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

ОТЧЕТ

по лабораторной работе №5

по дисциплине «Объектно-ориентированное программирование»

Тема: Сериализация состояния программы

Студент гр. 8381 _____ Почаев Н.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2020

Цель работы.

Реализация сохранения и загрузки состояния программы. Основные требования:

- Возможность записать состояние программы в файл
- Возможность считать состояние программы из файла

Задание.

- Выполнены основные требования к сохранению и загрузке
- Загрузка и сохранение должно выполняться в любой момент программы
- Взаимодействие с файлами должны быть по идиоме RAII
- *Сохранение и загрузка реализованы при помощи паттерна “Снимок”
- *Реализован контроль корректности файла с сохраненными данными

Выполнение работы.

Написание работы производилось на базе операционной системы Windows 10 в среде разработки Qt Creator, для компиляции и отладки использовалась UNIX-подобная среда Cygwi и набор адаптированных инструментов MiniGW. Были задействованы пакеты GCC, CMake, а также GDB. Для компиляции текущей версии программы под Windows необходим MinGW 8.10 (для более полноценной поддержки C++17) и Qt версии 14.10 и выше.

Реализованные классы

Классы, добавленные в программу в данной лабораторной работе и их функционал представлены в табл. 1. В ней приведено общее описание классов, отдельные моменты пояснены в комментариях к коду.

Таблица 1 – Основные добавленные классы

Класс	**Назначение**
GameMementoCaretacker (./Game/Saving)	Часть реализации паттерна *“Снимок”* – опекун, который отвечает за хранение текущего снимка, пути к файлу и вызов соответствующих классов для записи в него и чтений (описаны далее).
MementoWriter (./Game/Saving/MementoFiles)	Класс, предназначенный для записи снимка в текстовом формате в файл с специализированном формате .SGF (Save Game Format). Взаимодействие с файлом реализовано по *идиоме RAII*.
MementoReader (./Game/Saving/MementoFiles)	Класс, предназначенный для чтения снимка из файла сохранения с параллельным контролем корректности подаваемых данных. Проверка осуществляется, на отрицательные значения, на превышения установленных лимитов на элементы, а также соответствие специальным тегам разделов, например, для раздела прокси поля он выглядит следующим образом: `!--FIELD_PROXY--!`

Также во всех классах, участвующих в работе программы и, как следствие, сохранении, были реализованы методы создания снимка – createMemento(). Восстановление из файла, запись в него и создание снимков осуществляется вложенным образом (для сложных структур), когда каждый элемент самостоятельно создаёт свой снимок и возвращает в виде умного указателя на уровень выше.

Снимки хранятся в структурах, находящихся в папке Game/Saving/SaveStructures.h.

Структура файла логов приведена ниже:

```
(++) означает, что строк такого формата может быть несколько или не быть вовсе.

!--GAME SAVE--!
количество игроков
размер словаря баз имён
имя X Y (++)
!--FIELD_PROXY--!
количество элементов ландшафта
X Y имя (++)
количество нейтральных элементов
X Y имя (++)
!--FIELD--!
количество юнитов
максимальное кол-во юнитов
кол-во баз
максимальное кол-во баз
ширина поля
высота поля
!--BASES--!
{
X Y
здоровье базы
тип базы
тип юнита : максимальное кол-во
} (++)
!--UNITS--!
{
X Y
тип юнита
имя
здоровье
броня
рукопашная атака
диапазон передвижения
количество токенов действия
X Y
X базы создателя Y базы создателя
} (++)
```

Выводы.

В ходе выполнения лабораторной работы были написаны требуемые классы, а также реализована сериализация состояния программы: возможность в любой момент сохранить её в специальный файл и загрузить.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ. MAIN.CPP

```
#include <iostream>

#include <QApplication>
#include <QGridLayout>
#include <QWidget>
#include <QLabel>
#include <QScreen>

#include "Tests/examples.h"
#include "Game/UIFacade.h"

int main(int argc, char *argv[])
{
    std::shared_ptr<UIFacade> game = std::make_shared<UIFacade>(argc, argv);
    game->start();

    return 0;
}
```