**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

**отчет**

**по лабораторной работе №4**

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

**Тема: шаблонные классы, управление**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 0382 |  | Андрющенко К.С |
| Преподаватель |  | Жангиров Т. Р |

Санкт-Петербург

2021

**Цель работы.**

Создание шаблонного класса правил игры, определяющего ход и результат игры.

**Задание.**

## Необходимо определить набор правил для игры в виде классов (например, какие задачи необходимо выполнить, чтобы он мог выйти с поля; какое кол-во врагов и вещей должно быть на поле, и.т.д.). Затем определить класс игры, которое параметризуется правилами. Класс игры должен быть прослойком между бизнес-логикой и командами управления, то есть непосредственное изменение состояния игрой должно проходить через этот класс.

## Требования.

* Созданы шаблонные классы правил игры. В данном случае параметр шаблона должен определить конкретные значения в правилах.
* Создан шаблонный класс игры, который параметризуется конкретными правилами. Класс игры должен проводить управление врагами, передачей хода, передавать информацию куда переместить игрока, и.т.д.

**Выполнение работы.**

В ходе работы реализуется паттерн проектирования.

UML-диаграмма для данной лабораторной работы см. Рисунок 1.

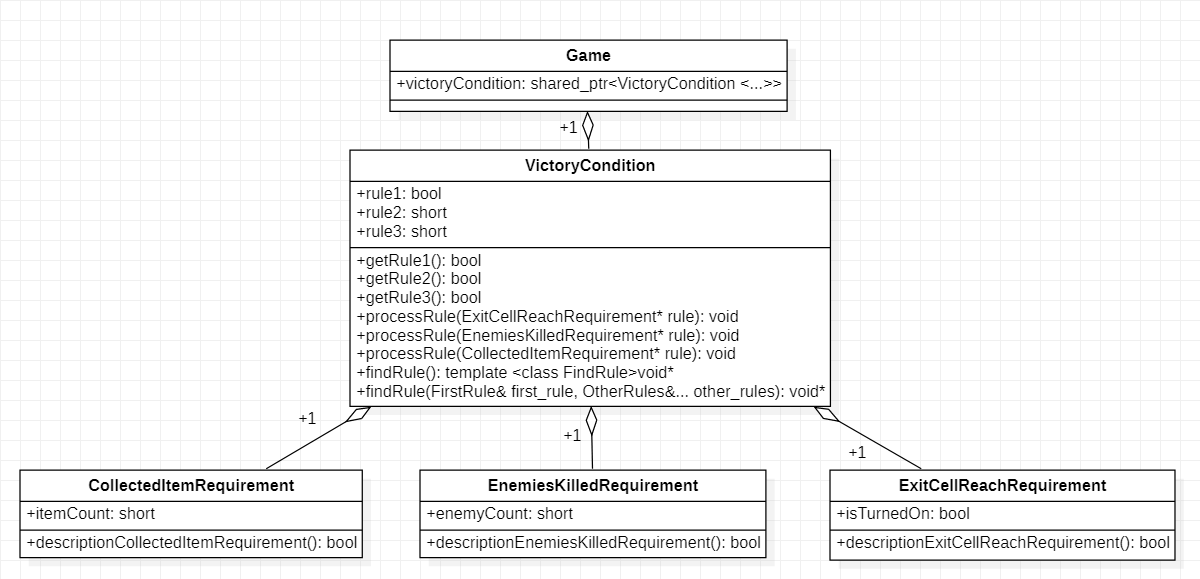


Рисунок 1 – UML-диаграмма лабораторной работы №4

Описание диаграммы.

## Описание диаграммы.

## Классы, созданные и модифицированные в данной лабораторной работе, их методы и свойства.

Класс VictoryCondition:

Заголовочный файл см. ПРИЛОЖЕНИЕ A.

Класс, являющийся контейнером для классов правил.

Свойства класса.

*rule1N*–Значение переменной зависит от соответствующего правила. При проверке выполнения правила после совершенных действий игрока, сравниваются значения данных переменных и измененные поля Game.

Методы класса.

*void processRule–* Получение значений rule.

*void\* template<>findRule()*– Проверка существования rule.

Классы правил:

У каждого класса правил есть одно свойство и метод, возвращающий это свойство.

Были созданы 3 класса правил.

Количество подобранных предметов.

*Class CollectedItemRequirement.*

Заголовочный файл см. ПРИЛОЖЕНИЕ B.

Условие прохождение клетки выхода.

*Class ExitCellReachRequirement.*

Заголовочный файл см. ПРИЛОЖЕНИЕ C.

Количество убитых врагов.

*Class EnemiesKilledRequirement.*

Заголовочный файл см. ПРИЛОЖЕНИЕ D.

Отношение данных классов с классом контейнером VictoryCondition – агрегация.

Отношение класса VictoryCondition с классом – контейнером Game – агрегация.

## Проверка классов.

Приложим скриншоты тестов, находящихся в main(), результат и при необходимости памяти после выполнения тестов. Создадим 3 правила с разными параметрами. Проведем игрока до «конца игры», выведем результаты подбора предметов и убийства врагов. Выедем результат игры.

