

**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: «Шаблонные классы, генерация карты»**

Студент гр. 1384

Белокобыльский И. В.

Преподаватель

Жангиров Т.Р.

Санкт-Петербург

2022

## **Цель работы.**

Научиться реализовывать генерацию карты при помощи абстрактных правил, не зависящих от конкретного интерфейса, и основанных на шаблонах.

## **Задание.**

Реализовать шаблонный класс генерирующий игровое поле. Данный класс должен параметризоваться правилами генерации (расстановка непроходимых клеток, как и в каком количестве размещаются события, расположение стартовой позиции игрока и выхода, условия победы, и.т.д.). Также реализовать набор шаблонных правил (например, событие встречи с врагом размещается случайно в заданном в шаблоне параметре, отвечающим за количество событий)

### **Требования:**

Реализован шаблонный класс генератор поля. Данный класс должен поддерживать любое количество правил, то есть должен быть variadic template.

Класс генератор создает поле, а не принимает его.

Класс генератор не должен принимать объекты классов правил в каком-либо методе, а должен сам создавать (в зависимости от реализации) объекты правил из шаблона.

Реализовано не менее 6 шаблонных классов правил

Классы правила должны быть независимыми и не иметь общего класса-интерфейса

При запуске программы есть возможность выбрать уровень (не менее 2) из заранее заготовленных шаблонов

Классы правила не должны быть только “хранилищем” для данных.

Так как используются шаблонные классы, то в генераторе не должны быть `dynamic_cast`

### **Примечания:**

Для задания способа генерации можно использовать стратегию, компоновщик, прототип

Не рекомендуется делать static методы в классах правилах

### **Выполнение работы.**

Для реализации генерации карты при помощи правил был создан класс MapGenerator, который в качестве параметров шаблона принимает классы – правила, которые в свою очередь должны являться функторами – так MapGenerator их вызывает.

Сами правила, как указано выше, реализованы в виде функторов. В каждое правило передается GameMediator, так как с его помощью происходит изменение и получение поля, событий и пр. Каждое правило по-своему изменяет или добавляет следующие элементы игры:

- Поле
- Препятствия
- Положение игрока в пространстве
- Выход
- Ловушки
- Туннель

Для создания готовых шаблонов из правил был реализован класс MapPresetsCollector, который в методе generate в зависимости от желания пользователя определяет набор правил и отдает их MapGenerator

### **Тестирование программы.**

В качестве примера была выбрана и сгенерирована карта «Озеро»



## Выводы.

В рамках данной лабораторной работы была доработана программа на языке C++ из предыдущей работы. Реализован генерации карты на основании классов правил.

# ПРИЛОЖЕНИЕ 1

## UML-диаграмма классов:

