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

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Объектно-ориентированное программирование»

Тема: Интерфейсы классов, взаимодействие классов, перегрузка операций

Студент гр. 8383 _____ Мололкин К.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2020

Цель работы.

Продолжить работу над игрой, изучить интерфейсы классов, их взаимодействие и перегрузку операций.

Постановка задачи.

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

Выполнение работы.

class Base : public Observer

Класс базы, описан в файлах Base.h, Base.cpp.

База размещается на клетке игрового поля, на карте размещаться только две базы, база может создавать юнитов, а также хранит информацию о юнитах, созданных на этой базе, так же класс реализует интерфейс наблюдатель, для обновления информации о юнитах.

Поля:

1. int health – здоровье базы;
2. int maxUnitsFromBase – максимальное количество юнитов, созданных на базе;
3. int numUnitsFromBase – текущее количество юнитов на базе;
4. int coordX – координата x базы;
5. int coordY - координата y базы;
6. std::vector<IUnit*> unitsFromBase – вектор юнитов на базе;

Методы:

1. Base(int coordX, int coordY) – конструктор базы;
2. int getX() – геттер для координаты x;
3. int getY() – геттер для координаты y;
4. bool checkCreating() – проверка можно ли создать юнита;
5. IUnit* createUnit(Units unitType, std::pair<int, int> possetion) – функция создания юнита;
6. void deleteUnit(IUnit* unit) – функция удаления юнита;
7. void updateNotify(IUnit* unit, Event event) override – функция для обновления информации о юнитах;

Классы ландшафтов

enum class Landscapes – класс перечислений всех видов ландшафтов

class ILandscape – интерфейс ландшафтов

методы:

1. getLandscapeType() - возвращает тип ланшафта;
2. bool canGo() – возвращает возможность прохода по ландшафту;

class Swamp – ландшафт болота, реализует интерфейс ILandscape;

class Plain – ландшафт болота, реализует интерфейс ILandscape;

class Hills – ландшафт болота, реализует интерфейс ILandscape;

class LandscapeFabric – фабрика ландшафтов

class LandscapeProxy – класс для реализации паттерна прокси

поля:

1. ILandscape* landscape – указательна на ландшафт;

методы:

1. LandscapeProxy(Landscapes landscapeType) – конструктор, принимает тип ландшафта и с помощью фабрики определяет поле landscape ;
2. Landscapes getLandscapeType() – возвращает тип ландшафта ;
3. bool canGo() - возвращает возможность прохода по ландшафту;
4. void landscapeInfluence(IUnit* unit) – функция для взаимодействия ландшафта и юнита;

Классы нейтральных объектов

enum class NeutralObject – класс перечислений всех типов нейтральных объектов.

class INeutralObject – интерфейс нейтральных объектов

методы:

1. virtual NeutralObject getType() – возвращает тип нейтрального объекта;

class Hospital : public INeutralObject – класс больница, увеличивает здоровье юнита.

поля:

1. const int HPToAdd = 20 – количество здоровья восстанавливаемое в больнице;

методы:

1. NeutralObject getType() возвращает тип нейтрального объекта;
2. friend IUnit& operator+=(IUnit& left, const Hospital& right) перегрузка оператора для взаимодействия юнит и нейтрального обьекта.

class AttackUpgrader:public INeutralObject – нейтральный объект увеличивающий атаку юнита;

class ArmourUpgrader:public INeutralObject - нейтральный объект увеличивающий защиту юнита.

class Spikes : public INeutralObject - нейтральный объект шипов, отнимает здоровье юнита.

Паттерн наблюдатель

enum class Event – класс перечислений возможных действий

class Observer – интерфейс наблюдатель

методы:

1. virtual void updateNotify(IUnit* unit, Event event), метод для обновления информации о юнитах

class ObserverManager – класс для хранения подписчиков и рассылки изменений

поля:

