образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА № 2

по дисциплине

‘ПРОГРАММИРОВАНИЕ’

Вариант № 31180045

Выполнил:
Студент группы P3118
Шипунов Илья
Михайлович
Преподаватель:
Письмак Алексей
Евгеньевич

Санкт-Петербург
2021

Задание:

На основе базового класса `Pokemon` написать свои классы для заданных видов покемонов. Каждый вид покемона должен иметь один или два типа и стандартные базовые характеристики:

- очки здоровья (HP)
- атака (attack)
- защита (defense)
- специальная атака (special attack)
- специальная защита (special defense)
- скорость (speed)

Классы покемонов должны наследоваться в соответствии с цепочкой эволюции покемонов. На основе базовых классов `PhysicalMove`, `SpecialMove` и `StatusMove` реализовать свои классы для заданных видов атак.

Атака должна иметь стандартные тип, силу (power) и точность (accuracy). Должны быть реализованы стандартные эффекты атаки. Назначить каждому виду покемонов атаки в соответствии с вариантом. Уровень покемона выбирается минимально необходимым для всех реализованных атак.

Используя класс симуляции боя `Battle`, создать 2 команды покемонов (каждый покемон должен иметь имя) и запустить бой.

Базовые классы и симулятор сражения находятся в jar-архиве (обновлен 9.10.2018, исправлен баг с добавлением атак и кодировкой). Документация в формате javadoc - здесь.

Информацию о покемонах, цепочках эволюции и атаках можно найти на сайтах http://poke-universe.ru, http://pokemondb.net, http://veekun.com/dex/pokemon

Комментарии

Цель работы: на простом примере разобраться с основными концепциями ООП и научиться использовать их в программах.

Что надо сделать (краткое описание)

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.

Ваши покемоны:

Seviper

Атаки:
- Thunder
- Harden
- Scary Face
- Poison Jab

Noibat

Атаки:
- Bulldoze
- Confide
- Charm

Noivern

Атаки:
- Bulldoze
- Confide
- Charm
- Headbutt

Chimchar

Атаки:
- Blizzard
- Rest

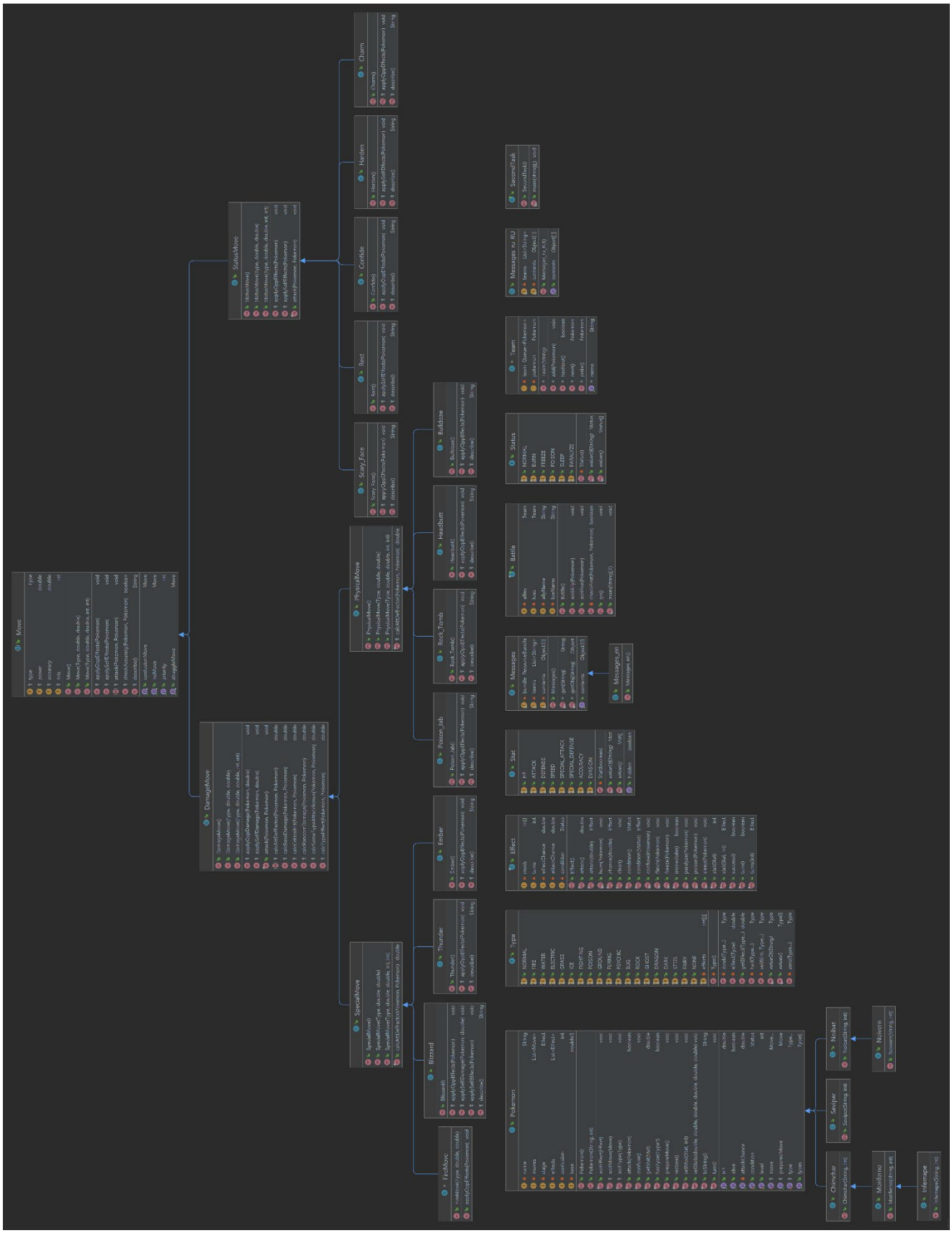
Monferno

Атаки:
- Blizzard
- Rest
- Rock Tomb

Infernape

Атаки:
- Blizzard
- Rest
- Rock Tomb
- Ember

UML-диаграмма классов



Исходный код:

```java
import PokemonList.*;
import ru.ifmo.se.pokemon.Battle;

public class SecondTask {
    public static void main (String [] arg) {
        Battle b = new Battle();

        Seviper p1 = new Seviper("Змій",1);
        Infernape p2 = new Infernape("Инферномат",1);
        Monferno p3 = new Monferno("Оберно", 1);
        Noibat p4 = new Noibat("Мышь", 1);
        Noivern p5 = new Noivern("Мырна", 1);
        Chimchar p6 = new Chimchar("ЧимСтринг", 1);

        b.addAlly(p1);
        b.addAlly(p4);
        b.addAlly(p5);

        b.addFoe(p2);
        b.addFoe(p3);
        b.addFoe(p6);

        b.go();
    }
}
```

Покемоны:

```java
package PokemonList;

import MoveList.Blizzard;
import MoveList.Rest;
import ru.ifmo.se.pokemon.*;


public class Chimchar extends Pokemon {
    public Chimchar(String name, int lvl) {
        super(name,lvl);
        setStats( v: 44.0, v1: 58.0, v2: 44.0, v3: 58.0, v4: 44.0, v5: 61.0);
        addType(Type.FIRE);
        setMove(new Blizzard(), new Rest());
    }
}
```

```java
package PokemonList;

import MoveList.Ember;
import ru.ifmo.se.pokemon.*;

public class Infernape extends Monferno {
    public Infernape(String name, int lvl) {
        super(name, lvl);
        setStats( v: 76.0,  v1: 104.0,  v2: 71.0,  v3: 104.0,  v4: 71.0,  v5: 108.0);
        setType(Type.FIRE, Type.FIGHTING);
        addMove(new Ember());
    }
}
```

```java
package PokemonList;

import MoveList.Rock_Tomb;
import ru.ifmo.se.pokemon.*;

public class Monferno extends Chimchar {
    public Monferno(String name, int lvl) {
        super(name, lvl);
        setStats( v: 64.0,  v1: 78.0,  v2: 52.0,  v3: 78.0,  v4: 52.0,  v5: 81.0);
        setType(Type.FIRE, Type.FIGHTING);
        addMove(new Rock_Tomb());
    }
}
```

```java
package PokemonList;

import MoveList.Bulldoze;
import MoveList.Charm;
import MoveList.Confide;
import ru.ifmo.se.pokemon.*;

public class Noibat extends Pokemon {
    public Noibat(String name, int lvl) {
        super(name, lvl);
        setStats( v: 40.0,  v1: 30.0,  v2: 35.0,  v3: 45.0,  v4: 40.0,  v5: 55.0);
        setType(Type.FLYING, Type.DRAGON);
        setMove(new Bulldoze(), new Confide(), new Charm());
    }
}
```

```java
package PokemonList;

import MoveList.Headbutt;
import ru.ifmo.se.pokemon.*;

public class Noivern extends Noibat {
    public Noivern(String name, int lvl) {
        super(name, lvl);
        setStats( v: 85.0, v1: 70.0, v2: 80.0, v3: 97.0, v4: 80.0, v5: 123.0);
        setType(Type.FLYING, Type.DRAGON);
        addMove(new Headbutt());
    }
}
```

```java
package PokemonList;

import MoveList.Harden;
import MoveList.Poison_Jab;
import MoveList.Scary_Face;
import MoveList.Thunder;
import ru.ifmo.se.pokemon.*;

public class Seviper extends Pokemon {
    public Seviper(String name, int lvl) {
        super(name,lvl);
        setStats( v: 73.0, v1: 100.0, v2: 60.0, v3: 100.0, v4: 60.0, v5: 65.0);
        addType(Type.POISON);
        setMove(new Thunder(), new Poison_Jab(), new Scary_Face(), new Harden());
    }
}
```

