Университет ИТМО

Факультет ФПИ и КТ

Лабораторная работа №2

По программированию

Вариант 76554

Выполнил: Умарова А. М.

Группа: P3118

Преподаватель: Ермаков М.К.

Санкт-Петербург

2024

Ход работы

1. 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. Все разработанные классы, не имеющие наследников, должны быть реализованы таким образом, чтобы от них нельзя было наследоваться.

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.

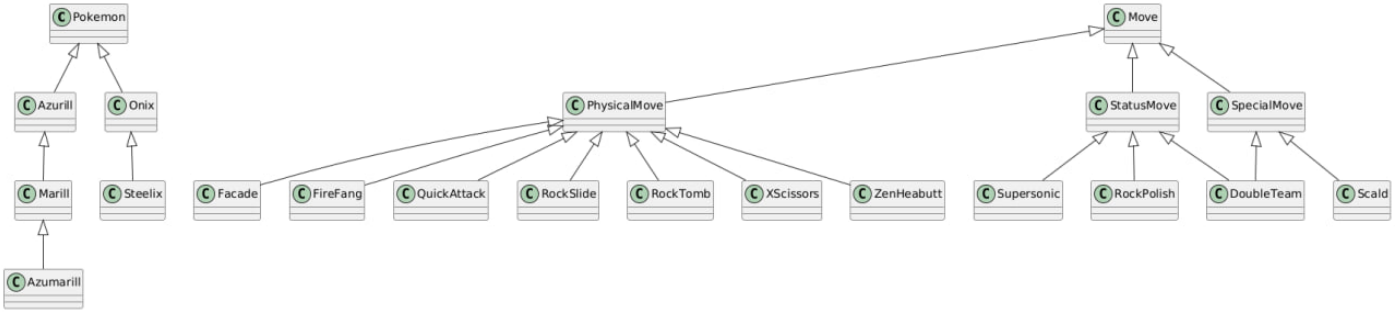
Используя класс симуляции боя `Battle`, создать 2 команды покемонов (каждый покемон должен иметь имя) и запустить бой.

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

Ваши покемоны:

2. Диаграмма классов реализованной объектной модели.



