**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №5**

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

Тема: Отслеживание изменений

|  |  |  |
| --- | --- | --- |
| Студент гр. 2300 |  | Шумов О.Д. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2023

## Цель работы.

Реализовать набор классов для отслеживания изменений в игре, и отрисовки поля в зависимости от изменений.

## Задание

а) Реализовать класс, который связывается с игрой, и отслеживает изменения в игре: перемещение игрока, победа или выигрыш, срабатывание событий. Данный класс должен реагировать на изменения и отрисовывать игровое поле, а также выводить информацию для игрока (например, предлагать начать новую игру).

б) При отрисовке поля должна считываться информация с поля и об игроке, и в зависимости от расположения происходит вывод представления поля в терминал. В представлении поля непроходимые клетки, игрок, события должны отображаться различными символами. Игрок, события, клетки и другие игровые сущности не должны знать ничего о том, каким символом они отрисовываются. За выбор символа отвечает класс выполняющий отрисовку

Примечания:

Класс отслеживания и класс отрисовки рекомендуется делать отдельными сущностями. Таким образом, класс отслеживания инициализирует отрисовку, и при необходимости можно заменить отрисовку (например, на GUI) без изменения самого отслеживания

При проверке типов события можно использовать dynamic\_cast / typeinfo, либо сделать связывание модель-представление

## Выполнение работы

Были созданы 4 классa:

1. Tracker – класс отслеживания изменений в игре.
2. Render – абстрактный класс отрисовки поля.
3. RenderTerminal - класс наследующий Render и реализующий отрисовку в терминале.
4. States – enum class, содержащий состояния игры.

В классе Tracker реализованы следующие поля:

1. Render& render\_ - ссылка на объект класса отрисовки.
2. Player& player\_ - ссылка на объект класса игрока.
3. Field& field\_ - ссылка на объект класса поля.

Все перечисленные поля имею модификатор доступа private, что защищает их от нежелательного изменения из вне.

В классе Tracker реализованы следующие методы:

1. void check\_state(States state) – метод для проверки состояния игры. Принимает объект класса States. В switch проверяет состояние игры и вызывает нужную отрисовку.
2. Tracker(Render& render, Player& player, Field& field) – конструктор класса отслеживания. Принимает ссылки на объекты классов отриисовки, игрока и поля.

В классе Render реализованы следующие методы:

1. virtual void print\_start() const - виртуальный метод вывода информации о начале игры.
2. virtual void print\_level() const - виртуальный метод вывода информации о выборе уровня игры.
3. virtual void print\_win() const - виртуальный метод вывода информации при победе.
4. virtual void print\_lose() const - виртуальный метод вывода информации при проигрыше.
5. virtual void print\_player(Player& player) const - виртуальный метод вывода информации о игроке. Принимает ссылку на объект класса игрока.
6. virtual void print\_field(Player& player, Field& field) const - виртуальный метод вывода игрового поля. Принимает ссылки на объекты класса игрока и класса поля.
7. virtual void print\_game() const - виртуальный метод вывода информации о клавишах для управления игроком.
8. virtual void print\_new\_game() const - виртуальный метод вывода информации о предложении начать новую игру.
9. virtual void print\_end() const - виртуальный метод вывода информации о завершении игры.

Класс RenderTerminal наследует класс Render и реализует метод отрисовки в терминале.

В классе Statets реализовано перечисление enum в котором объявлены состояния игры.

Разработанную UML-диаграмму см. в приложении А.

## Выводы.

Был реализован набор классов для отслеживания изменений в игре, и отрисовки поля в зависимости от изменений.

**Приложение А  
UML-диаграмма**