Атаки:

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Blizzard extends SpecialMove {
    public  Blizzard () {
        type = Type.ICE;
        power = 110.0;
        accuracy = 70.0;
    }

    @Override
    protected void applyOppEffects(Pokemon var1) {
        if (!var1.hasType(Type.ICE)) {
            Effect e = (new Effect()).condition(Status.FREEZE).attack(0.0D).turns(-1).chance(0.1);
            var1.setCondition(e);
        }
    }

    protected void applySelfEffects(Pokemon var1) {
        if (!var1.hasType(Type.ICE)) {
            Effect e = (new Effect()).condition(Status.FREEZE).attack(0.0D).turns(-1).chance(0.1);
            var1.setCondition(e);
        }
    }

    protected void applySelfDamage(Pokemon var1, double var2) { var1.setMod(Stat.HP, (int) Math.round(var2)); }

    protected String describe() { return ("использует Blizzard"); }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Bulldoze extends PhysicalMove {
    public Bulldoze () {
        type = Type.GROUND;
        power = 60;
        accuracy = 100;
    }

    @Override
    protected void applyOppEffects(Pokemon var1) {
        Effect e = new Effect().stat(Stat.SPEED, -1);
        var1.setCondition(e);
    }

    protected String describe() { return "использует Bulldoze"; }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Charm extends StatusMove {
    public Charm () {
        type = Type.FAIRY;
        accuracy = 100;
    }

    @Override
    protected void applyOppEffects(Pokemon var1) {
        Effect e = new Effect().stat(Stat.ATTACK, -2);
        var1.setCondition(e);
    }

    protected String describe() { return "использует Charm"; }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Confide extends StatusMove {
    public Confide () { type = Type.NORMAL; }

    @Override
    protected void applyOppEffects(Pokemon var1) {
        Effect e = new Effect().stat(Stat.SPECIAL_ATTACK, -1);
        var1.setCondition(e);
    }

    protected String describe() { return "использует Confide"; }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Ember extends SpecialMove {
    public  Ember () {
        type = Type.FIRE;
        power = 40.0;
        accuracy = 100.0;
    }

    @Override
    protected void applyOppEffects(Pokemon var1) {
        if (!var1.hasType(Type.FIRE)) {
            Effect e = (new Effect()).condition(Status.BURN).turns(-1).stat(Stat.ATTACK, -2).stat(Stat.HP,
                    (int)var1.getStat(Stat.HP) / 16).chance(0.1);
            var1.setCondition(e);
        }
    }

    protected String describe() { return ("использует Ember"); }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Harden extends StatusMove {
    public Harden () { type = Type.NORMAL; }

    @Override
    protected void applySelfEffects(Pokemon var1) {
        Effect e = new Effect().stat(Stat.DEFENSE, 1);
        var1.setCondition(e);
    }

    protected String describe() { return "использует Harden"; }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Headbutt extends PhysicalMove {
    public Headbutt () {
        type = Type.NORMAL;
        power = 60;
        accuracy = 100;
    }

    @Override
    protected void applyOppEffects(Pokemon var1) {
        Effect e = (new Effect()).attack(0.0D).turns((int)(Math.random() * 4.0D + 1.0D)).chance(0.3);
        var1.setCondition(e);
    }

    protected String describe() { return "использует Headbutt"; }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;


public class Poison_Jab extends PhysicalMove {
    public Poison_Jab () {
        type = Type.POISON;
        power = 80;
        accuracy = 100;
    }