3. Код программы:

```
package pokemons;

import moves.QuickAttack;
import moves.RockSlide;
import moves.XScissors;
import moves.ZenHeadbutt;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class Azurill extends Pokemon {
    public Azurill(String name, int lvl) {
        super(name, lvl);
        super.setStats(65, 130, 60, 75, 75, 60);
        setType(Type.DARK);
        setMove(new QuickAttack(), new ZenHeadbutt(), new XScissors(), new RockSlide());

    }
}
package pokemons;

import moves.BubbleBeam;
import moves.Scald;
import moves.Supersonic;
import ru.ifmo.se.pokemon.Type;

public class Marill extends Azurill {
    public Marill(String name, int lvl) {
        super(name, lvl);
        super.setStats(70, 20, 50, 20, 50, 40);
        setType(Type.WATER, Type.FAIRY);
        setMove(new BubbleBeam(), new Scald(), new Supersonic());

    }
}
package pokemons;

import moves.*;
import ru.ifmo.se.pokemon.Type;

public final class Azumarill extends Marill {
    public Azumarill(String name, int lvl) {
        super(name, lvl);
        super.setStats(100, 50, 80, 60, 80, 50);
        setType(Type.WATER, Type.FAIRY);
        setMove(new BubbleBeam(), new Scald(), new Supersonic(), new DoubleTeam());
```

```
    }
}
package pokemons;

import moves.*;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class Onix extends Pokemon {
    public Onix(String name, int lvl) {
        super(name, lvl);
        super.setStats(35, 45, 160, 30, 45, 70);
        setType(Type.ROCK, Type.GROUND);
        setMove(new RockTomb(), new Facade(), new RockPolish());
    }
    }

package pokemons;

import moves.*;
import ru.ifmo.se.pokemon.Type;

public final class Steelix extends Onix {
    public Steelix(String name, int lvl) {
        super(name, lvl);
        super.setStats(75, 85, 200, 55, 65, 30);
        setType(Type.STEEL, Type.GROUND);
        setMove(new RockTomb(), new Facade(), new RockPolish(), new FireFang());

    }
}


package moves;

import ru.ifmo.se.pokemon.*;

public class Facade extends PhysicalMove {
    public Facade() {
        super(Type.NORMAL, 70, 100);
    }

    @Override
    protected void applySelfEffects(Pokemon p) {
        Status cond = p.getCondition();
        if (cond == Status.BURN || cond == Status.POISON || cond == Status.PARALYZE) {
            double curAt = p.getStat(Stat.ATTACK);
            Effect ef = new Effect().stat(Stat.ATTACK, (int) curAt * 2);
            p.addEffect(ef);
        }

    }
    @Override
    protected String describe(){
        return "is using Facade";
    }
}


package moves;

import ru.ifmo.se.pokemon.Effect;
import ru.ifmo.se.pokemon.PhysicalMove;
```

```java
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class FireFang extends PhysicalMove {
    boolean flinch = false;
    public FireFang(){
        super(Type.FIRE, 65, 95);
    }
    @Override
    protected void applyOppEffects(Pokemon p){
        if (Math.random() < 0.1){
            Effect.burn(p);
        }
        if (Math.random() < 0.1){
            Effect.flinch(p);
            flinch = true;
        }
    }
    @Override
    protected String describe(){
        return "is using Fire Fang";
    }
}
```

```java
package moves;

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

public class QuickAttack extends PhysicalMove {
    public QuickAttack() {
        super(Type.NORMAL, 100, 40, 1, 1);}

    @Override protected String describe(){
        return "is using Quick Attack";}
}
```

```java
package moves;

import ru.ifmo.se.pokemon.*;

public class RockSlide extends PhysicalMove {
    boolean flinch = false;
    public RockSlide(){
        super(Type.ROCK, 75, 90);
    }
    @Override
    protected void applyOppEffects(Pokemon pokemon) {
        if (Math.random() <= 0.3){
            Effect.flinch(pokemon);
            flinch = true;
        }
    }

    @Override
    protected String describe() {
        return "is using Rock Slide";
    }
}
```

```java
package moves;
```

```java
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 RockTomb extends PhysicalMove {
    public RockTomb(){
        super(Type.ROCK, 60, 95);
    }
    @Override
    protected void applyOppEffects(Pokemon p){
        p.setMod(Stat.SPEED, -1);
    }
    @Override
    protected String describe(){
        return "is using Rock Tomb";
    }
}
```

```java
package moves;

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

public class XScissors extends PhysicalMove {
    public XScissors() {
        super(Type.BUG, 100, 80);
    }

    @Override
    protected String describe(){
        return "is using Horn Leech";
    }
}
```

```java
package moves;

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 ZenHeadbutt extends PhysicalMove {
    boolean flinch = false;
    public ZenHeadbutt(){
        super(Type.PSYCHIC, 80, 90);}

    @Override
    protected void applyOppEffects(Pokemon pokemon) {
        if (Math.random() <= 0.2){
            Effect.flinch(pokemon);
            flinch = true;
        }
    }
    @Override
    protected String describe(){
        return "is using Zen Headbutt";}
}
```

```java
package moves;

import ru.ifmo.se.pokemon.*;
```

```java
public class BubbleBeam extends SpecialMove {
    public BubbleBeam(){
        super(Type.WATER, 65, 100);
    }
    @Override
    protected void applyOppEffects(Pokemon p){
        if (Math.random() < 0.1){
            p.setMod(Stat.SPEED, -1);
        }
    }
     @Override
    protected String describe(){
        return "is using Bubble Beam";
     }
}

package moves;

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 Scald extends SpecialMove {
    public Scald() {
        super(Type.WATER, 80, 100);
    }

    @Override
    protected void applyOppEffects(Pokemon p) {
        if (Math.random() < 0.3) {
            Effect.burn(p);
        }
    }

    @Override
    protected String describe() {
        return "is using Scald";
    }
}

package moves;

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

public class Supersonic extends StatusMove {
    public Supersonic(){
        super(Type.NORMAL, 0, 55);
    }
    @Override
    protected void applyOppEffects(Pokemon p){
        Effect.confuse(p);}
    @Override
    protected String describe(){
        return "is using Sypersonic";
    }
}

package moves;
```

```java
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 RockPolish extends StatusMove {
    public RockPolish(){
        super(Type.ROCK, 0, 0);
    }
    @Override
    protected void applySelfEffects(Pokemon p){
        p.setMod(Stat.SPEED, 2);
    }
    @Override
    protected String describe(){
        return "is using Rock Polish";
    }
}
```

```java
package moves;

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 {
    public DoubleTeam(){
    super(Type.NORMAL, 0, 0);}

    @Override
    protected void applySelfEffects(Pokemon p){
        p.setMod(Stat.EVASION, 1);
    }
    @Override
    protected String describe(){
        return "is using Double Team";
    }
}
```

```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 Pokemon("Чужой", 1);
        Pokemon p2 = new Pokemon("Хищник", 1);
        Absol a1 = new Absol("Absol", 1);
        Azumarill a2 = new Azumarill("Azumarill", 1);
        Marill a3 = new Marill("Marill", 1);
        Steelix a5 = new Steelix("Steelix", 1);
        Onix a4 = new Onix("Onix", 1);
        Azurill a6 = new Azurill("Azurill", 1);

        b.addAlly(a1);
        b.addFoe(a2);
        b.addAlly(a3);
        b.addFoe(a4);
        b.addAlly(a5);
```

```
        b.addFoe(a6);

        b.go();
    }
}
```

