МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №6
по дисциплине «ООП»
Тема: Шаблонные классы

Студент гр. 8383 _____ Шишкин И.В.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2020

Цель работы.

Разработать и реализовать наборы классов правил игры.

Постановка задачи.

Разработка и реализация набора классов правил игры. Основные требования:

Правила игры должны определять начальное состояние игры

Правила игры должны определять условия выигрыша игроков

Правила игры должны определять очередность ходов игрока

Должна быть возможность начать новую игру

Ход работы.

Были реализованы правила игры, добавлена возможность начать новую игру. Передача хода между игроками реализована при помощи паттерна “Состояние”. Класс игры один единственный и создается паттерном “Синглтон”

Описание дополнений к программе.

Реализован класс GameSingleton (файлы GameSingleton.h, GameSingleton.cpp), который реализует паттерн «Синглтон». Он следит за тем, чтобы класс игры был единственным. Так же в методе startGame() этого класса реализована возможность начать новую игру.

Реализован паттерн «Состояние», который следит за ходами игрока. Создан виртуальный класс State (файлы State.h, State.cpp), от него наследуются 2 класса: State 1 – для первого игрока (файлы State1.h, State1.cpp) и State 2 – для второго игрока (файлы State2.h, State2.cpp). В этих классах в методах handle идет проверка на то, принадлежит ли юнит тому игроку, чей сейчас ход. Так же создан класс StateContext (файлы StateContext.h, StateContext.cpp), в котором можно получить текущее состояние, установить его (создана переменная int state, которая обозначает то, какой игрок сейчас ходит. Если она равна 0, то

ходит 1-й игрок. Если она равна 1, то ходит 2-й игрок). Передача хода другому игроку осуществляется методом nextState.

Для правил игры реализован класс Rules, от которого наследуются RuleWith и RuleWithout (файлы Rules.h, RuleWith.h, RuleWith.cpp, RuleWithout.h, RuleWithout.cpp). Существуют 2 вида игры: с базой и без базы. Если в начале игры выбирается игра без базы, то создается экземпляр класса RuleWithout, значение maxNumOfBases принимает значение 0 (обозначает количество баз), значение needToDestroyTheBase принимает значение false. В типе игры без базы сокращено минимальное количество юнитов: теперь количество юнитов должно быть больше 2, а не больше 4, как при игре с базой. Так же, размер поля теперь тоже может быть меньше, так как на поле не нужно располагать базу. Цель игры в данном случае – уничтожить всех юнитов. Если же выбирается тип игры с базой, то создается экземпляр класса RuleWith значение maxNumOfBases принимает значение 2, значение needToDestroyTheBase принимает значение true. Цель игры в данном случае – уничтожить всех юнитов и базу противника.

Реализован шаблонный класс игры, в качестве параметра шаблона передаются конкретные правила.

Тестирование.

Выбор правил игры

```
Какой тип игры вам интересен?
1 - игра без баз
2 - игра с базами
1
Вам предложено ввести количество орков, эльфов и людей.
Так как в игре 2 игрока: один за орков, а другой за людей и эльфов, то
количество людей и эльфов должно соответствовать количеству орков.
(Первый игрок за людей и эльфов, второй - за орков. 2 < Кол-во орков < 10)

Введите количество людей:
2
Введите количество эльфов:
1
Создается поле размерностью GxG. Введите G:
2
G введено неверно! Оно должно быть больше 4
5
Выберите формат логирования:
1 - Логирование в файл
2 - Логирование в терминал
3 - Без логирования
```

Пример выигрыша 1-го игрока в игре без баз посредством удаления юнитов. Логирование приведено в приложении А

```
_____________________
| ~~~ ~~~ $$$ $$$ OD0|
| ~~~ ~~~ $$$ $$$ +AR|
| EA0 ~~~ $$$ $$$ $$$|
| ~~~ ~~~ $$$ $$$ $$$|
| ??? +HP ??? ??? ***|
_____________________



Подсказки:
OD - Orcs Devourers - орки пожиратели; OR - Orcs Robbers - орки разбойники
MS - Mens Spearman - люди копьеносцы; MW - Mens Wizard - люди маги
EA - Elfs Archer - эльф лучник; EI - Elfs Invisible - эльф невидимка
Ландшафт: ~~~ - лес; $$$ - раскаленные земли; ??? - неизвестные земли
Нейтральные объекты: +HP - пополнение здоровья; +AR - пополнение брони; /I\ - убивалка; *** - заморозка
Введите "exit" - для выхода, "add" - для добавления персонажа, "del" - для удаления персонажа,
"save" - для сохранения игры, "load" - для загрузки игры

2-й игрок, ваш ход:
Введите персонажа из орков:
del
Введите обозначение персонажа, которого вы хотите удалить:
OD0

Введите "info", чтобы вывести информацию о юнитах и базе. Введите что угодно, чтобы ничего не выводить
info
Персонаж EA0 : здоровье - 50, броня - 12, урон - 10

Для продолжения нажмите любую клавишу . . .
Победил 1-й игрок!!!
Начать новую игру?
1 - да
2 - нет
```

Выводы.

В ходе выполнения лабораторной работы были реализованы классы правил игры.

ПРИЛОЖЕНИЕ А

Пример логирования

[16:28:27] #GAME: Создана игра с размером поля 5; количество орков - 3; количество людей - 2

[16:28:33] #GAME: Удаление юнита " MS1"

[16:28:40] #GAME: Удаление юнита " OR0"

[16:28:43] #GAME: Удаление юнита " MS0"

[16:28:48] #GAME: Удаление юнита " OD1"

[16:28:52] #GAME: попытка передвижения юнита "EA0" по направлению 4

[16:28:54] #GAME: попытка передвижения юнита "EA0" по направлению 4

[16:28:59] #GAME: Удаление юнита " OD0"

[16:29:04] #GAME: Победа 1-го игрока

[16:29:04] #GAME: Game over!