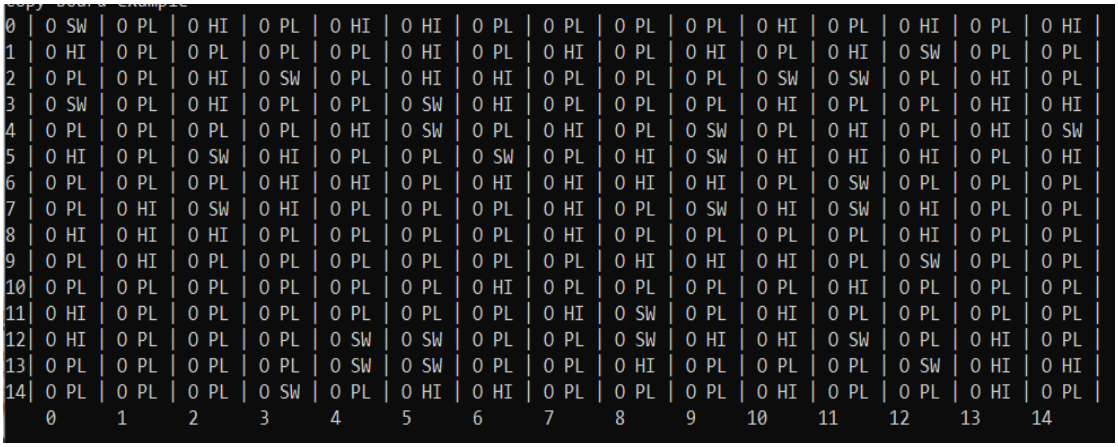
1. std::vector<Observer*> observers – вектор для хранения всех подписчиков.

методы:

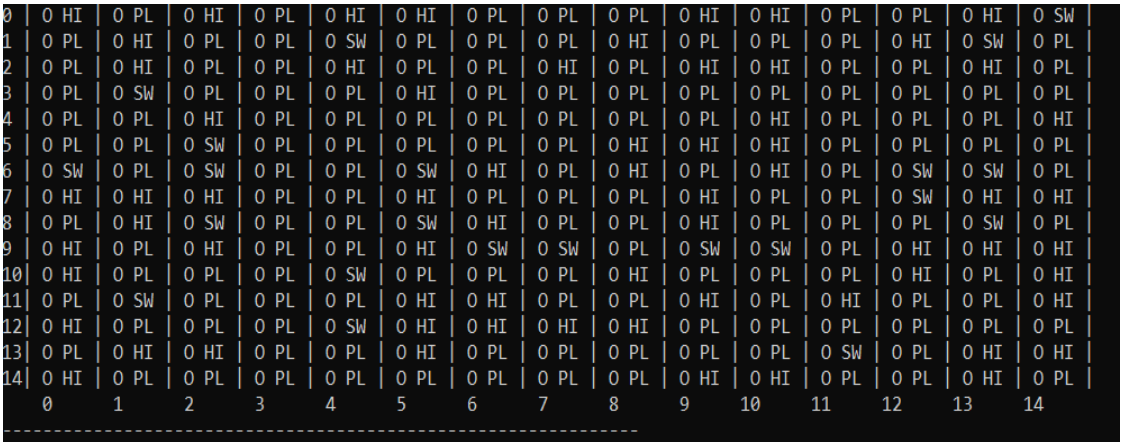
1. ObserverManager() = default - конструктор;
2. void addObserver(Observer* observer) – добавление подписчика;
3. void removeObserver(Observer* observer) – удаление подписчика;
4. void notify(IUnit* unit, Event event) – рассылка изменения;

Примеры:

1. Рандомная генерация ландшафтов (ландшафт – правая часть клетки):

