**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Объекто-ориентированное программирование»**

Тема: Уровни абстракции

|  |  |  |
| --- | --- | --- |
| Студент гр. 2382 |  | Ваньков Я.С. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2023

## Цель работы

Изучить методы работы с классами. Написать простую программу в стиле ООП. Изучить работу конструктора и деструктора в С++. Изучить конструкторы перемещения и конструкторы копирования.

## **Задание**

а) Создать класс игры. Класс игры отвечает за запуск игры (в начале и во время текущей сессии), выход из игры, выбор уровня, а также инициализирующего перемещение игрока. Также класс должен проводить проверку проигрыша или выигрыша игрока и давать возможность начать новую игру или завершить работу программы.

б) Создать класс (или набор классов) считывания ввода пользователя. Данный класс(ы) должен сопоставить ввод пользователя с командой, которую необходимо выполнить. И передать эту информацию в класс игры. Клавиши управления (то на какую клавишу назначено определенное действия) должны считываться из файла. Класс(ы) должен быть разработан так, чтобы:

потенциально можно было масштабировать управление с минимальным количеством изменений в коде

можно было заменить схему управления, например, заменить ввод команду через терминал на чтение команд из файла или из сети с другого устройства

Примечания:

После считывания клавиши, считанный символ должен сразу обрабатываться, и далее работа должна проводить с сущностью, которая представляет команду.

Для представления команды можно разработать системы классов или использовать перечисление enum.

Хорошей практикой является создание “прослойки” между считыванием/обработкой команды и классом игры, которая сопоставляет команду и вызываемым методом игры. Существуют альтернативные решения без явной “прослойки”

При считывания управления необходимо делать проверку, что на все команды назначена клавиша, что на одну клавишу не назначено две команды, что на одну команду не назначено две клавиши.

При работе с файлом используйте идиому RAII.

## Выполнение работы

**GameManager**

Класс игры. Класс имеет приватные методы, создающие уровни 1 или 2. Также класс имеет публичные методы, отвечающие за считывание ввода пользователя, запуск игры, окончания игры, выбора и смены уровня.

В приватных полях лежит игрок, контроллер над игроком, а также поле. Каждый раз в зависимости от уровня эти поля обновляются. Также в приватных полях лежит конфиг для считывания комманд и кконфиг для считывания ввода пользователя

*start\_level\_1()* - создает первый уровень. Передает в приватные поля нынешние характеристики уровня.

*start\_level\_2()* - создает второй уровень. Передает в приватные поля нынешние характеристики уровня.

*chooseLevel(int n)* — создает уровень с определенным номером.

s*tartLevel()* - запускает уровень. Запускает уровень, если все характеристики переданы правильно, иначе — ошибка.

*startGame()* - запускает игру. - бесконечный цикл. Считывает пользовательский вод. Каждое нажатие обрабатывается с помощью InputHandlerа.

*isWin / isLose* — проверяет выиграл игрок или програл после окончания игры.

*EndLevel( bool win)* — заканчивает уровень

**IconfigReader —** виртуальный класс, для считывания конфига. От него унаследован **FileConfigReader** выполненный в принципах RAII

**InputHandler** — класс, который в зависимости от переданного char значения запускает нужную комманду

**Icommand —** виртуальный класс

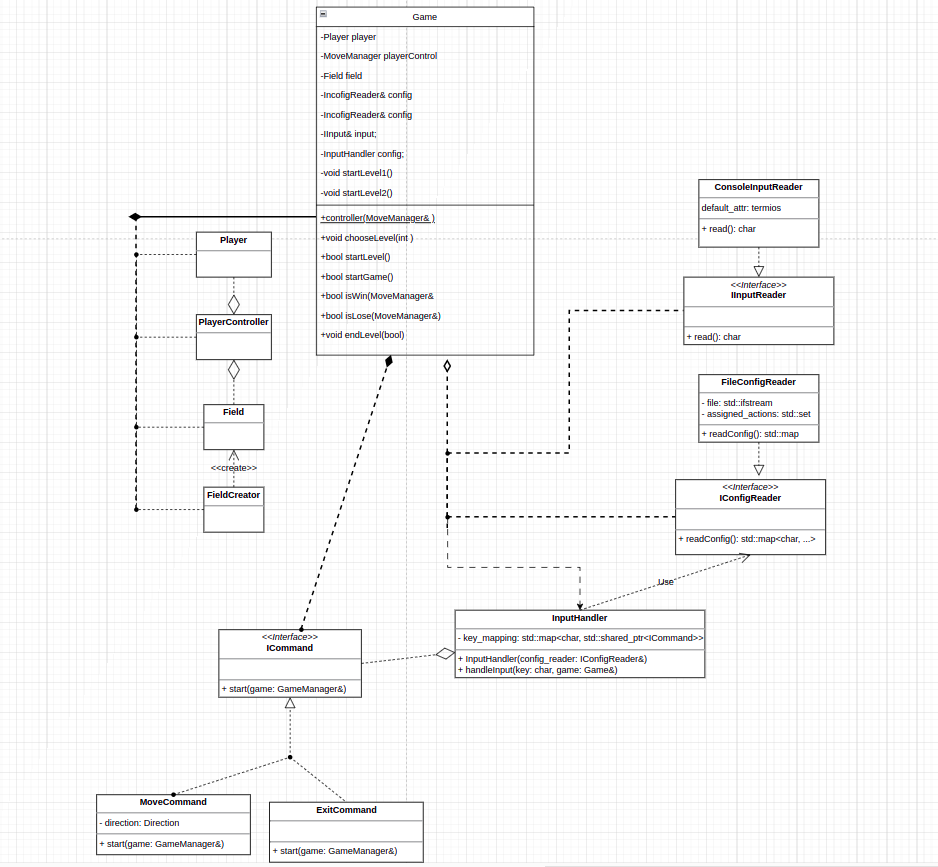
Метод start — запускает команды. В старт стоит передавать ссылку на менеджера игры. Унаследованы классы **ExitCommand и MoveCommand,** которые выполняют действия согласно названия(классы поднострочники)

**IInput**

Виртуиальный класс — от него можно отнаследовать разные классы ввода.

Метод read — вовзвращает char, введенный пользователь.

**ConsoleInput : public Iinput** — класс считывания пользовательского ввода с консоли. Обрабатываетс с помощью библиотеки termios.



## Выводы

Были изучены уровни абстракции. Были написаны классы игры, ввода.

# Приложение А Исходный код программы

# https://github.com/DoAlvaro/OOP\_2382