Main():

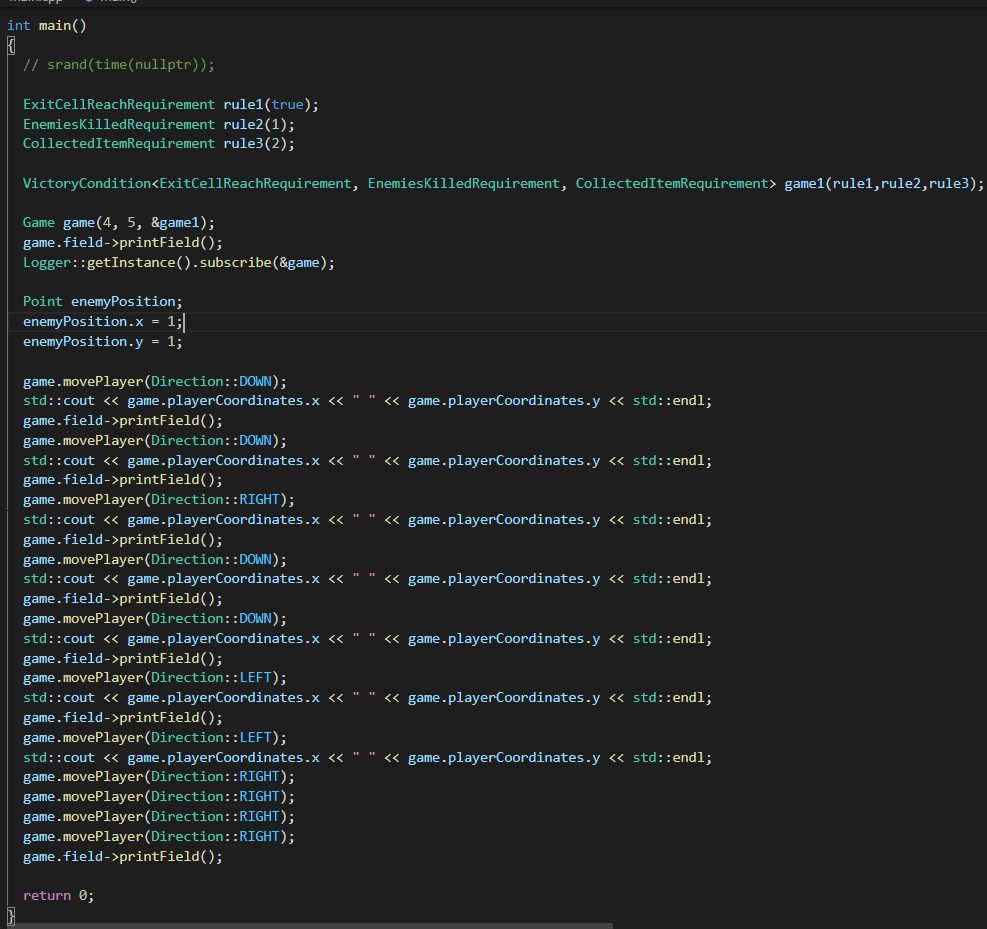


Рисунок 2 - Движение игрока по полю, запись логов.

Output(консоль)/(файл game.log):

2 ent 0 3 0 0

1 0 4 4 0

1 3 0 1 0

0 0 0 0 0 ext

0 1

0 ent 2 0 3 0

1 0 4 4 0

1 0 3 1 0

0 0 0 0 0 ext

Non traversable for enemy

0 2

0 ent 0 2 0 3

1 0 4 4 0

1 0 3 1 0

0 0 0 0 0 ext

MOVEMENT: Player's current health is max

MOVEMENT: Player's current defence: 2

MOVEMENT: Player's current attack: 20

MOVEMENT: Attack: 0

MOVEMENT: Defense: 1

MOVEMENT: Health: 5

Non traversable for enemy

1 2

0 ent 0 0 0 3

1 0 2 4 0

1 0 3 1 0

0 0 0 0 0 ext

MOVEMENT: Player's current health: 100

MOVEMENT: Player's current defence: 0

MOVEMENT: Player's current attack: 25

MOVEMENT: Attack: 5

MOVEMENT: Defense: 0

MOVEMENT: Health: 0

Non traversable for enemy

1 3

0 ent 0 0 0 3

1 0 0 2 0

1 0 3 1 0

0 0 0 0 0 ext

Non traversable for enemy

1 4

0 ent 0 0 0 3

1 0 0 0 2

1 0 3 1 0

0 0 0 0 0 ext

You have been damaged

BATTLE: Damage: 10

MOVEMENT: Player's current health: 90

BATTLE: Emeny is dead.

Non traversable for enemy

Non traversable for enemy

0 4

0 ent 0 0 0 2

1 0 0 0 0

1 0 3 1 0

0 0 0 0 0 ext

Non traversable for enemy

0 4

Non traversable for enemy

Non traversable for enemy

You Win!