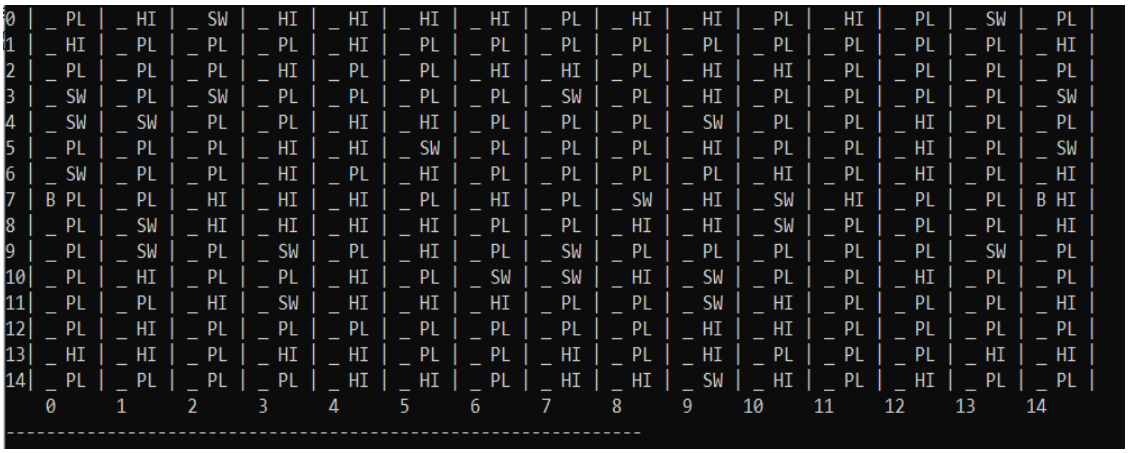


	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	O SW	O PL	O HI	O PL	O HI	O HI	O PL	O PL	O PL	O HI	O PL	O HI	O PL	O HI	
1	O HI	O PL	O PL	O PL	O PL	O HI	O PL	O HI	O PL	O HI	O PL	O HI	O SW	O PL	O PL
2	O PL	O PL	O HI	O SW	O PL	O HI	O HI	O PL	O PL	O PL	O SW	O SW	O PL	O HI	O PL
3	O SW	O PL	O HI	O PL	O PL	O SW	O HI	O PL	O PL	O PL	O HI	O PL	O PL	O HI	O HI
4	O PL	O PL	O PL	O PL	O HI	O SW	O PL	O HI	O PL	O SW	O PL	O HI	O PL	O HI	O SW
5	O HI	O PL	O SW	O HI	O PL	O PL	O SW	O PL	O HI	O SW	O HI	O HI	O HI	O PL	O HI
6	O PL	O PL	O PL	O HI	O HI	O PL	O HI	O HI	O HI	O HI	O PL	O SW	O PL	O PL	O PL
7	O PL	O HI	O SW	O HI	O PL	O PL	O PL	O HI	O PL	O SW	O HI	O SW	O HI	O PL	O PL
8	O HI	O HI	O HI	O PL	O PL	O PL	O PL	O HI	O PL	O PL	O PL	O PL	O HI	O PL	O PL
9	O PL	O HI	O PL	O PL	O PL	O PL	O PL	O PL	O HI	O HI	O HI	O PL	O SW	O PL	O PL
10	O PL	O PL	O PL	O PL	O PL	O PL	O HI	O PL	O PL	O PL	O PL	O HI	O PL	O PL	O PL
11	O HI	O PL	O PL	O PL	O PL	O PL	O PL	O HI	O SW	O PL	O HI	O PL	O PL	O PL	O PL
12	O HI	O PL	O PL	O PL	O SW	O SW	O PL	O PL	O SW	O HI	O HI	O SW	O PL	O HI	O PL
13	O PL	O PL	O PL	O PL	O SW	O SW	O PL	O PL	O HI	O PL	O PL	O PL	O SW	O HI	O HI
14	O PL	O PL	O PL	O SW	O PL	O HI	O HI	O PL	O PL	O PL	O HI	O PL	O PL	O HI	O PL



