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

ОТЧЕТ

по лабораторной работе №9

по дисциплине «Объектно-ориентированное программирование»

Тема: Разработка системы динамического изменения карты

Студент гр. 1304 _____ Павлов Д.Р.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2022

Цель работы.

Разработать набор классов, реализующим систему динамического изменения части игрового поля (например, погода). Данные классы должны менять игровую карту разными способами: изменять передвижение игрока по полю, делать игровые события неактивными, менять проходимость клеток. Данная система должна срабатывать случайно вне зависимости от действий игрока, и менять лишь часть игрового поля. Например, туман делает передвижения игрока случайным, гроза делает неактивными события, сильный ветер позволяет проходить непроходимые клетки. Эффект от такого изменения должен длиться ограниченное количество ходов.

Требования.

Разработан интерфейс изменения поля

Реализованы три разных типа изменения поля, по разному влияющих на

игровой процесс, и влияющих на разные систему

Изменение поля должно длиться ограниченное количество ходов, после

исчезновения, возвращать к изначальным состояниям. Но, если было

подобрано событие, оно не должно восстанавливаться

Изменение поля должно срабатывать случайным образом

Описание архитектурных решений и классов.

Описание Классов:

Интерфейсы:

1) IWeather — интерфейс класса погоды. Имеет чисто виртуальный метод *void set_weather(Field*, weather_zone*)*, который должен устанавливать конкретную погоду. *Вторым аргументом идет указатель на структуру ***weather_zone***, которая хранит в себе координаты вершин прямоугольника, внутри которого происходит какая-то конкретная погода. Сама структура хранится в заголовочном файле вместе с интерфейсом.

Конкретная погода:

3) Fog — наследник интерфейса погоды. Void set_weather() - заменяет погодный статус заданных клеток на FOG. Во время тумана Пользователь не видит где Герой на заданной окрестности.

4) Fire — наследник интерфейса погоды. Void set_weather() - заменяет погодный статус заданных клеток на FIRE. Во время пожара Герой, идя по заданной окрестности, получает урон.

5) Freeze — наследник интерфейса погоды. Void set_weather() - заменяет погодный статус заданных клеток на FREEZE. Во время Заморозки все события в заданной окрестности неактивны.

6) Move_Blocker — наследник интерфейса погоды. Void set_weather() - заменяет погодный статус заданных клеток на MOVE_BLOCKER. Во время Мув_Блокера (не придумал нормальное название в данном контексте) запрещается ходить по заданному полю.

7) Weather_Wrapper— Класс обертки для классов выше. Хранит вектор указателей на Интерфейс Погоды, который нужен для определения возможной погоды; указатель на погодную зону, а так же булевое поле, которое подсказывает, действует ли сейчас погода на поле или нет. Так же класс имеет несколько приватных методов: *void set_zone(Field*)* - смена погодной зоны исходя из ограничений поля, *void cause_the_weather(Field*, int)* — делегирует *set_weather* у заданного элемента вектора с погодами, *void remove_the_weather(Field*)* - убирает погодное действие на поле; а так же один публичный метод — *void roll_the_dice_to_weather(Field*)*, который «кидает кубик» на вызов/удаление погодного эффекта.

Демонстрация работы программы и тестирование.

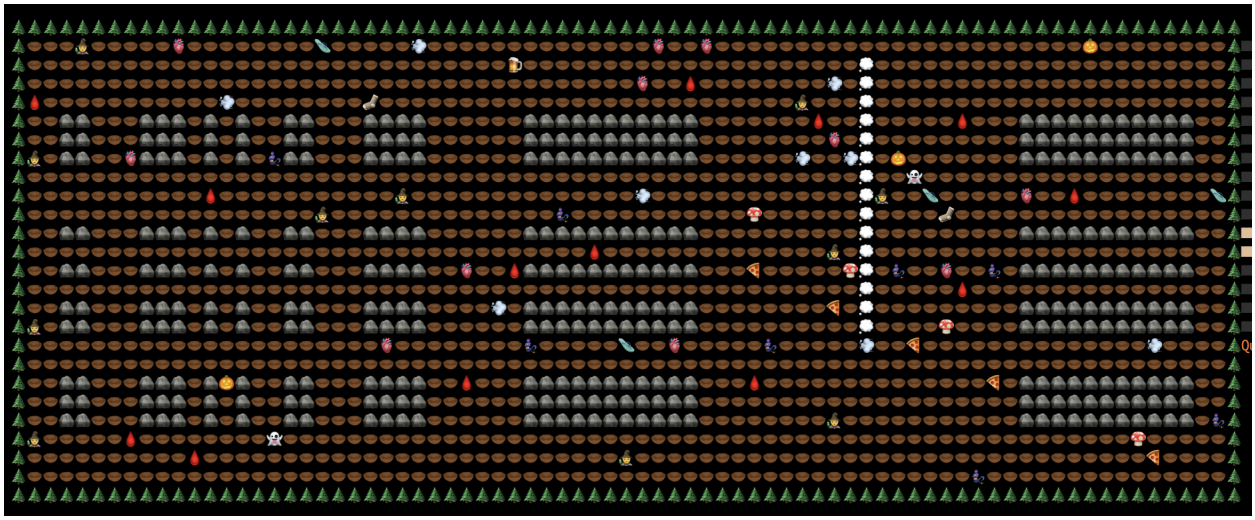

Рисунок 1 — Герой спрятался в Тумане.

