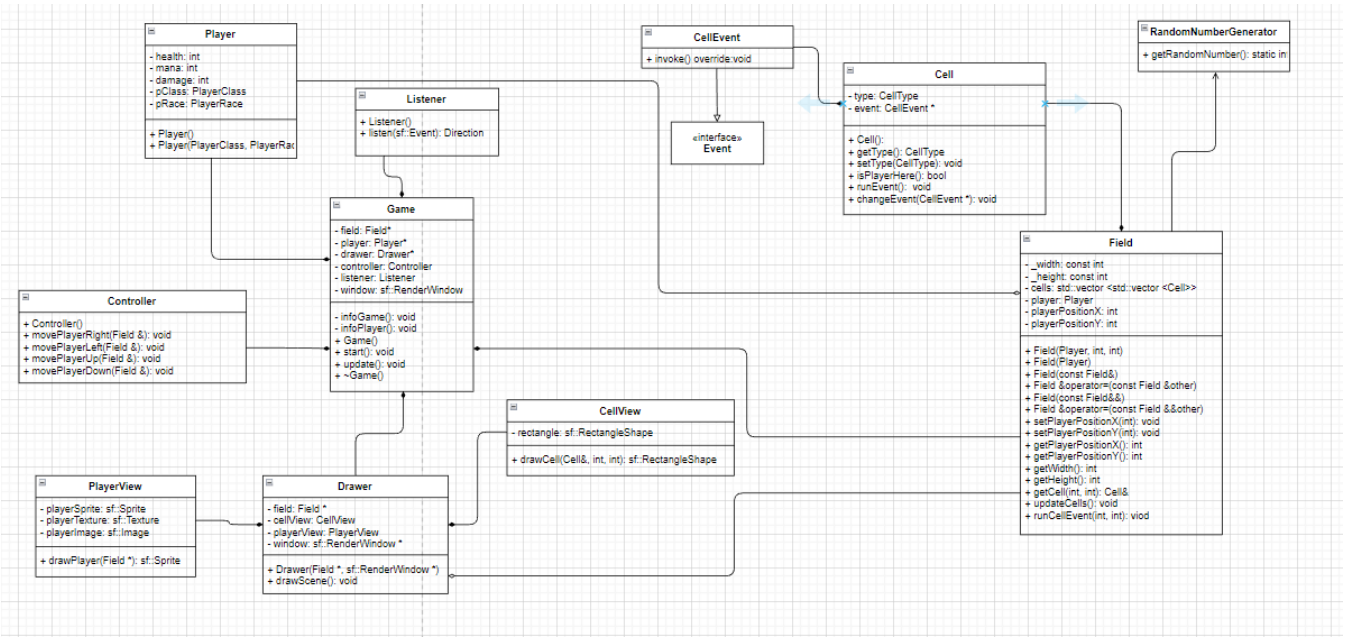

Рисунок 6 — UML-диаграмма классов.

Вывод.

Реализовано игровое поле, состоящее из клеток. Поле имеет зацикленную структуру, игрок может передвигаться по нему с помощью стрелочек. Клетки имеют разный тип и окрашиваются в соответствии с ним, а также реагирую на присутствие игрока.

Была проведена работа с использованием классов при помощи языка C++, изучены основы использования библиотеки SFML для реализации GUI, а также принципы составления UML-диаграмм.