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

ОТЧЕТ

по лабораторной работе №6

по дисциплине «Объектно-ориентированное программирование»

Тема: Сериализация, исключения.

Студент гр. 1304 _____ Павлов Д.Р.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2022

Цель работы.

Реализовать систему классов позволяющих проводить сохранение и загрузку состояния игры. При загрузке должна соблюдаться транзакционность, то есть при неудачной загрузки, состояние игры не должно меняться. Покрыть программу обработкой исключительных состояний.

Требования.

Разработан класс/набор классов отслеживающий изменения разных уровней

Разработаны классы для вывода в консоль и файл с соблюдением идиомы RAII и перегруженным оператором вывода в поток.

Разработанные классы спроектированы таким образом, чтобы можно было добавить новый формат вывода без изменения старого кода (например, добавить возможность отправки логов по сети)

Выбор отслеживаемых уровней логирования должен происходить в runtime

В runtime должен выбираться способ вывода логов (нет логирования, в консоль, в файл, в консоль и файл)

Описание архитектурных решений и классов.

В данной лабораторной работе ***описывается паттерн Memento.***

Описание классов:

1) Memento — Класс Интерфейс «Снимка». Интерфейс Снимка предоставляет способ извлечения данных снимка, таких как игровое поле, количество атрибутов героя, позиция героя на поле, уровень игрвого поля. Однако он не раскрывает состояние Создателя.

2) Memento_Save — Класс Наследник от Memento, является Конкретным Снимком. Перегружает все абстрактные методы Интерфейса таким образом, чтобы они считывали информацию с конкретных файлов сохранений. Конкретный снимок так же отслеживает извлекаемые данные на корректность, если они неверные — выкидывается исключение. Данные могут быть неверными в случае если: в файле сохранения указаны отрицательные значения; размеры поля неверные; поле содержит больше/меньше одного героя; и тд...

3) Saver — Класс Создателя Снимков. Создатель содержит некоторое важное состояние, которое может со временем меняться. Он также объявляет метод сохранения состояния внутри снимка и метод восстановления состояния из него. Так же создатель осуществляет запись конкретного момента игры в файлы сохранений.

4) Caretaker – Класс Опекун. Опекун не зависит от класса Конкретного Снимка. Таким образом, он не имеет доступа к состоянию создателя,

хранящемуся внутри снимка. Он работает со всеми снимками через базовый интерфейс Снимка.

Демонстрация работы программы и тестирование.

Рисунок 1 — Игра сохранена.

На Рисунке 1 можно заметить надпись, всплывающая при сохранении данного момента игры.

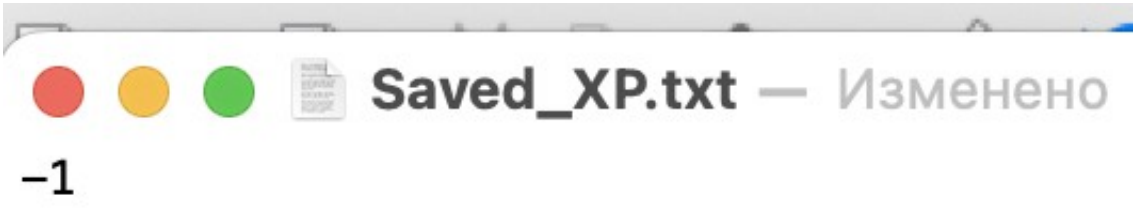

Рисунок 2 — Изменение сохраненного атрибута Героя на некорректный.

Рисунок 3 — Не удается загрузить игру.

Как видно из Рисунков 2-3, при неправильных сохраненных данных срабатывает ошибка.