4. Вывод программы:

Absol Absol из команды белых вступает в бой!
Azumarill Azumarill из команды полосатых вступает в бой!
Absol Absol is using Zen Headbutt.
Azumarill Azumarill теряет 7 здоровья.

Azumarill Azumarill is using Scald.
Absol Absol теряет 6 здоровья.

Absol Absol is using Quick Attack.
Azumarill Azumarill теряет 5 здоровья.

Azumarill Azumarill is using Bubble Beam.
Absol Absol теряет 6 здоровья.

Absol Absol is using Horn Leech.
Azumarill Azumarill теряет 3 здоровья.
Azumarill Azumarill теряет сознание.
Onix Onix из команды полосатых вступает в бой!
Onix Onix is using Facade.
Absol Absol теряет 6 здоровья.
Absol Absol теряет сознание.
Marill Marill из команды белых вступает в бой!
Onix Onix is using Facade.
Marill Marill теряет 4 здоровья.

Marill Marill is using Bubble Beam.
Onix Onix теряет 23 здоровья.
Onix Onix теряет сознание.
Azurill Azurill из команды полосатых вступает в бой!
Azurill Azurill is using Rock Slide.
Marill Marill теряет 6 здоровья.

Marill Marill is using Bubble Beam.
Azurill Azurill теряет 7 здоровья.

Azurill Azurill is using Quick Attack.
Marill Marill теряет 7 здоровья.
Marill Marill теряет сознание.
Steelix Steelix из команды белых вступает в бой!
Azurill Azurill is using Zen Headbutt.
Steelix Steelix теряет 3 здоровья.

Steelix Steelix промахивается

Azurill Azurill is using Horn Leech.
Steelix Steelix теряет 3 здоровья.

Azurill Azurill is using Horn Leech.
Steelix Steelix теряет 3 здоровья.

Azurill Azurill is using Rock Slide.
Steelix Steelix теряет 1 здоровья.

Steelix Steelix is using Fire Fang.
Azurill Azurill теряет 4 здоровья.

Azurill Azurill is using Rock Slide.
Steelix Steelix теряет 1 здоровья.

Steelix Steelix is using Rock Tomb.
Azurill Azurill теряет 4 здоровья.
Azurill Azurill уменьшает скорость.
Azurill Azurill теряет сознание.
В команде полосатых не осталось покемонов.
Команда белых побеждает в этом бою!

5. **Вывод:** во время работы над программой изучила создание и работу с классами в java, а также принципы ООП, такие как наследование и полиморфизм, научилась собирать jar-архив для множества классов, а также работала с документацией.