

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

ОТЧЕТ
по лабораторной работе №3
по дисциплине «Объектно-ориентированное программирование»
Тема: Логическое разделение классов

Студентка гр. 8383

Максимова А.А.

Преподаватель

Жангиров Т.Р.

Санкт-Петербург

2020

Цель работы.

Разработать и реализовать набор классов для взаимодействия пользователя с юнитами и базой.

Постановка задачи.

- Должен быть реализован функционал управления юнитами
- Должен быть реализован функционал управления базой

Выполнены все основные требования к взаимодействию	6 баллов
Добавлен функционал просмотра состояния базы	3 балла
Имеется 3+ демонстрационных примера	1 балл
<i>*Реализован паттерн “Фасад” через который пользователь управляет программой</i>	<i>1 балл</i>
<i>*Объекты между собой взаимодействуют через паттерн “Посредника”</i>	<i>3 балла</i>
<i>*Для передачи команд используется паттерн “Команда”</i>	<i>3 балла</i>
<i>*Для приема команд от пользователя используется паттерн “Цепочка обязанностей”</i>	<i>3 балла</i>
Кол-во баллов за основные требования	10 баллов
Максимальное кол-во баллов за лаб. работу	20 баллов

Абстрактный класс Functh.

Абстрактный класс, от которого наследуются конкретные классы: UnitsFunct, представляющий собой реализацию функционала управления юнитами и BaseFuncth, необходимый для реализации управления базой и просмотра ее состояния.

Поля:

- String command; - содержит команду, введенную пользователем по шаблону, задаваемым в зависимости от выбранной функции управления.
- Char act; - содержит сокращенное название выбранной функции управления.
- Int x, y; - координаты расположения юнита или базы.
- int Gamer; - номер игрока (важно, когда игрок хочет переместить не своих юнитов, находясь в режиме настроек).
- Gamer* pointer; - указатель на объект класса Gamer, для обращения к его методам.

Методы (будут переопределены как):

- void setPointer(Gamer* gamer); - сеттер для pointer.
- void setParam(string cmmnd); - сеттер для command.
- void recognizer(); - для сопоставления запроса пользователя с методом класса.

Функционал управления юнитами.

Для реализации функционала управления юнитами был создан класс UnitsFunct, наследуемый от абстрактного класса Functh. Функции в данном классе разделяются по моменту вызова: режим настроек или режим игры.

Функции, предоставляемые пользователю для работы с юнитами:

Режим настроек:

1. Помощь: для подробного объяснения правил вызова функции с помощью команд. Метод void help();
2. Перемещение (своего) юнита на другую позицию в пределах своего поля. Команда: начальные координаты юнита, направление движения, величина шага. Метод void movUnit();
3. Информация о юните – вывод параметров здоровья, урона и брони. Команда: координаты юнита. Метод void infoUnit();
4. Swap юнитов – поменять друг с другом два юнита местами. Метод void swapUnit();

Режим игры:

1. Перемещение юнита (+атака): в данной игре атака производится наступлением на юнита, поэтому для нее также используется метод для перемещения. При этом, если юнит хочет переместиться на пустую позицию, то атаки соответственно не происходит. Атака реализована с помощью перегрузки оператора -=. Во время режима настроек нападение запрещено, используется флаг int flagAttack. Метод void movUnit();

Примечание: методы, исполняющий заданные команды были реализованы в прошлых работах.

Функционал управления базой.

Для реализации управления базой был написан класс BaseFunct, наследуемый от абстрактного класса Funct. Функции в данном классе разделяются на управление базой и просмотр состояния базы.

Просмотр состояния базы:

1. Вывод информации о базе Монстров. Метод infoBase().
2. Вывод информации о базе Героев. Метод infoBase().
3. Вывод информации о базе людей. Метод infoBase().

Под информацией в данном случае имеются в виду такие параметры, как: максимальное количество юнитов, относящихся к текущей базе; максимальное количество юнитов, создаваемых базой во время игры; здоровье базы.

Управление базой:

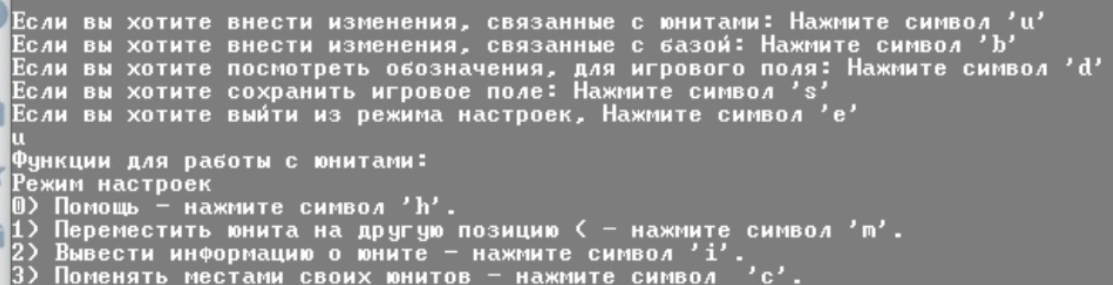
1. Восстановить состояние базы – то есть юнит, может увеличить здоровье своей базы ровно на величину своего здоровья (без ущерба для себя), затратив на это один ход. Метод recoveryBase();
2. Напасть на базу – идея аналогична нападению на юнитов. Так как здоровье у базы много больше, чем у любого юнита, то такие нападения должны быть не едино разовыми. При этом разрушение базы не является автоматическим проигрышем, но пополнение команды происходить соответственно не будет, поэтому рекомендуется не пренебрегать первой командой. Метод attackBase();

Реализация паттерна “Фасад”.

Для реализации фасада был написан класс Facade, который принимая команды от пользователя перенаправляет их в один из классов: BaseFunct или UnitFunct. Класс фасада предоставляет упрощенный интерфейс более сложной логики, взаимодействие с фасадом можно описать так: выбор одной из категорий – ‘u’ – функции для юнитов, ‘b’ – функции для базы, пользователь принимает решения и в ответ получает список имеющихся команд (методы listOptionBase, listOptionUnit).

В зависимости от выбранной команды, пользователь получает шаблон (описаны в методах shablonCommandBase, shablonCommandUnit), по которому нужно ввести команду. Далее введенная команда обрабатывается (считывается) с помощью метода readerCommands и посылается на выполнение. Кроме того описаны методы checkListBase, checkListUnit, используемые для проверки корректности выбранной команды.

Демонстрационные примеры.



```
Если вы хотите внести изменения, связанные с юнитами: Нажмите символ 'u'
Если вы хотите внести изменения, связанные с базой: Нажмите символ 'b'
Если вы хотите посмотреть обозначения, для игрового поля: Нажмите символ 'd'
Если вы хотите сохранить игровое поле: Нажмите символ 's'
Если вы хотите выйти из режима настроек, Нажмите символ 'e'
u
Функции для работы с юнитами:
Режим настроек
0) Помощь – нажмите символ 'h'.
1) Переместить юнита на другую позицию < – нажмите символ 'm'.
2) Вывести информацию о юните – нажмите символ 'i'.
3) Поменять местами своих юнитов – нажмите символ 'c'.
```

Рисунок 1 – Меню

Если вы хотите посмотреть обозначения, для игрового поля: Нажмите символ 'd'
 Если вы хотите сохранить игровое поле: Нажмите символ 's'
 Если вы хотите выйти из режима настроек, Нажмите символ 'e'
 и
 Функции для работы с юнитами:
 Режим настроек
 0) Помощь – нажмите символ 'h'.
 1) Переместить юнита на другую позицию < – нажмите символ 'm'.
 2) Вывести информацию о юните – нажмите символ 'i'.
 3) Поменять местами своих юнитов – нажмите символ 'c'.
 h
 Пожалуйста, строго следуйте формату.
 Формат команды: отсутствует.
 Для того, чтобы переместить юнита на другую позицию, необходимо ввести:
 Текущие координаты расположения юнита (x, y);
 Направление движения юнитов:
 "l" – влево; "r" – вправо; "u" – вверх; "d" – вниз
 "a" – диагональ вверх-направо; "b" – диагональ вверх-налево;
 "c" – диагональ вниз-направо; "f" – диагональ вниз-налево.
 Количество шагов (из допустимого).
 Допустимое количество шагов:
 Вампир : от 1 до 3.
 Оборотень: от 1 до 2.
 Целитель: 1 шаг.
 Маг: от 1 до 2 шагов.
 Для того, чтобы узнать информацию о юните нужно ввести координаты его местоположения
 Для того, чтобы поменять юнитов местами нужно указать координаты их местоположений.
 Атака доступна только в режиме игры, чтобы напасть, достаточно переместиться на коорд
 Если вы хотите внести изменения, связанные с юнитами: Нажмите символ 'u'
 Если вы хотите внести изменения, связанные с базой: Нажмите символ 'b'
 Если вы хотите посмотреть обозначения, для игрового поля: Нажмите символ 'd'
 Если вы хотите сохранить игровое поле: Нажмите символ 's'
 Если вы хотите выйти из режима настроек, Нажмите символ 'e'

Рисунок 2 – help()

