МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский университет ИТМО»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И
КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине

«ПРОГРАММИРОВАНИЕ»

Вариант №3

Выполнила:
Студентка группы P3114
Зуйкова Елизавета Владимировна
Проверил преподаватель:
Бобрусь Александр Владимирович

Санкт-Петербург, 2025

Оглавление

Задание

1. Ознакомиться с документацией, обращая особое внимание на классы Pokemon и Move. При дальнейшем выполнении лабораторной работы читать документацию еще несколько раз.
2. Скачать файл Pokemon.jar. Его необходимо будет использовать как для компиляции, так и для запуска программы. Распаковывать его не надо! Нужно научиться подключать внешние jar-файлы к своей программе.
3. Написать минимально работающую программу и посмотреть как она работает.

```
Battle b = new Battle();
Pokemon p1 = new Pokemon("Чужой", 1);
Pokemon p2 = new Pokemon("Хищник", 1);
b.addAlly(p1);
b.addFoe(p2);
b.go();
```

4. Создать один из классов покемонов для своего варианта. Класс должен наследоваться от базового класса Pokemon. В конструкторе нужно будет задать типы покемона и его базовые характеристики. После этого попробуйте добавить покемона в сражение.
5. Создать один из классов атак для своего варианта (лучше всего начать с физической или специальной атаки). Класс должен наследоваться от класса PhysicalMove или SpecialMove. В конструкторе нужно будет задать тип атаки, ее силу и точность. После этого добавить атаку покемону и проверить ее действие в сражении. Не забудьте переопределить метод describe, чтобы выводилось нужное сообщение.
6. Если действие атаки отличается от стандартного, например, покемон не промахивается, либо атакующий покемон также получает повреждение, то в классе атаки нужно дополнительно переопределить соответствующие методы (см. документацию). При реализации атак, которые меняют статус покемона (наследники StatusMove), скорее всего придется разобраться с классом Effect. Он позволяет на один или несколько ходов изменить состояние покемона или модификатор его базовых характеристик.
7. Доделать все необходимые атаки и всех покемонов, распределить покемонов по командам, запустить сражение.

