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

ОТЧЕТ

по лабораторной работе №6

по дисциплине «Объектно-ориентированное программирование»

Тема: Сериализация, исключения

Студентка гр.1381 _____ Рымарь М.И.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2022

Цель работы.

Реализовать систему классов, позволяющих создавать сохранение и загрузку игры. Покрыть программу обработкой исключительных состояний. При неудачной попытке загрузки игры её состояние не должно меняться, то есть должна соблюдаться транзакционность.

Задание.

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

Требования:

Реализована загрузка и сохранение состояния игры.

Сохранение и загрузка могут воспроизведены в любой момент работы программы.

Загрузка может произведена после закрытия и открытия программы.

Программа покрыта пользовательскими исключениями.

Пользовательские исключения должны хранить полезную информацию, например значения переменных, при которых произошло исключение, а не просто сообщение об ошибке. Соответственно, сообщение об ошибке должно учитывать это поля, и выводить информацию с учетом значений полей.

Исключения при загрузке обеспечивают транзакционность.

Присутствует проверка на корректность файла сохранения. (Файл отсутствует; в файле некорректные данные, которые нарушают логику; файл был изменен, но данные корректны с точки зрения логики).

Примечания:

Исключения должны обрабатываться минимум на фрейм выше, где они были возбуждены.

Для реализации сохранения и загрузки можно использовать мементо и посетителя. Для проверки файлов можно рассчитывать хэш от данных.

Выполнение работы.

В ходе выполнения лабораторной работы для реализации всех подзадач созданы такие классы, как: Memento; Originator; семейство классов исключений, наследованных от класса GameException.

Класс Memento имеет два метода saveState() и restoreState(). Первый метод принимает на вход две строки: информацию, которую нужно сохранить, и название файла, в который нужно сохранить состояние. Здесь открывается файл, при невозможности его открытия прокидывается исключение OpenFileException с сообщением невозможности открыть файл и его название. Второй метод принимает на вход одну строку с названием файла. При невозможности открытия файла прокидывается такое же исключение, как и в прошлом методе. Если файл открыть удалось, то данные считываются и возвращаются. Таким образом реализован поведенческий паттерн.

Класс Originator является интерфейсом с двумя виртуальными методами: saveState() и restoreState(). Первый метод сохраняет состояние игры, обращаясь к Memento, второй метод восстанавливает сохранённые данные. Классы Player и Field являются объектами, чьи состояния необходимо восстановить, поэтому они должны реализовать интерфейс.

Класс Player реализует интерфейс Originator. Метод setState() получает данные из файла сохранения, преобразует их в характеристики игрока. Если хэш от данных параметров совпадает с хэшем данных из файла сохранения, то восстанавливает характеристики игрока. Если условие не выполняется, то пробрасывается исключение InvalidDataException с сообщение об ошибке хэширования. Метод restoreState() восстанавливает характеристики, полученные после использования метода setState().

Класс Field реализует интерфейс Originator. В классе написаны методы setState(), restoreState() и hash(), аналогичные методам класса Player, однако применимые для класса поля. Метод saveGame() реализует сохранение игры, создавая при этом экземпляр Memento, передавая в него строку сохранения и вызывая метод записи в файл. Метод getState() возвращает состояние игры.

Метод hash() получает хэш код параметров поля, а из объектов самого игрового поля получает хэш, побитово смещает на определённое значение и прибавляет к переменной, хранящей промежуточный хэш. Метод restoreGame() реализует восстановление игры, создавая при этом экземпляр класса Memento. Все ошибки считывания отлавливаются и пробрасывается исключение OpenFileException со строкой и её номером в файле, где была отловлена ошибка. Если данные удалось считать, то берётся хэш этих данных и сравнивается с хэшем, записанным в файл. Если они отличаются, то пробрасывается исключение RestoreStateException с сообщением отличающихся хэшей. Иначе данные записываются в временную переменную, а затем в поле.

Семейство классов исключений, наследованных от GameException. У интерфейса есть чистый виртуальный метод what(), конструктор, принимающий строку и поле типа string. Его реализуют классы OpenFileException, RestoreStateException, SaveStateException, InvalidDataException. В каждом классе в методе what() возвращается строка, состоящая из префикса, характерного для каждого типа исключений и постфикса – переменной message, которая задаётся из аргумента конструктора.

Тестирование.

Интерфейс командной строки при загрузке игры без выхода из неё представлен на рисунке 1.