Non traversable for enemy

Non traversable for enemy

0 ent 0 0 0 0

1 0 0 0 0

1 0 3 1 0

0 0 0 0 2 ext

Обработка случая несоответствия количества врагов и вещей на поле и в требованиях:

2 ent 0 3 0 0

1 0 4 4 0

1 3 0 1 0

0 0 0 0 0 ext

There are not so many enemies here.

You must kill: 2 enemies

0 1

0 ent 2 0 3 0

1 0 4 4 0

1 0 3 1 0

0 0 0 0 0 ext

Non traversable for enemy

0 2

0 ent 0 2 0 3

1 0 4 4 0

1 0 3 1 0

0 0 0 0 0 ext

MOVEMENT: Player's current health is max

MOVEMENT: Player's current defence: 2

MOVEMENT: Player's current attack: 20

MOVEMENT: Attack: 0

MOVEMENT: Defense: 1

MOVEMENT: Health: 5

Non traversable for enemy

1 2

0 ent 0 0 0 3

1 0 2 4 0

1 0 3 1 0

0 0 0 0 0 ext

MOVEMENT: Player's current health: 100

MOVEMENT: Player's current defence: 0

MOVEMENT: Player's current attack: 25

MOVEMENT: Attack: 5

MOVEMENT: Defense: 0

MOVEMENT: Health: 0

Non traversable for enemy

1 3

0 ent 0 0 0 3

1 0 0 2 0

1 0 3 1 0

0 0 0 0 0 ext

Non traversable for enemy

1 4

0 ent 0 0 0 3

1 0 0 0 2

1 0 3 1 0

0 0 0 0 0 ext

You have been damaged

BATTLE: Damage: 10

MOVEMENT: Player's current health: 90

BATTLE: Emeny is dead.

Non traversable for enemy

Non traversable for enemy

0 4

0 ent 0 0 0 2

1 0 0 0 0

1 0 3 1 0

0 0 0 0 0 ext

Non traversable for enemy

0 4

Non traversable for enemy

Non traversable for enemy

You Win!

Non traversable for enemy

Non traversable for enemy

0 ent 0 0 0 0

1 0 0 0 0

1 0 3 1 0

0 0 0 0 2 ext

Если требования не выполнены:

**Вывод.**

В ходе работы были написан класс правил игры, определяющий ход и результат игры.

ПРИЛОЖЕНИЕ A

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

**Файл VictoryCondition.hpp**

#pragma once

#include "ExitCellReachRequirement.hpp"

#include "EnemiesKilledRequirement.hpp"

#include "CollectedItemRequirement.hpp"

template <class... Types>

class VictoryCondition{

public:

VictoryCondition(Types&... parameters){

//найдем правило 1

auto rule1 = reinterpret\_cast<ExitCellReachRequirement\*>(findRule<ExitCellReachRequirement>(parameters...));

processRule(rule1);

//найдем правило 2

auto rule2 = (EnemiesKilledRequirement\*)findRule<EnemiesKilledRequirement>(parameters...);

processRule(rule2);

//найдем правило 3

auto rule3 = (CollectedItemRequirement\*)findRule<CollectedItemRequirement>(parameters...);

processRule(rule3);

}

bool rule1 = false;

short rule2 = false;

short rule3 = false;

bool getRule1(){

return rule1;

}

bool getRule2(){

return rule2;

}

bool getRule3(){

return rule3;

}

void processRule(ExitCellReachRequirement\* rule) {

rule1 = rule->descriptionExitCellReachRequirement();

}

void processRule(EnemiesKilledRequirement\* rule) {

rule2 = rule->descriptionEnemiesKilledRequirement();

}

void processRule(CollectedItemRequirement\* rule) {

rule3 = rule->descriptionCollectedItemRequirement();

}

template <class FindRule>

void\* findRule() {

return nullptr;

}

template <class FindRule, class FirstRule, class... OtherRules>

void\* findRule(FirstRule& first\_rule, OtherRules&... other\_rules)

{

if (typeid(first\_rule) == typeid(FindRule)){

return &first\_rule;

} else{

return findRule<FindRule>(other\_rules...);

}

}

};

ПРИЛОЖЕНИЕ B

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

**Файл** **CollectedItemRequirement.hpp**

#pragma once

class CollectedItemRequirement

{

public:

CollectedItemRequirement(short itemCount) : itemCount(itemCount) { }

short itemCount;

bool descriptionCollectedItemRequirement();

};

ПРИЛОЖЕНИЕ C

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

**Файл ExitCellReachRequirement.hpp**

#pragma once

class ExitCellReachRequirement

{

public:

  ExitCellReachRequirement(bool isTurnedOn) : isTurnedOn(isTurnedOn) { }

  bool isTurnedOn;

  bool descriptionExitCellReachRequirement();

};

ПРИЛОЖЕНИЕ D

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

**Файл EnemiesKilledRequirement.hpp**

#pragma once

class EnemiesKilledRequirement

{

public:

  EnemiesKilledRequirement(short enemyCount) : enemyCount(enemyCount) { }

  short enemyCount;

  bool descriptionEnemiesKilledRequirement();

};