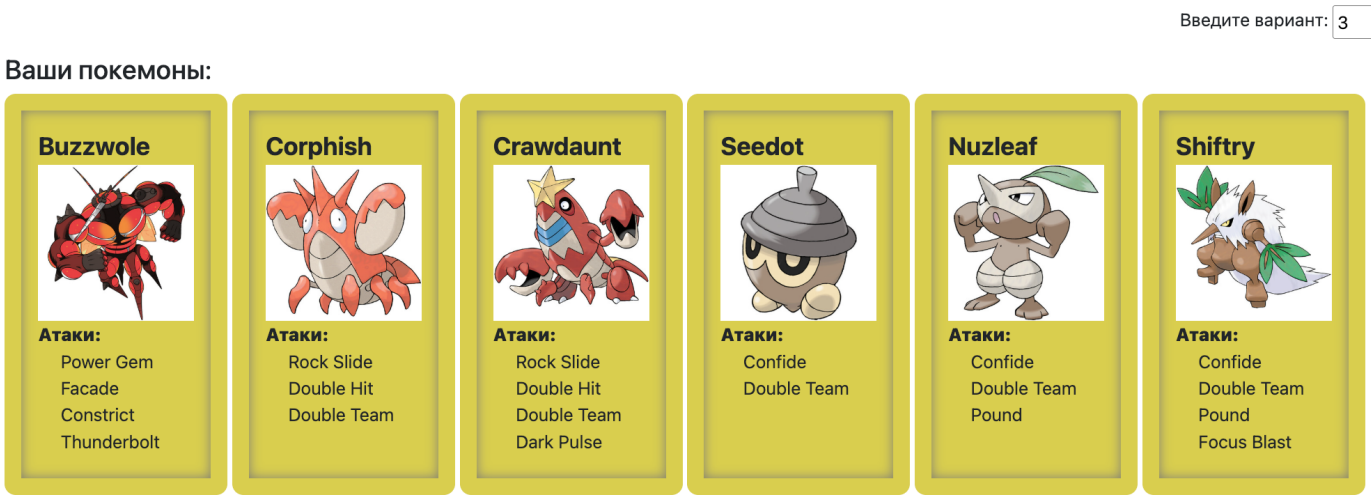

Рисунок 1 – покемоны

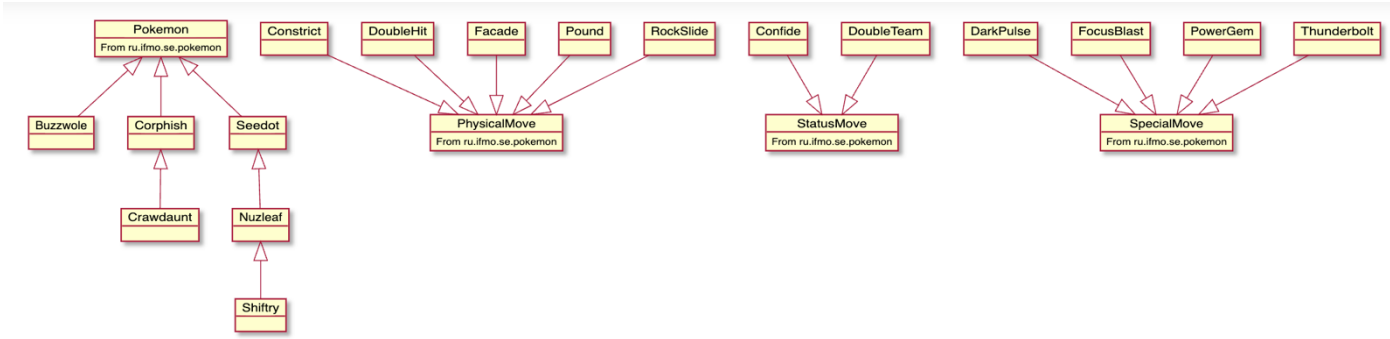

Рисунок 2 – диаграмма классов

Исходный код программы

```java
package PhMove;

import ru.ifmo.se.pokemon.PhysicalMove;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Stat;
import ru.ifmo.se.pokemon.Type;

public class Constrict extends PhysicalMove {  2 usages
    public Constrict(){  1 usage
        super(Type.NORMAL,  v: 10,  v1: 100);
    }
    @Override
    protected String describe() {
        return "использует Constrict";
    }
    // наносит урон и с вероятностью 10% снижает Скорость цели на одну ступень.
    @Override  no usages
    protected void applyOppEffects(Pokemon p) { //
        if (Math.random()<=0.1){
            p.setMod(Stat.SPEED,  i: -1);
        }
    }

}
```

Рисунок 3 – Constrict.java

```java
package PhMove;

import ru.ifmo.se.pokemon.PhysicalMove;
import ru.ifmo.se.pokemon.Type;

public class DoubleHit extends PhysicalMove {  2 usages
    public DoubleHit(){  1 usage
        //наносит урон и ударит дважды
        super(Type.NORMAL,  v: 35,  v1: 90,  i: 0,  i1: 2); // 0 - приоритетность стандартная
    }
    @Override
    protected String describe() {
        return "использует Double Hit";
    }

}
```

Рисунок 4 – DoubleHit.java

```java
package PhMove;

import ru.ifmo.se.pokemon.*;

public class Facade extends PhysicalMove {  2 usages

    public Facade(){  1 usage
        super(Type.NORMAL,  v: 70,  v1: 100);
    }
    @Override
    protected String describe() {
        return "использует Facade";
    }
    //наносит урон и бьет с удвоенной силой (140), если пользователь обожжен, отравлен или парализован.
    @Override  no usages
    protected double calcBaseDamage(Pokemon att, Pokemon def) { //
        if ((def.getCondition()== Status.BURN) || (def.getCondition()==Status.PARALYZE) ||
                (def.getCondition()==Status.POISON)){
            return 2 * (0.4 * (double)att.getLevel() + (double)2.0F) * this.power / (double)150.0F;
        }
        return (0.4 * (double)att.getLevel() + (double)2.0F) * this.power / (double)150.0F;
    }
}
```

Рисунок 5 – Facade.java

```java
package PhMove;

import ru.ifmo.se.pokemon.PhysicalMove;
import ru.ifmo.se.pokemon.Type;

public class Pound extends PhysicalMove {  2 usages
    public Pound(){  1 usage
        super(Type.NORMAL,  v: 40,  v1: 100);
    }
    @Override
    protected String describe() {
        return "использует Pound";
    }
}
```