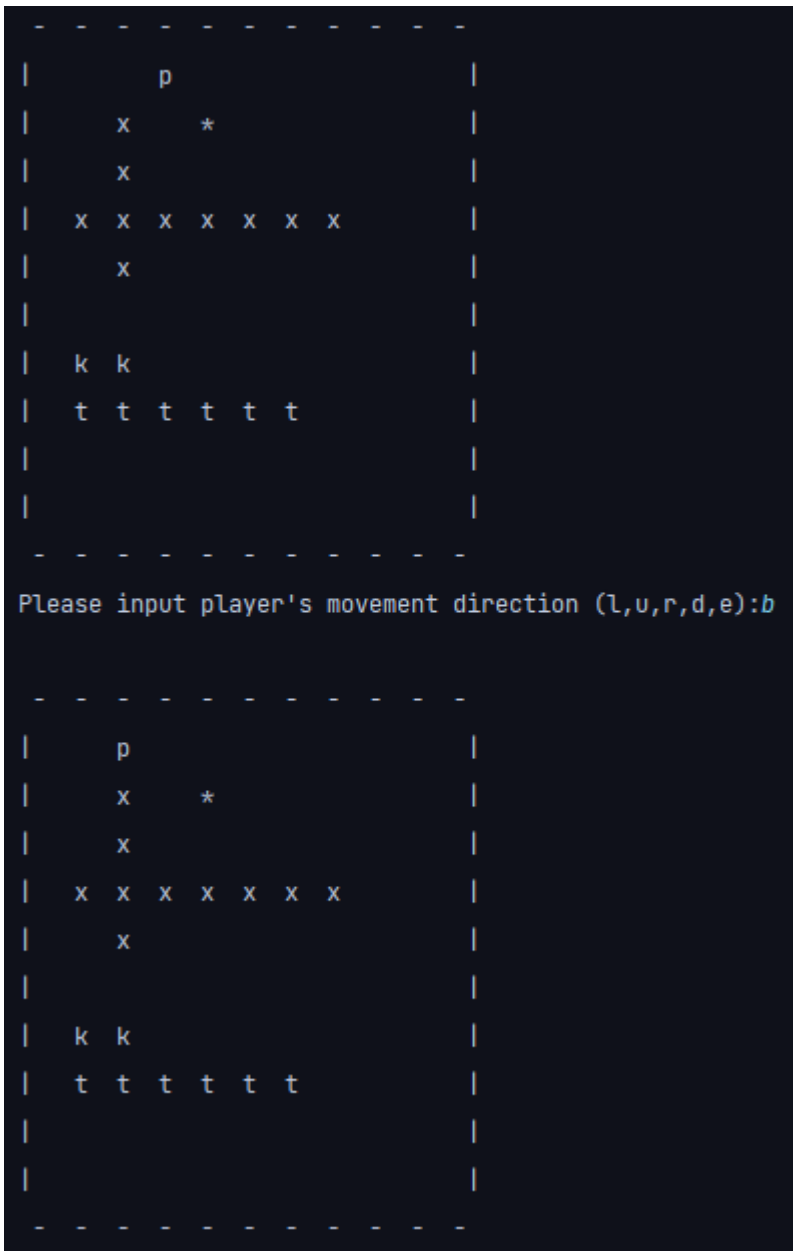

Рисунок 1 – Загрузка игры без выхода из неё

Вывод ошибки при попытке загрузки игры с изменённым файлом представлен на рисунке 2.

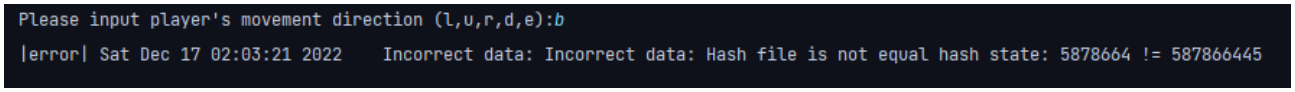

Рисунок 2 – Вывод ошибки из-за изменённого файла

Вывод ошибке при попытке загрузки игры с несуществующим файлом сохранения игрока представлен на рисунке 3.

```
Please input player's movement direction (l,u,r,d,e):b
|error| Sat Dec 17 02:09:20 2022    Failed to restore state: Failed to open the file: Failed to open the file savePlayer.txt for restoring
```

Рисунок 3 – Вывод ошибки с несуществующим файлом

UML-диаграмма межклассовых отношений.

