



**«Московский государственный технический университет  
имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)**

---

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Компьютерные системы и сети

**О т ч е т**

**по лабораторной работе № 4**

**Дисциплина:** Современные средства разработки программного обеспечения

**Название лабораторной работы:** Рефакторинг. Выделение сущностей, значений и служб модели

Студент гр. ИУ6-56

\_\_\_\_\_  
(Подпись, дата)

Т.А. Тищенко

\_\_\_\_\_  
(И.О. Фамилия)

Преподаватель

\_\_\_\_\_  
(Подпись, дата)

М.В. Фетисов

\_\_\_\_\_  
(И.О. Фамилия)

Цель работы: Приобрести навыки оценки необходимых изменений при уточнении модели предметной области.

Задание: Необходимо определить степень увеличения загрязнения населённых пунктов, которое может произойти из-за выброса ядовитых веществ на предприятии в одном из них. При этом увеличение загрязнения рассчитывается по степени удалённости населённого пункта от эпицентра (обратно квадрату расстояния). Получите отчёт о 5 наиболее загрязнённых населённых пунктах.

Задача: «Война в долине теней»

Карточка: «Воин Долины теней». Карточка должна содержать поля: фракция (перечисляемый тип), сила удара, защита, здоровье, ловкость, уклонение, тип боя (ближний, дальний).

Код программы:

```
class BasicStats {
    int _power, _protection, _health, _skill, _deviation;

protected:
    bool invariant() const
    {
        return _power >= 0 && _power <= 9
            && _protection >= 0 && _protection <= 9
            && _health >= 0 && _health <= 9
            && _skill >= 0 && _skill <= 9
            && _deviation >= 0 && _deviation <= 9;
    }

public:
    BasicStats() = delete;
    BasicStats(int power, int protection, int health, int skill,
        int deviation): _power(power),
        _protection(protection), _health(health), _skill(skill),
        _deviation(deviation) {
        assert(invariant());
    }

    int getPower() const { return _power; }
    int getProtection() const { return _protection; }
    int getHealth() const { return _health; }
    int getSkill() const { return _skill; }
    int getDeviation() const { return _deviation; }

    string getBasicStatsString() const {
        return to_string(_power) + " " +
            to_string(_protection) + " " + to_string(_health) + " " +
            to_string(_skill) + " " + to_string(_deviation);
    };
};
```

```

class WarriorVS : public ICollectable {
    int _group;
    int _battle_type;

protected:
    bool invariant() const {
        return _group >= 1 && _group <= 5
            && _battle_type >= 1 && _battle_type <= 2;
    }

public:
    WarriorVS() = delete;
    WarriorVS(const WarriorVS & p) = delete;
    WarriorVS & operator = (const WarriorVS & p) = delete;

    WarriorVS(int group, BasicStats stats, int battle_type) :
        _group(group),
        _stats(stats),
        _battle_type(battle_type){
        assert(invariant());
    }

    string getOutputString() const {
        return to_string(_group) + " " + _stats.getBasicStatsString() + " " +
            to_string(_battle_type);
    }

    virtual bool write(ostream& os) override {
        writeNumber(os, _group);
        writeNumber(os, _stats.getPower());
        writeNumber(os, _stats.getProtection());
        writeNumber(os, _stats.getHealth());
        writeNumber(os, _stats.getSkill());
        writeNumber(os, _stats.getDeviation());
        writeNumber(os, _battle_type);

        return os.good();
    }
};

class ItemCollector: public ACollector {
public:
    virtual shared_ptr

```



Вывод программы:

```
a 2 2 2 5 3 2 1
update 0 1 10 9 10 10 10 2
v
[0] 1 10 9 10 10 10 2
Количество элементов в коллекции: 1
s
```

Выполнение завершено успешно

Вывод: Были приобретены навыки оценки необходимых изменений при уточнении модели предметной области.