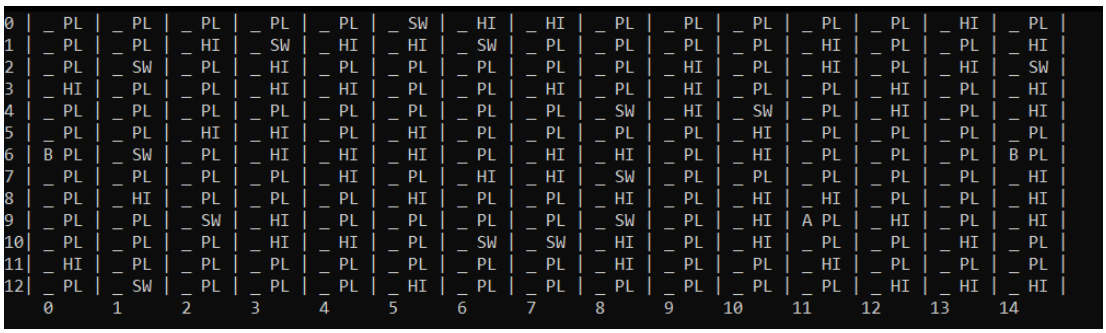
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	O HI	O PL	O HI	O PL	O HI	O HI	O PL	O PL	O PL	O HI	O HI	O PL	O PL	O HI	O SW
1	O PL	O HI	O PL	O PL	O SW	O PL	O PL	O PL	O HI	O PL	O PL	O PL	O HI	O SW	O PL
2	O PL	O HI	O PL	O PL	O HI	O PL	O PL	O HI	O PL	O HI	O HI	O PL	O PL	O HI	O PL
3	O PL	O SW	O PL	O PL	O PL	O HI	O PL	O PL	O PL	O PL	O PL	O PL	O PL	O PL	O PL
4	O PL	O PL	O HI	O PL	O PL	O PL	O PL	O PL	O PL	O PL	O HI	O PL	O PL	O PL	O HI
5	O PL	O PL	O SW	O PL	O PL	O PL	O PL	O PL	O HI	O HI	O HI	O PL	O PL	O PL	O PL
6	O SW	O PL	O SW	O PL	O PL	O SW	O HI	O PL	O HI	O PL	O HI	O PL	O SW	O SW	O PL
7	O HI	O HI	O HI	O PL	O PL	O HI	O PL	O PL	O PL	O HI	O PL	O PL	O SW	O HI	O HI
8	O PL	O HI	O SW	O PL	O PL	O SW	O HI	O PL	O PL	O HI	O PL	O PL	O PL	O SW	O PL
9	O HI	O PL	O HI	O PL	O PL	O HI	O SW	O SW	O PL	O SW	O SW	O PL	O HI	O HI	O HI
10	O HI	O PL	O PL	O PL	O SW	O PL	O PL	O PL	O HI	O PL	O PL	O PL	O HI	O PL	O HI
11	O PL	O SW	O PL	O PL	O PL	O HI	O HI	O PL	O PL	O HI	O PL	O HI	O PL	O PL	O HI
12	O HI	O PL	O PL	O PL	O SW	O HI	O HI	O HI	O HI	O PL	O PL	O PL	O PL	O PL	O PL
13	O PL	O HI	O HI	O PL	O PL	O HI	O PL	O PL	O PL	O PL	O PL	O SW	O PL	O HI	O HI
14	O HI	O PL	O PL	O PL	O PL	O PL	O PL	O PL	O PL	O HI	O HI	O PL	O PL	O HI	O PL

