

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

отчет
по лабораторной работе №1
по дисциплине «Объектно-ориентированное программирование»
Тема: Создание классов, конструкторов классов, методов классов,
наследование

Студент гр.8304

Преподаватель

Холковский К.В.

Размочаева Н.В.

Санкт-Петербург

2020

Цель работы.

Разработать и реализовать набор классов:

- Класс игрового поля
- Набор классов юнитов

Игровое поле является контейнером для объектов представляющим прямоугольную сетку. Основные требования к классу игрового поля:

- Создание поля произвольного размера
- Контроль максимального количества объектов на поле
- Возможность добавления и удаления объектов на поле
- Возможность копирования поля (включая объекты на нем)
- Для хранения запрещается использовать контейнеры из stl

Юнит является объектом, размещаемым на поле боя. Один юнит представляет собой отряд. Основные требования к классам юнитов:

- Все юниты должны иметь как минимум один общий интерфейс
- Реализованы 3 типа юнитов (например, пехота, лучники, конница)
- Реализованы 2 вида юнитов для каждого типа (например, для пехоты могут быть созданы мечники и копейщики)
- Юниты имеют характеристики, отражающие их основные атрибуты, такие как здоровье, броня, атака.
- Юнит имеет возможность перемещаться по карте

Описание функций и структур данных.

```
struct Field {  
  
    struct element  
  
    struct iterator: std::iterator<std::bidirectional_iterator_tag, element  
    iterator begin()  
    iterator end()  
    struct SomeStruct  
    SomeStruct operator[](int a);  
  
    explicit Field(int s = 10, int l = 10);  
    Field(Field const& a);  
    Field(Field&& a) noexcept;  
    ~Field();  
    void addUnit(Point const& a, char Name);  
    void remUnit(Point const &a);  
    bool move(Point const& old_p, Point const& new_p );  
    void show();  
};
```

```

    int height;
    int width;
    int MaxCount;
    int Count;
private:
    element** field;
};

```

Структура Field – где хранится высота, ширина и содержимое поля, также максимальное и актуальное количество юнитов на поле. Здесь же реализованы методы добавления, удаления и движения юнитов на поле, конструкторы по умолчанию, копирования и перемещения. Был перегружен operator[] и создан итератор по полю.

```

struct element {
    element();
    element(element const& a);
    ~element() = default;
    std::unique_ptr<Unit> unit;
};

```

Структура element – где хранится юнит и реализованы конструктор по умолчанию и конструктор копирования.

Так же были реализованы 3 типа юнитов(мечник, стрело, маг) и 2 вида юнитов для каждого типа(рыцарь, “щитовик”, лучник, следопыт, “кастер”, лекарь)

Для создания юнитов используется паттерн абстрактная фабрика

Тестирование

1) Добавление 3 разных юнитов на поле 5*5:

```

K * * * *
* * * * *
* * * H *
* * * * *
* R * * *

```

2) Удаление юнита в {0,0} и перемещение юнита из {3,2} в {4, 4}

```

* * * * *
* * * * *
* * * * *
* * * * *
* R * * H

```

3)Использование range based for:

```
int k=0;  
for(auto i:b){  
    ++k;  
}  
std::cout << std::endl << k;
```

Выводится 25, т.к. размерность 5*5.

Выводы.

Были выполнены все основные и дополнительные требования работы