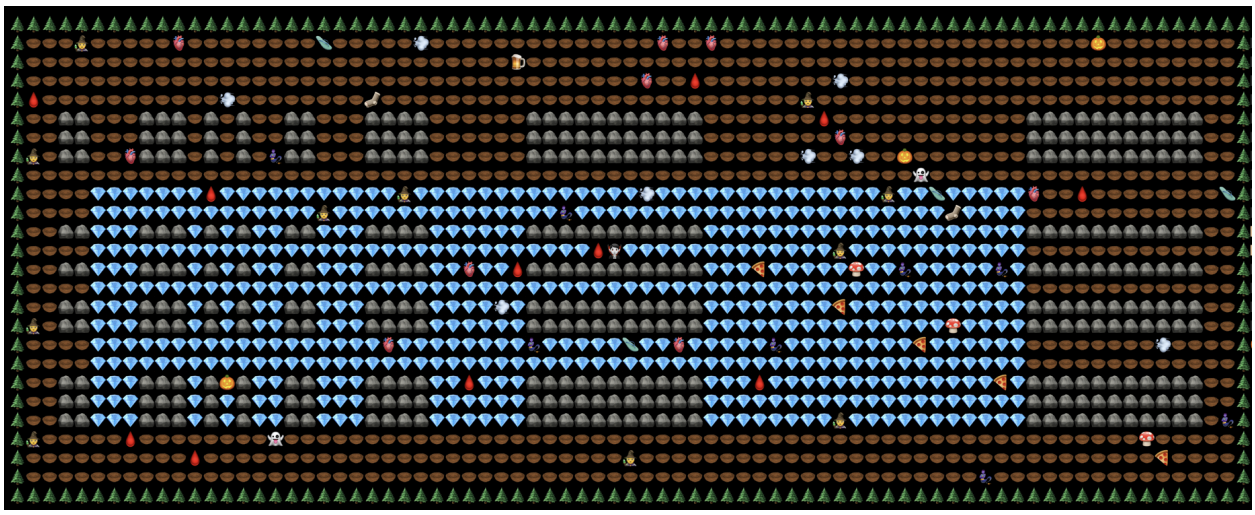

Рисунок 2 — Заморозка.

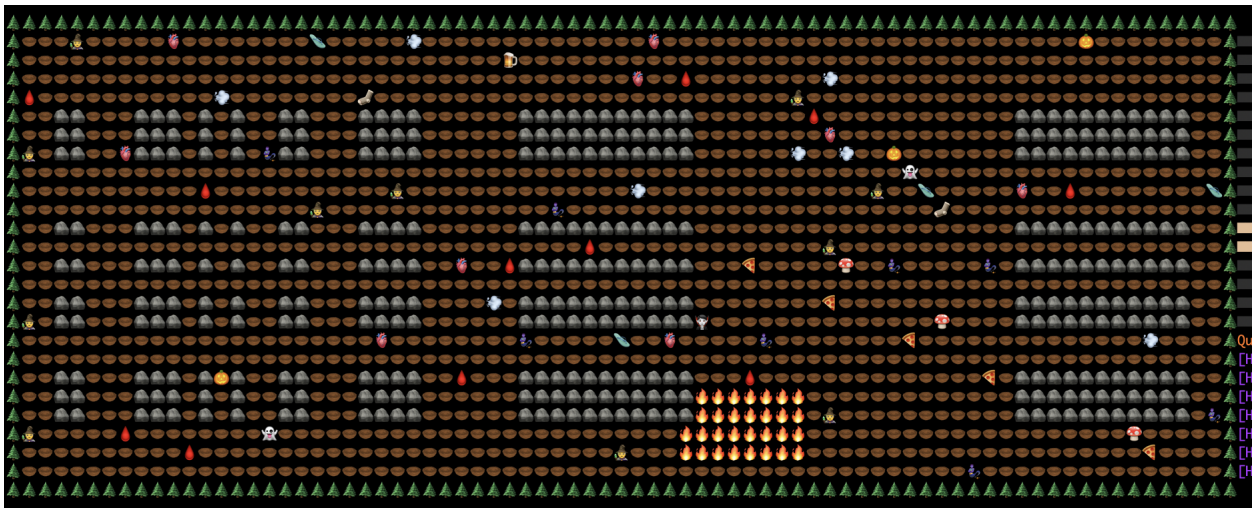

Рисунок 3 — Пожар.