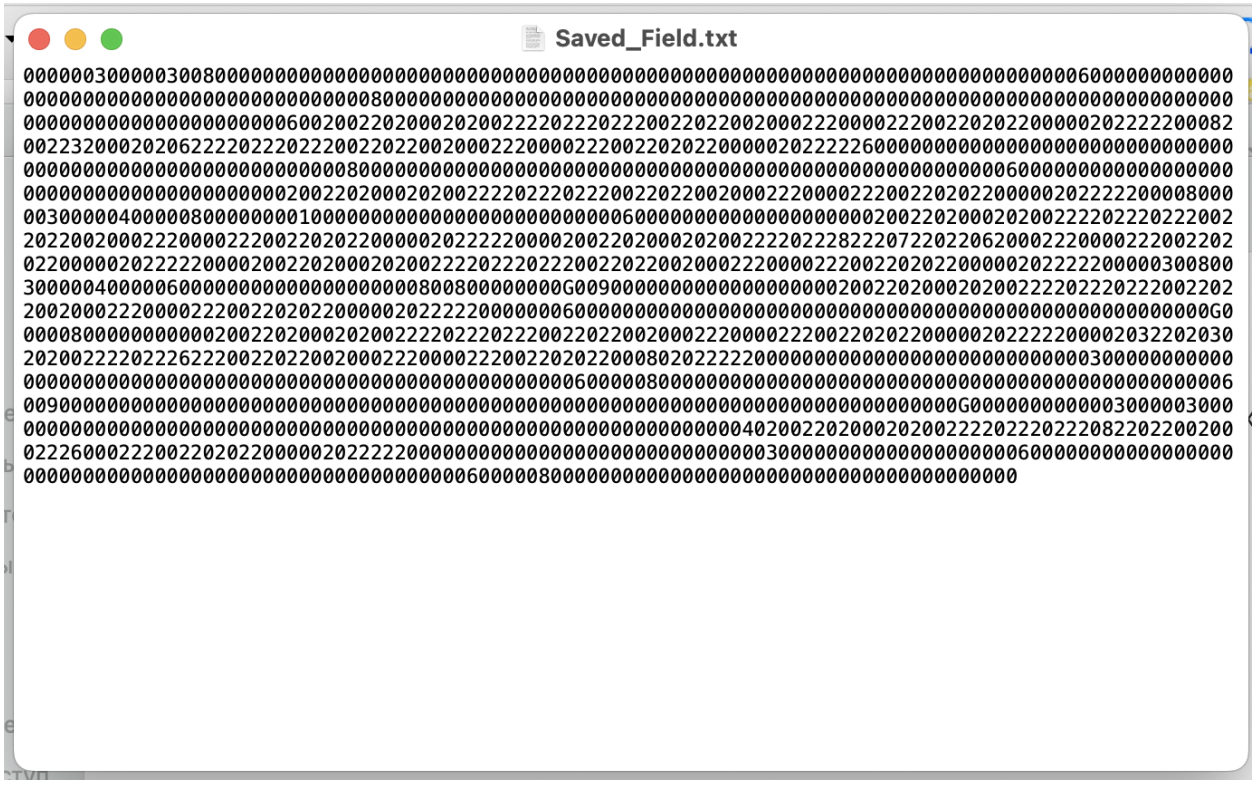

Рисунок 4 — сохраненное поле

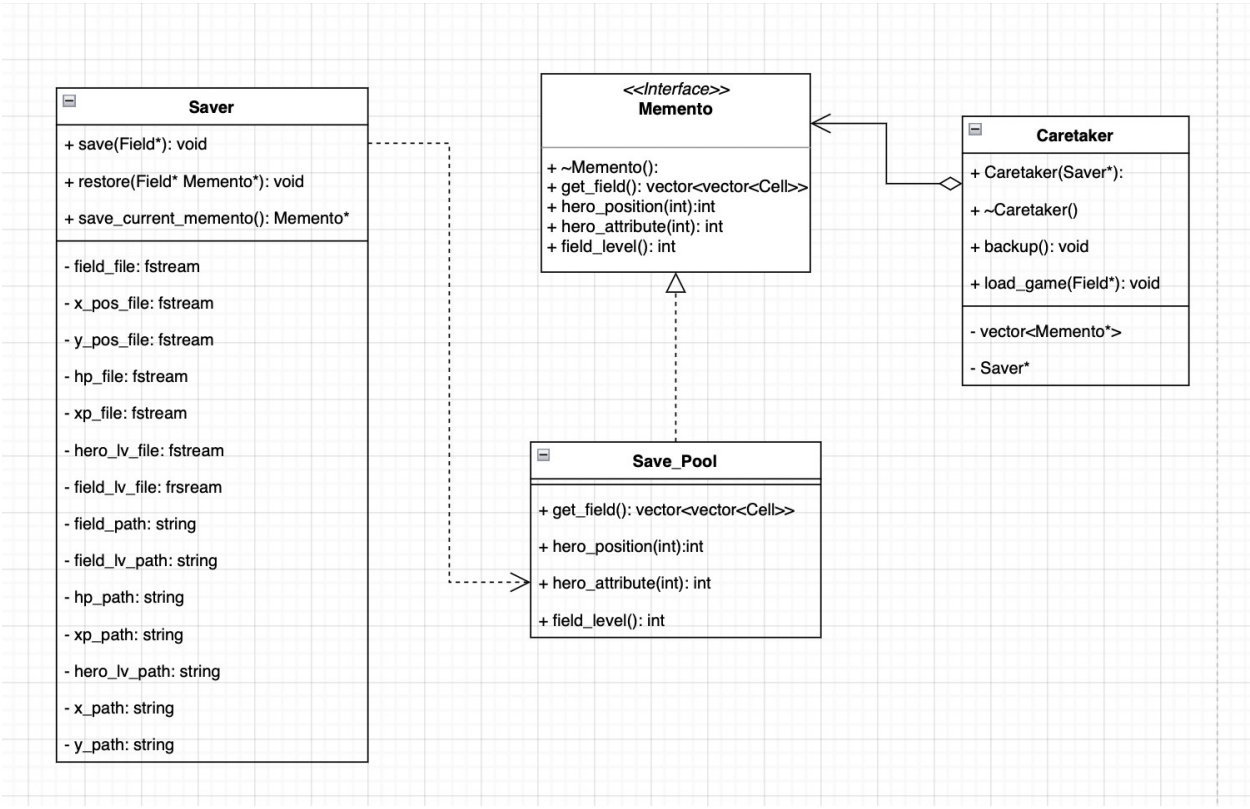

Рисунок 4 — UML Диаграмма Лабораторной работы.

Вывод.

Реализовано загрузка/сохранения игрового процесса. Так же программа была покрыта исключениями.

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