2. В начале игры будут создаваться две базы



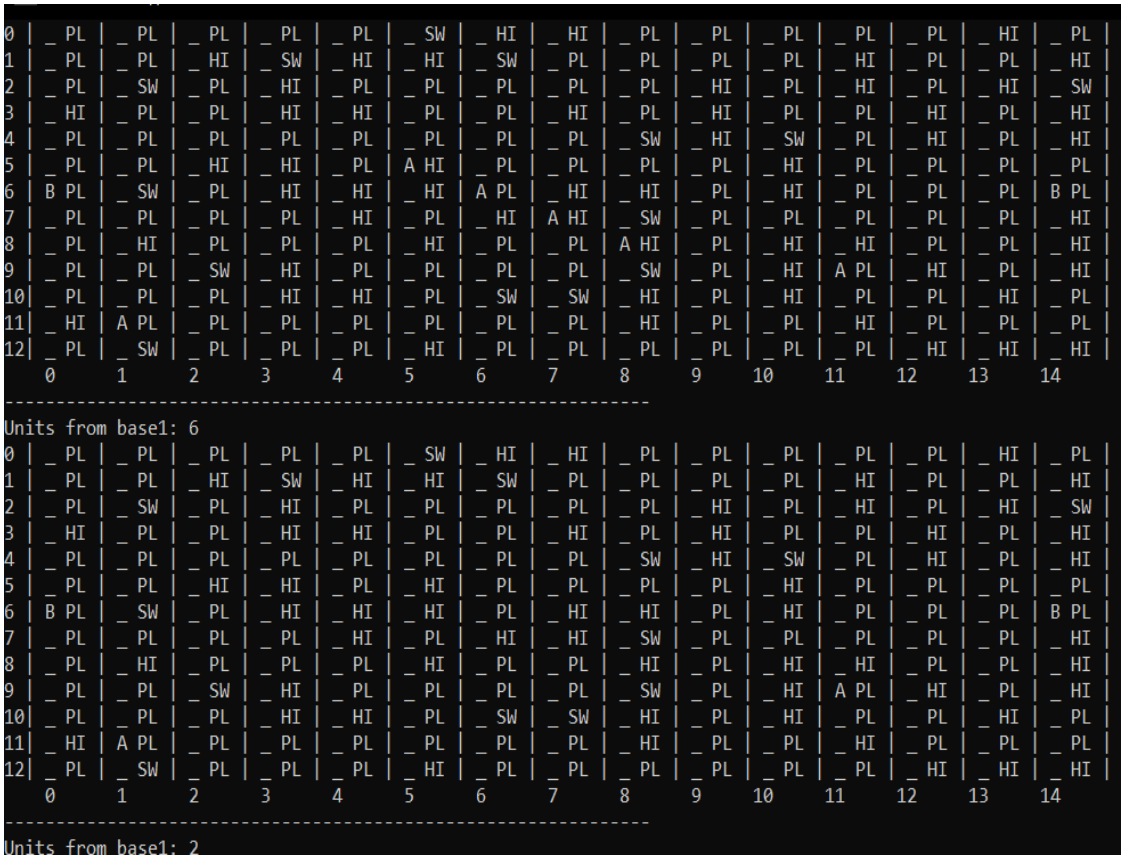
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	_ PL	_ HI	_ SW	_ HI	_ HI	_ HI	_ HI	_ PL	_ HI	_ HI	_ PL	_ HI	_ PL	_ SW	_ PL
1	_ HI	_ PL	_ PL	_ PL	_ HI	_ PL	_ PL	_ PL	_ PL	_ PL	_ PL	_ PL	_ PL	_ PL	_ HI
2	_ PL	_ PL	_ PL	_ HI	_ PL	_ PL	_ HI	_ HI	_ PL	_ HI	_ HI	_ PL	_ PL	_ PL	_ PL
3	_ SW	_ PL	_ SW	_ PL	_ PL	_ PL	_ PL	_ SW	_ PL	_ HI	_ PL	_ PL	_ PL	_ PL	_ SW
4	_ SW	_ SW	_ PL	_ PL	_ HI	_ HI	_ PL	_ PL	_ PL	_ SW	_ PL	_ PL	_ HI	_ PL	_ PL
5	_ PL	_ PL	_ PL	_ HI	_ HI	_ SW	_ PL	_ PL	_ PL	_ HI	_ PL	_ PL	_ HI	_ PL	_ SW
6	_ SW	_ PL	_ PL	_ HI	_ PL	_ HI	_ PL	_ PL	_ PL	_ PL	_ HI	_ PL	_ HI	_ PL	_ HI
7	B PL	_ PL	_ HI	_ HI	_ HI	_ PL	_ HI	_ PL	_ SW	_ HI	_ SW	_ HI	_ PL	_ PL	B HI
8	_ PL	_ SW	_ HI	_ HI	_ HI	_ HI	_ PL	_ PL	_ HI	_ HI	_ SW	_ PL	_ PL	_ PL	_ HI
9	_ PL	_ SW	_ PL	_ SW	_ PL	_ HI	_ PL	_ SW	_ PL	_ PL	_ PL	_ PL	_ PL	_ SW	_ PL
10	_ PL	_ HI	_ PL	_ PL	_ HI	_ PL	_ SW	_ SW	_ HI	_ SW	_ PL	_ PL	_ HI	_ PL	_ PL
11	_ PL	_ PL	_ HI	_ SW	_ HI	_ HI	_ HI	_ PL	_ PL	_ SW	_ HI	_ PL	_ PL	_ PL	_ HI
12	_ PL	_ HI	_ PL	_ PL	_ PL	_ PL	_ PL	_ PL	_ PL	_ HI	_ HI	_ PL	_ PL	_ PL	_ PL
13	_ HI	_ HI	_ PL	_ HI	_ HI	_ PL	_ PL	_ HI	_ PL	_ HI	_ PL	_ PL	_ PL	_ HI	_ HI
14	_ PL	_ PL	_ PL	_ PL	_ HI	_ HI	_ PL	_ HI	_ HI	_ SW	_ HI	_ PL	_ HI	_ PL	_ PL

3. Создадим два юнита одного на клетке с ландшафтом поле (11, 9), а другого на болоте (1, 12)



как видим на клетке болото, юнит не создался, потому что по нему нельзя ходить.

4. пример работы наблюдателя, до удаления юнитов на базе было 6, после стало 2



Выводы.

В ходе выполнения данной лабораторной работы были изучены интерфейсы классов, их взаимодействие и перегрузка операций. Написаны классы базы, нейтральных объектов и ландшафтов, реализованы паттерны наблюдатель и прокси.