Рисунок 6 – Pound.java

```java
package PhMove;

import ru.ifmo.se.pokemon.Effect;
import ru.ifmo.se.pokemon.PhysicalMove;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class RockSlide extends PhysicalMove {  2 usages
    public RockSlide(){  1 usage
        super(Type.ROCK,  v: 75,  v1: 90);
    }
    @Override
    protected String describe() {
        return "использует Rock Slide";
    }
    //Оползень наносит урон и с вероятностью 30% заставляет цель вздрогнуть.
    @Override  no usages
    protected void applyOppEffects(Pokemon p){
        if (Math.random()<=0.3){
            Effect.flinch(p);
        }
    }
}
```

Рисунок 7 – RockSlide.java

```java
package SpMove;

import ru.ifmo.se.pokemon.Effect;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.SpecialMove;
import ru.ifmo.se.pokemon.Type;

public class DarkPulse extends SpecialMove {  2 usages
    public DarkPulse(){  1 usage
        super(Type.DARK,  v: 80,  v1: 100);
    }
    @Override
    protected String describe() {
        return "использует Dark Pulse";
    }
    //наносит урон и с вероятностью 20% заставляет цель вздрогнуть
    @Override  no usages
    protected void applyOppEffects(Pokemon p){ //
        if (Math.random()<=0.2){
            Effect.flinch(p);
        }
    }
}
```

Рисунок 8 – DarkPulse.java

```java
package SpMove;

import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.SpecialMove;
import ru.ifmo.se.pokemon.Stat;
import ru.ifmo.se.pokemon.Type;

public class FocusBlast extends SpecialMove {  2 usages
    public FocusBlast() {  1 usage
        super(Type.FIGHTING,  v: 120,  v1: 70);
    }

    @Override
    protected String describe() {
        return "использует Focus Blast";
    }
    //наносит урон и с вероятностью 10% снижает специальную защиту цели на одну ступень.
    @Override  no usages
    protected void applyOppEffects(Pokemon p) {
        if (Math.random() <= 0.1) {
            p.setMod(Stat.SPECIAL_ATTACK,  i: -1);

        }
    }
}
```

Рисунок 9 – FocusBlast.java

```java
package SpMove;

import ru.ifmo.se.pokemon.SpecialMove;
import ru.ifmo.se.pokemon.Type;

public class PowerGem extends SpecialMove {  2 usages
    public PowerGem(){  1 usage
        super(Type.ROCK,  v: 80,  v1: 100);
    }
    @Override
    protected String describe() {
        return "использует Power Gem";
    }
}
```

Рисунок 10 – PowerGem.java

```java
package SpMove;

import ru.ifmo.se.pokemon.*;

public class Thunderbolt extends SpecialMove {  2 usages
    public Thunderbolt(){  1 usage
        super(Type.ELECTRIC,  v: 90,  v1: 100);
    }
    @Override
    protected String describe() {
        return "использует Thunderbolt";
    }
    //Удар молнии наносит урон и с вероятностью 10% парализует цель.
    //Покемоны электрического типа не могут быть парализованы.
    @Override  no usages
    protected void applyOppEffects(Pokemon p){ //
        for (int i = 0; i < p.getTypes().length; i++) {
            Type x = p.getTypes()[i];
            if (x.equals(Type.ELECTRIC)) {
                return;
            }
        }
        if (Math.random()<=0.1){
            Effect.paralyze(p);
        }
    }
}
```

Рисунок 11 – Thunderbold.java

```java
package StMove;

import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Stat;
import ru.ifmo.se.pokemon.StatusMove;
import ru.ifmo.se.pokemon.Type;

public class Confide extends StatusMove {  2 usages
    public Confide(){  1 usage
        super(Type.NORMAL,  v: 0,  v1: 100);
    }
    @Override
    protected String describe() {
        return "использует Confide";
    }
    //снижает показатель «Специальной атаки» цели на одну ступень.
    @Override  no usages
    protected void applyOppEffects(Pokemon p) { //
        p.setMod(Stat.SPECIAL_ATTACK,  i: -1);

    }
}
```

