**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МОЭВМ**

**отчет**

**по лабораторной работе** №**2**

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

**Тема: Интерфейсы классов; взаимодействие классов; перегрузка операций**

Студент гр.8304 \_\_\_\_\_\_\_\_\_\_\_\_\_ Холковский К.В.

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_ Размочаева Н.В.

Санкт-Петербург

2020

**Цель работы.**

Разработать и реализовать набор классов:

* Класс базы
* Набор классов ландшафта карты
* Набор классов нейтральных объектов поля

Класс базы должен отвечать за создание юнитов, а также учитывать юнитов, относящихся к текущей базе. Основные требования к классу база:

* База должна размещаться на поле
* Методы для создания юнитов
* Учет юнитов, и реакция на их уничтожение и создание
* База должна обладать характеристиками такими, как здоровье, максимальное количество юнитов, которые могут быть одновременно созданы на базе, и.т.д.

Набор классов ландшафта определяют вид поля. Основные требования к классам ландшафта:

Должно быть создано минимум 3 типа ландшафта

* Все классы ландшафта должны иметь как минимум один интерфейс
* Ландшафт должен влиять на юнитов (например, возможно пройти по клетке с определенным ландшафтом или запрет для атаки определенного типа юнитов)
* На каждой клетке поля должен быть определенный тип ландшафта

Набор классов нейтральных объектов представляют объекты, располагаемые на поле и с которыми могут взаимодействие юнитов. Основные требования к классам нейтральных объектов поля:

* Создано не менее 4 типов нейтральных объектов
* Взаимодействие юнитов с нейтральными объектами, должно быть реализовано в виде перегрузки операций
* Классы нейтральных объектов должны иметь как минимум один общий интерфейс

**Описание функций и структур данных.**

**struct** Base {  
 **explicit** Base(**int** a = 5);  
 Composite Units;  
 **void** addUnit(Point **const**& a);  
 **void** removeUnit(Point **const**& a);  
 **int** count;  
 **int** max\_count;  
 **int** Hp;  
};

Структура базы с полями здоровья, количества юнитов и максимального количества хранимых в композите юнитов(паттерн “Компоновщик”). Методы добавления и удаления юнитов из базы. Для реакции на удаление и добавление юнитов, был реализован паттерн “Наблюдатель”.

Был создан интерфейс класса ландшафта (Landspace) и три разных типа ландшафта(Grass, Mount, Dust), был реализован паттерн “Прокси”.

Был создан интерфейс класса нейтрального объекта(NtrlObj) и 4-ре типа нейтральных объектов(Oasis, Crocodile, PowerStatue, RangeStatue), для взаимодействия юнитов и нейтральных объектов был реализован паттерн “Стратегия”

Был написан метод Attack для атаки на другого юнита.

**Тестирование**

Были созданы юниты в разных типах ландшафта и показано как ландшафт влияет на юнитов. К примеру, у первого юнита была уменьшена дальность.

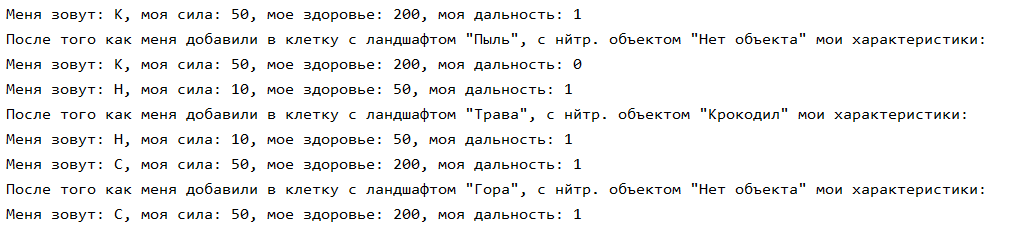


Рисунок 1 – Проверка воздействия ландшафта

Были созданы юниты в точках с разными нейтральными объектами. К примеру, у второго юнита оказался крокодил, но юнит является Healer, поэтому крокодил его не трогает.

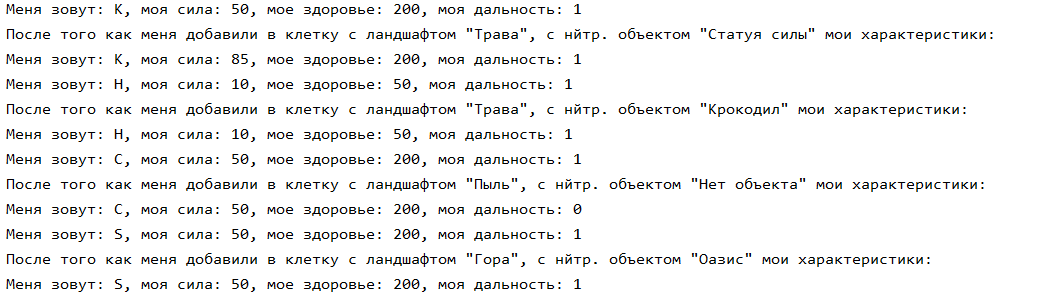


Рисунок 2 – Проверка воздействия нейтральных объектов

Было создано 2 юнита, один ударил другого 3 раза, результат на рисунке 3.



Рисунок 3 – Проверка удара рыцаря

**Выводы.**

Был получен опыт в создании паттернов, таких как паттерн стратегия, наблюдатель, прокси, компоновщик, было реализовано взаимодействие юнитов через перегрузку оператора\*.