На рисунке 4 представлена UML-диаграмма межклассовых отношений.

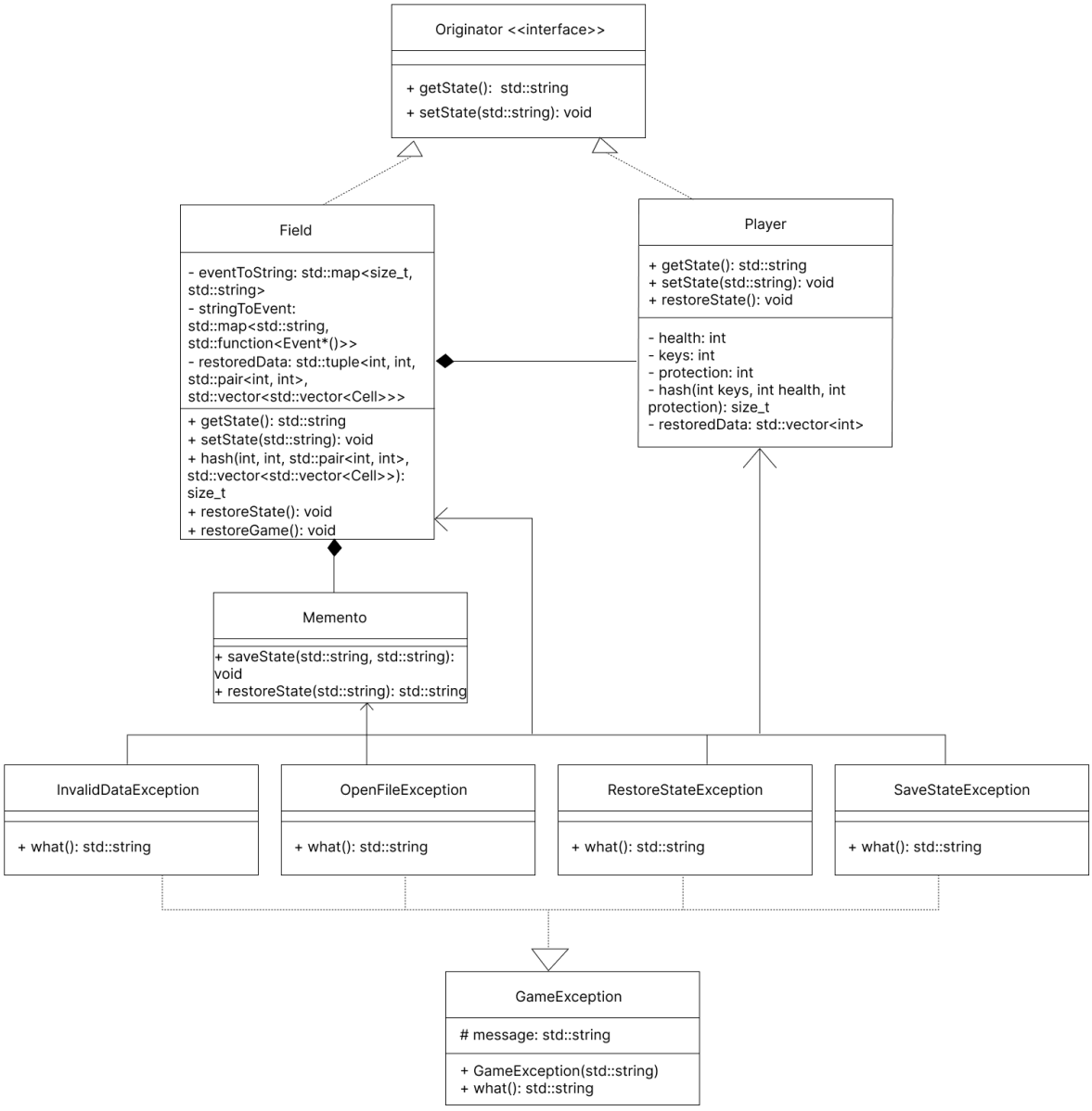

Рисунок 4 – UML-диаграмма

Выводы.

В ходе выполнения лабораторной работы изучена и реализована обработка исключительных ситуаций. Выполнено сохранение и загрузка игры. Соблюдается транзакционность при неудачной попытке сохранения игры. Реализован паттерн «мементо» и построена UML-диаграмма межклассовых отношений.