**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №7**

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

Тема: Шаблонные классы

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

Санкт-Петербург

2023

## Цель работы.

Изучить понятие шаблонных классов. Реализовать систему классов для создания врага в игровом поле.

## Задание

а) Создать шаблонный класс врага (не должен наследовать от события), который размещается на игровом поле и может по нему перемещаться и взаимодействовать с игроком. В качестве шаблонов указывается схема управления (то как двигается враг) и схема взаимодействия с игроком (то как враг взаимодействует на игрока)

б) Создать минимум 2 схемы управления для врага. Например, враг, который ищет игрока и идет к нему, и враг, который случайно бродить по игровому полю.

в) Создать минимум 2 схемы взаимодействия врага с игроком. Например, враг наносящий урон и враг, толкающий игрока.

г) Внедрить врагов на игровое поле, чтобы они совершали ход после хода игрока, а также взаимодействовали с игроком.

Примечания:

Схемы не должно иметь общий базовый класс. В данном работе применяется принцип статического полиморфизма через шаблоны.

Не забудь добавить добавление врагов на поле в классе создающих уровни.

## Выполнение работы

Были созданы 6 классов:

1. *Enemy* – абстрактный класс врага.
2. *OrdinaryEnemy* – шаблонный класс наследующий *Enemy* и реализующий создания врага.
3. *SearchPlayer* – класс передающийся в *OrdinaryEnemy* в качестве шаблона поиска игрока в игровом поле.
4. *RandomMove* – класс передающийся в *OrdinaryEnemy* в качестве шаблона случайного передвижения врага в игровом поле.
5. *DamagePlayer* – класс передающийся в *OrdinaryEnemy* в качестве шаблона нанесения урона игроку.
6. *PushPlayer* – класс передающийся в *OrdinaryEnemy* в качестве шаблона толкания игрока врагом.

В классе *Enemy* объявлены следующие методы:

1. *virtual int get\_x() const* – виртуальный метод для получения координаты врага по оси x.
2. *virtual int get\_y() const* – виртуальный метод для получения координаты врага по оси y.
3. *virtual void move(Player& player, Field& field)* – виртуальный метод для передвижения игрока в игровом поле.
4. *virtual void interact(Player& player, Control& control)* – виртуальный метод для взаимодействия врага с игроком.
5. *virtual Enemy\* create() const* – виртуальный метод для создания нового объекта класса врага.

Шаблонный класс *OrdinaryEnemy* наследует класс *Enemy* и реализует класс врага.

В классе *SearchPlayer* реализованы следующие методы:

1. *static void move(Player& player, Field& field, int& x, int& y)* – метод передвижения врага по полю, ищет игрока в поле.
2. *static void find(int\*\* arr, int x, int y, int value, int& width, int& height)* – метод поиска пути от врага к игроку.

В классе *RandomMove* реализован следующий метод:

1. *static void move(Player& player, Field& field, int& x, int& y)* – метод передвижения врага по полю, случайно передвигает врага.

В классе *DamagePlayer* реализован следующий метод:

1. *static void interact(Player& player, Control& control, int& x, int& y)* – метод взаимодействия врага с игроком, наносит урон игроку.

В классе *PushPlayer* реализован следующий метод:

1. *static void interact(Player& player, Control& control, int& x, int& y)* – метод взаимодействия врага с игроком, толкает игрока.

Разработанную UML-диаграмму см. в приложении А.

## Выводы.

Было изучено понятие шаблонного класса. Также был реализован набор классов для создания врага в игровом поле.

**Приложение А  
UML-диаграмма**