Рисунок 12 – Confide.java

```java
package StMove;

import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Stat;
import ru.ifmo.se.pokemon.StatusMove;
import ru.ifmo.se.pokemon.Type;

public class DoubleTeam extends StatusMove {  4 usages
    public DoubleTeam(){  2 usages
        super(Type.NORMAL,  v: 0,  v1: 100);
    }
    @Override
    protected String describe() {
        return "использует Double Team";
    }
    //повышает показатель «Уклончивости» пользователя на одну ступень, из-за чего по нему сложнее попасть.
    @Override  no usages
    protected void applySelfEffects(Pokemon p) { //
        p.setMod(Stat.EVASION,  i: 1);

    }
}
```

Рисунок 13 – DoubleTeam.java

```java
package Pokemons;

import PhMove.Constrict;
import PhMove.Facade;
import SpMove.PowerGem;
import SpMove.Thunderbolt;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class Buzzwole extends Pokemon {  1 usage
    public Buzzwole (String name, int level){  1 usage
        super(name, level);
        setStats( v: 107,  v1: 139,  v2: 139,  v3: 53,  v4: 53,  v5: 79);
        setType(Type.BUG, Type.FIGHTING);
        setMove(new PowerGem(), new Facade(), new Constrict(), new Thunderbolt());
    }

}
```

Рисунок 14 – Buzzwole.java

```java
package Pokemons;

import PhMove.DoubleHit;
import PhMove.RockSlide;
import StMove.DoubleTeam;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class Corphish extends Pokemon {  2 usages  1 inheritor
    public Corphish (String name, int level){  2 usages
        super(name, level);
        setStats( v: 43,  v1: 80,  v2: 65,  v3: 50,  v4: 35,  v5: 35);
        setType(Type.WATER);
        setMove(new RockSlide(), new DoubleHit(), new DoubleTeam());
    }
}
```

Рисунок 15 – Corphish.java

```java
package Pokemons;

import SpMove.DarkPulse;
import ru.ifmo.se.pokemon.Type;

public class Crawdaunt extends Corphish{  1 usage
    public Crawdaunt(String name, int level) {  1 usage
        super(name, level);
        setStats( v: 63,  v1: 120,  v2: 85,  v3: 90,  v4: 55,  v5: 55);
        addType(Type.DARK);
        addMove(new DarkPulse());
    }

}
```

Рисунок 16 – Crawdaunt.java

```java
package Pokemons;


import PhMove.Pound;
import ru.ifmo.se.pokemon.Type;

public class Nuzleaf extends Seedot{  2 usages  1 inheritor
    public Nuzleaf(String name, int level) {  2 usages
        super(name, level);
        setStats( v: 70,  v1: 70,  v2: 40,  v3: 60,  v4: 40,  v5: 60);
        addType(Type.DARK);
        addMove(new Pound());
    }
}
```

Рисунок 17 – Nuzleaf.java

```java
package Pokemons;

import StMove.Confide;
import StMove.DoubleTeam;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class Seedot extends Pokemon {  2 usages  2 inheritors
    public Seedot(String name, int level){  2 usages
        super(name, level);
        setStats( v: 40,  v1: 40,  v2: 50,  v3: 30,  v4: 30,  v5: 30);
        setType(Type.GRASS);
        setMove(new Confide(), new DoubleTeam());
    }
}
```

Рисунок 18 – Seedot.java

```java
package Pokemons;

import SpMove.FocusBlast;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Stat;
import ru.ifmo.se.pokemon.Status;
import ru.ifmo.se.pokemon.Type;

public class Shiftry extends Nuzleaf{ 1 usage
    public Shiftry(String name, int level) { 1 usage
        super(name, level);
        setStats( v: 90, v1: 100, v2: 60, v3: 90, v4: 60, v5: 80);
        addMove(new FocusBlast());
    }
}
```

Рисунок 19 – Shiftry.java

```java
import Pokemons.*;
import ru.ifmo.se.pokemon.Battle;
import ru.ifmo.se.pokemon.Pokemon;

public class Main {
    public static void main(String[] args) {
        Battle b = new Battle();
        Pokemon p1 = new Buzzwole( name: "Енотик",  level: 50);
        Pokemon p2 = new Corphish( name: "Мопсик",  level: 12);
        Pokemon p3 = new Crawdaunt( name: "Паучача",  level: 3);
        b.addAlly(p1);
        b.addAlly(p2);
        b.addAlly(p3);
        Pokemon p4 = new Seedot( name: "Шкаф-купе",  level: 100);
        Pokemon p5 = new Nuzleaf( name: "Ананасик",  level: 8);
        Pokemon p6 = new Shiftry( name: "Шуроповертик",  level: 23);
        b.addFoe(p4);
        b.addFoe(p5);
        b.addFoe(p6);
        b.go();
    }
}
```