	0	1	2	3	4	5	6	7	8	9
0	■	■	■	■	■	■	■	■	■	■
1	■	■	■	■	■	■	■	■	■	■
2	■	■	■	■	■	■	■	■	■	■
3	■	■	■	■	■	■	■	■	■	■
4	■	■	■	■	■	■	■	■	■	■
5	■	■	■	■	■	■	■	■	■	■
6	■	■	■	■	■	■	■	■	■	■
7	■	■	■	■	■	■	■	■	■	■
8	■	■	■	■	■	■	■	■	■	■
9	■	■	■	■	■	■	■	■	■	■

Счетчик юнитов:

.People = 12

.Monster = 6

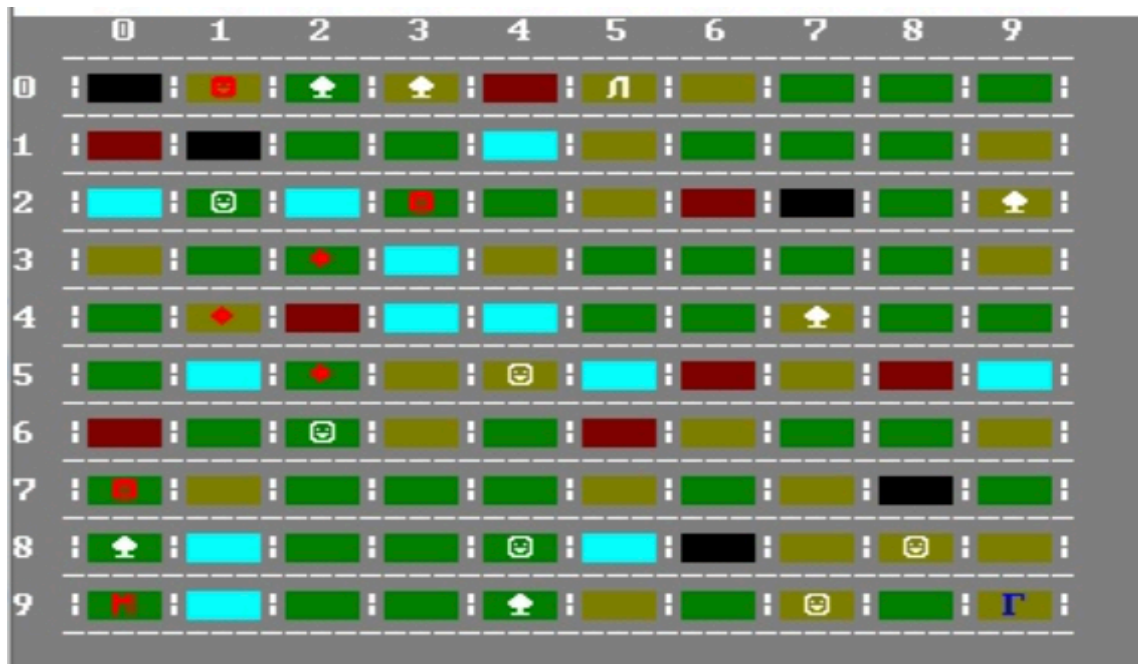
.Hero = 0

Если вы хотите внести изменения, связанные с юнитами: Нажмите символ 'u'
 Если вы хотите внести изменения, связанные с базой: Нажмите символ 'b'
 Если вы хотите посмотреть обозначения, для игрового поля: Нажмите символ 'd'
 Если вы хотите сохранить игровое поле: Нажмите символ 's'
 Если вы хотите выйти из режима настроек, Нажмите символ 'e'
 и

```

Функции для работы с юнитами:
Режим настроек
0) Помощь - нажмите символ 'h'.
1) Переместить юнита на другую позицию < - нажмите символ 'm'.
2) Вывести информацию о юните - нажмите символ 'i'.
3) Поменять местами своих юнитов - нажмите символ 'c'.
Пожалуйста, строго следуйте формату.
Формат команды: x, y, направление, количество шагов.
5, 0, r, 2.

```



Рисунки 3-5: Перемещение юнита

```

Функции для работы с юнитами:
Режим настроек
0) Помощь - нажмите символ 'h'.
1) Переместить юнита на другую позицию < - нажмите символ 'm'.
2) Вывести информацию о юните - нажмите символ 'i'.
3) Поменять местами своих юнитов - нажмите символ 'c'.
i
Команда не доступна
Пожалуйста, строго следуйте формату.
Формат команды: x, y.
7, 0.
Здоровье: 1000 единиц.
Урон: 294 единиц.
Броня: 394 единиц.

```

Рисунок 6 – infoUnit()

```

Пожалуйста, строго следуйте формату.
База монстров
Здоровье: 10000 единиц.
Максимальное количество юнитов, относящихся к одной базе 6 единиц.
Максимальное количество юнитов, создаваемое во время игры 2 единиц.

```

Рисунок 7 – infoBase()

Выводы.

В результате выполнения лабораторной работы были написаны классы, реализующий взаимодействие пользователя с юнитами и базой через фасад.