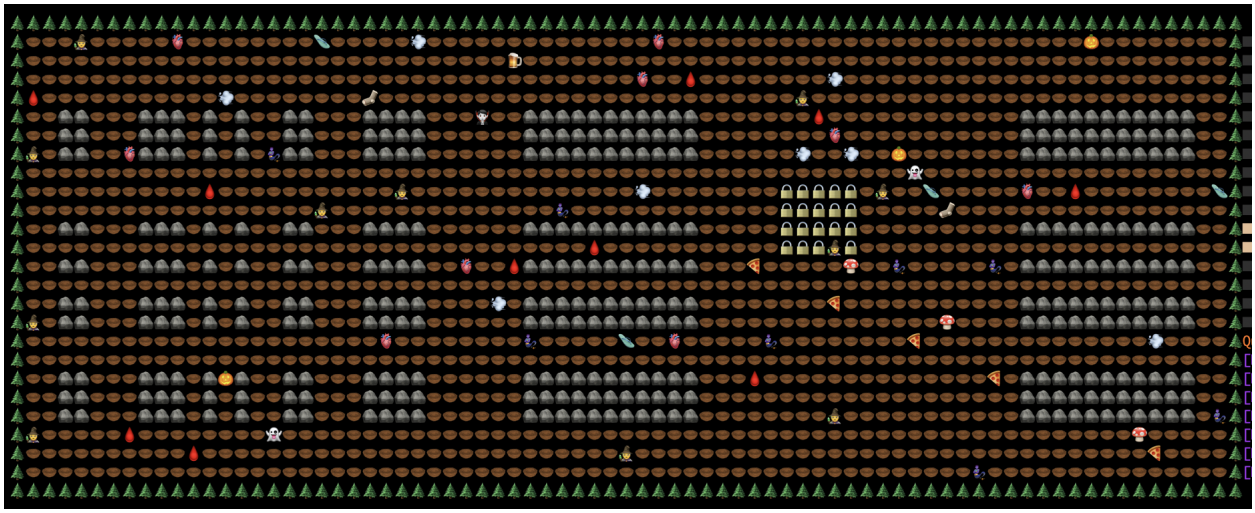

Рисунок 4 — Запрет на передвижение героя (Погода).

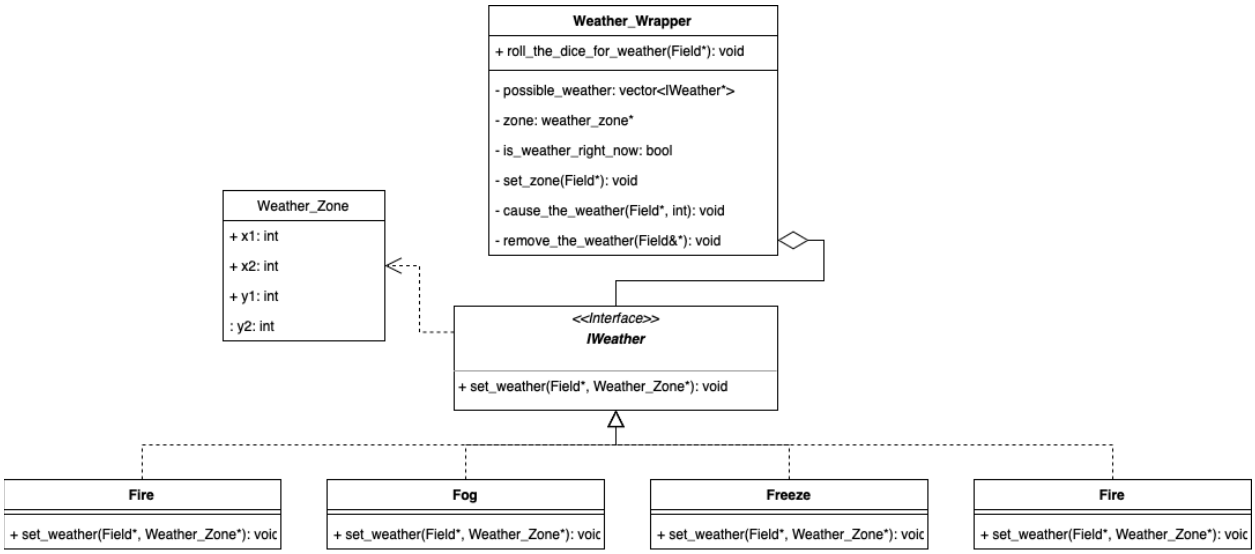

Рисунок 5 — uml диаграмма лаб работы.

Вывод.

Реализована динамическая система смены части игрового поля(погода).

Была изучена работа с классами на языке C++, паттерны проектирования, основы составления UML-диаграмм.