    @Override
    protected void applyOppEffects(Pokemon var1) {
        if (!var1.hasType(Type.POISON) && !var1.hasType(Type.STEEL)) {
            Effect e = (new Effect()).condition(Status.POISON).turns(-1).stat(Stat.HP,
                    (int)var1.getStat(Stat.HP) / 8).chance(0.3);
            var1.setCondition(e);
        }
    }

    protected String describe() { return "использует Poison Jab"; }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Rest extends StatusMove {
    public  Rest () { type = Type.PSYCHIC; }

    @Override
    protected void applySelfEffects(Pokemon var1) {
        var1.restore();
        Effect e = (new Effect()).condition(Status.SLEEP).attack(0.0D).turns(2);
        var1.setCondition(e);
    }

    protected String describe() { return ("использует Rest"); }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Rock_Tomb extends PhysicalMove {
    public  Rock_Tomb () {
        type = Type.ROCK;
        power = 60.0;
        accuracy = 95.0;
    }

    @Override
    protected void applyOppEffects(Pokemon var1) {
        Effect e = new Effect().stat(Stat.SPEED,  1);
        var1.setCondition(e);
    }

    protected String describe() {
        return ("использует Rock Tomb");
    }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Scary_Face extends StatusMove {

    public Scary_Face() {
        type = Type.NORMAL;
        accuracy = 100;
    }

    @Override
    protected void applyOppEffects(Pokemon var1) {
       Effect e = new Effect().stat(Stat.SPEED, -2);
       var1.setCondition(e);
    }

    protected String describe() { return "использует Scary Face"; }
}
```

```java
package MoveList;

import ru.ifmo.se.pokemon.*;

public class Thunder extends SpecialMove {
    public  Thunder () {
        type = Type.ELECTRIC;
        power = 110.0;
        accuracy = 70.0;
    }

    @Override
    protected void applyOppEffects(Pokemon var1) {
        if (!var1.hasType(Type.ELECTRIC)) {
            Effect e = (new Effect()).condition(Status.PARALYZE).attack(0.75D).turns(-1).stat(Stat.SPEED,
                    -2).chance(0.3);
            var1.setCondition(e);
        }
    }

    protected String describe() { return ("использует Thunder"); }
}
```

Результат работы:

```
Seviper Змій из команды белых вступает в бой!
Infernape Инферномат из команды полосатых вступает в бой!
Infernape Инферномат использует Blizzard.
Seviper Змій теряет 6 здоровья.
Infernape Инферномат теряет 6 здоровья.

Seviper Змій использует Scary Face.

Infernape Инферномат использует Blizzard.
Seviper Змій теряет 5 здоровья.
Infernape Инферномат теряет 5 здоровья.

Seviper Змій использует Harden.

Infernape Инферномат использует Blizzard.
Seviper Змій теряет 6 здоровья.
Infernape Инферномат теряет 6 здоровья.
Infernape Инферномат замерзает
Оба покемона теряют сознание.
Noibat Мышь из команды белых вступает в бой!
Monferno Оберно из команды полосатых вступает в бой!
Monferno Оберно использует Rest.
Monferno Оберно засыпает

Noibat Мышь использует Bulldoze.
Monferno Оберно теряет 7 здоровья.
Monferno Оберно

Monferno Оберно использует Rest.
Monferno Оберно засыпает

Noibat Мышь использует Confide.
Monferno Оберно

Monferno Оберно использует Blizzard.
Noibat Мышь теряет 35 здоровья.
Monferno Оберно теряет 35 здоровья.
Оба покемона теряют сознание.
Noivern Мырна из команды белых вступает в бой!
```

```
Chimchar ЧимСтринг из команды полосатых вступает в бой!
Noivern Мырна использует Headbutt.
Chimchar ЧимСтринг теряет 6 здоровья.

Chimchar ЧимСтринг использует Rest.
Chimchar ЧимСтринг засыпает

Noivern Мырна использует Bulldoze.
Chimchar ЧимСтринг теряет 8 здоровья.
Chimchar ЧимСтринг


Noivern Мырна использует Bulldoze.
Chimchar ЧимСтринг теряет 9 здоровья.
Chimchar ЧимСтринг теряет сознание.
В команде полосатых не осталось покемонов.
Команда белых побеждает в этом бою!

Process finished with exit code 0
```

Вывод:

Во время выполнения лабораторной работы были изучены базовые принципы наследования в языке программирования Java, конструкторы классов, принципы ООП (полиморфизм, инкапсуляция, наследование).