Рисунок 20 – Main.java

Результат работы программы

Buzzwole Енотик из команды черных вступает в бой!
Seedot Шкаф-купе из команды желтых вступает в бой!
Buzzwole Енотик использует Thunderbolt.
Критический удар!
Seedot Шкаф-купе теряет 18 здоровья.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Constrict.
Seedot Шкаф-купе теряет 7 здоровья.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Constrict.
Критический удар!
Seedot Шкаф-купе теряет 10 здоровья.
Seedot Шкаф-купе уменьшает скорость.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Seedot Шкаф-купе использует Double Team.
Seedot Шкаф-купе увеличивает уклоняемость.

Buzzwole Енотик использует Facade.
Seedot Шкаф-купе теряет 30 здоровья.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Thunderbolt.
Seedot Шкаф-купе восстанавливает 3 здоровья.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Thunderbolt.
Критический удар!
Seedot Шкаф-купе восстанавливает 2 здоровья.

Seedot Шкаф-купе использует Double Team.
Seedot Шкаф-купе увеличивает уклоняемость.

Buzzwole Енотик использует Constrict.
Seedot Шкаф-купе теряет 6 здоровья.
Seedot Шкаф-купе уменьшает скорость.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Thunderbolt.

Seedot Шкаф-купе использует Double Team.
Seedot Шкаф-купе увеличивает уклоняемость.

Buzzwole Енотик использует Power Gem.

Seedot Шкаф-купе использует Double Team.
Seedot Шкаф-купе увеличивает уклоняемость.

Buzzwole Енотик использует Thunderbolt.
Критический удар!

Seedot Шкаф-купе использует Double Team.
Seedot Шкаф-купе увеличивает уклоняемость.

Buzzwole Енотик использует Power Gem.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Power Gem.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Facade.
Критический удар!
Seedot Шкаф-купе теряет 46 здоровья.

Seedot Шкаф-купе использует Double Team.
Seedot Шкаф-купе увеличивает уклоняемость.

Buzzwole Енотик использует Power Gem.

Seedot Шкаф-купе использует Double Team.
Seedot Шкаф-купе увеличивает уклоняемость.

Buzzwole Енотик использует Thunderbolt.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Power Gem.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Power Gem.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Thunderbolt.
Seedot Шкаф-купе парализован

Buzzwole Енотик использует Power Gem.

Seedot Шкаф-купе использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Facade.
Критический удар!
Seedot Шкаф-купе теряет 109 здоровья.
Seedot Шкаф-купе теряет сознание.
Nuzleaf Ананасик из команды желтых вступает в бой!
Buzzwole Енотик использует Facade.
Nuzleaf Ананасик теряет 206 здоровья.
Nuzleaf Ананасик теряет сознание.
Shiftry Шуроповертик из команды желтых вступает в бой!
Buzzwole Енотик использует Power Gem.
Shiftry Шуроповертик восстанавливает 3 здоровья.

Shiftry Шуроповертик использует Confide.
Buzzwole Енотик уменьшает специальную атаку.

Buzzwole Енотик использует Thunderbolt.
Критический удар!
Shiftry Шуроповертик восстанавливает 5 здоровья.

Shiftry Шуроповертик использует Focus Blast.
Buzzwole Енотик теряет 5 здоровья.

Buzzwole Енотик использует Power Gem.
Shiftry Шуроповертик восстанавливает 4 здоровья.

Shiftry Шуроповертик использует Double Team.
Shiftry Шуроповертик увеличивает уклоняемость.

Buzzwole Енотик использует Facade.
Shiftry Шуроповертик теряет 79 здоровья.

Shiftry Шуроповертик использует Double Team.
Shiftry Шуроповертик увеличивает уклоняемость.

Buzzwole Енотик использует Constrict.
Shiftry Шуроповертик теряет 13 здоровья.
Shiftry Шуроповертик теряет сознание.
В команде желтых не осталось покемонов.
Команда черных побеждает в этом бою!

Заключение

Во время выполнения лабораторной работы я изучила и применила основные концепции ООП, такие как наследование, инкапсуляция, полиморфизм и абстракция, и применила их на практике. Научилась работать с готовой документацией проекта, реализовала свои классы покемонов и их атак по заданию.