Факультет программной инженерии и компьютерной техники

Программирование

Лабораторная работа №2

Вариант 1033423

Преподаватель: Сорокин Роман Борисович

Выполнил: Берелехис Светлана Михайловна

Р3112

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.

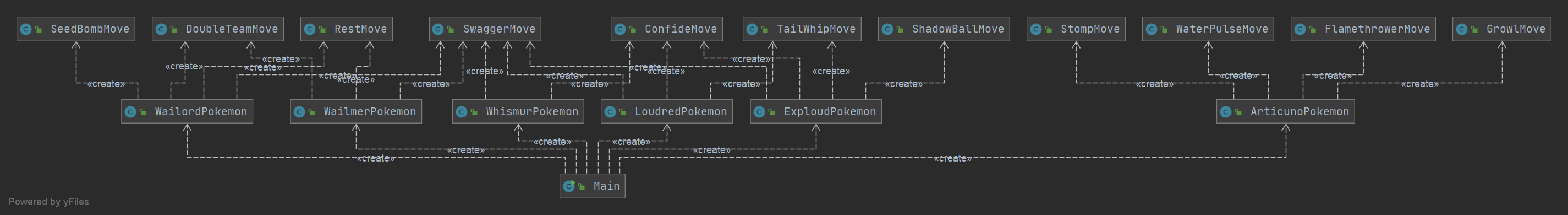
Базовые классы и симулятор сражения находятся в [jar-архиве](https://se.ifmo.ru/documents/10180/660917/Pokemon.jar/a7ce60af-6ee6-47d0-a95e-e5ed9a697bd2) (обновлен 9.10.2018, исправлен баг с добавлением атак и кодировкой). Документация в формате javadoc - [здесь](https://se.ifmo.ru/~tony/doc/).

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

Покемоны:



Диаграмма классов:



Исходный код:

Main.java

import ru.ifmo.se.pokemon.Battle;  
  
public class Main {  
 public static void main(String[] args){  
 Battle field = new Battle();  
 field.addAlly(new ArticunoPokemon("Articuno", 1));  
 field.addAlly(new ExploudPokemon("Exploud", 2));  
 field.addAlly(new LoudredPokemon("Loudred", 3));  
 field.addFoe(new WailmerPokemon("Waimer", 1));  
 field.addFoe(new WailordPokemon("Wailord", 2));  
 field.addFoe(new WhismurPokemon("Whismur", 3));  
 field.go();  
 }  
}

ArticunoPokemon.java

import ru.ifmo.se.pokemon.\*;  
  
  
public class ArticunoPokemon extends Pokemon {  
 public ArticunoPokemon(String name, int level){  
 super(name, 1);  
 setStats(90, 85, 100, 95, 125, 85);  
 setType(Type.*ICE*);  
 setMove(new WaterPulseMove(), new FlamethrowerMove(), new GrowlMove(), new StompMove());  
 }  
}

ExploudPokemon.java

import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
public class ExploudPokemon extends Pokemon {  
 public ExploudPokemon(String name, int level){  
 super(name, level);  
 setStats(104, 91, 63, 91, 73, 68);  
 setType(Type.*NORMAL*);  
 setMove(new SwaggerMove(), new ConfideMove(), new TailWhipMove(), new ShadowBallMove());  
 }  
}

LoudredPokemon.java

import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
public class LoudredPokemon extends Pokemon {  
 public LoudredPokemon(String name, int level){  
 super(name, level);  
 setStats(84, 71, 43, 71, 43, 48);  
 setType(Type.*NORMAL*);  
 setMove(new SwaggerMove(), new ConfideMove(), new TailWhipMove());  
 }  
}

WailmerPokemon.java

import ru.ifmo.se.pokemon.Type;  
  
public class WailmerPokemon extends Pokemon {  
 public WailmerPokemon(String name, int level){  
 super(name, level);  
 setStats(130, 70, 35, 70, 35, 60);  
 setType(Type.*WATER*);  
 setMove(new SwaggerMove(), new DoubleTeamMove(), new RestMove());  
 }  
}

WailordPokemon.java

import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
public class WailordPokemon extends Pokemon {  
 public WailordPokemon(String name, int level){  
 super(name, level);  
 setStats(170, 90, 45, 90, 45, 60);  
 setType(Type.*WATER*);  
 setMove(new SwaggerMove(), new DoubleTeamMove(), new RestMove(), new SeedBombMove());  
 }  
}

WhismurPokemon.java

import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
public class WhismurPokemon extends Pokemon {  
 public WhismurPokemon(String name, int level){  
 super(name, level);  
 setStats(64, 51, 23, 51, 23, 28);  
 setType(Type.*NORMAL*);  
 setMove(new SwaggerMove(), new ConfideMove());  
 }  
}

ConfideMove.java

import ru.ifmo.se.pokemon.\*;  
  
public class ConfideMove extends StatusMove{  
 public ConfideMove(){  
 super(Type.*NORMAL*, 0, 0);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p){  
 p.setMod(Stat.*SPECIAL\_ATTACK* , -1);  
 }  
 @Override  
 protected String describe(){  
 return "Снижает специальную атаку цели на один уровень";  
 }  
}

DoubleTeamMove.java

import ru.ifmo.se.pokemon.\*;  
  
  
public class DoubleTeamMove extends StatusMove {  
 public DoubleTeamMove(){  
 super(Type.*NORMAL*, 0, 0);  
 }  
 @Override  
 protected void applySelfEffects (Pokemon p){  
 p.setMod(Stat.*EVASION* , 1);  
 }  
 @Override  
 protected String describe(){  
 return "Повышает своё уклонение на одну ступень.";  
 }  
  
}

FlamethrowerMove.java

import ru.ifmo.se.pokemon.\*;  
  
import java.util.Random;  
  
public class FlamethrowerMove extends PhysicalMove {  
 public FlamethrowerMove(){  
 super(Type.*FIRE*, 95, 100);  
 }  
 protected void applyOppEffects(Pokemon p){  
 if (new Random().nextInt(100)<10) Effect.*burn*(p);  
 }  
 @Override  
 protected String describe(){  
 return "Имеет 10% шанс ожечь цель";  
 }  
}

GrowlMove.java

import ru.ifmo.se.pokemon.Stat;  
import ru.ifmo.se.pokemon.Type;  
  
public class GrowlMove extends PhysicalMove{  
 public GrowlMove(){  
 super(Type.*NORMAL*, 0, 100);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p){  
 p.setMod(Stat.*ATTACK*, -1);  
 }  
 @Override  
 protected String describe(){  
 return "Понижает атаку цели на одну ступень";  
 }  
}

RestMove.java

import ru.ifmo.se.pokemon.\*;  
  
public class RestMove extends StatusMove {  
 public RestMove(){  
 super(Type.*PSYCHIC*, 0, 0);  
 }  
  
 @Override  
 protected void applySelfEffects(Pokemon pokemon) {  
 Effect.*sleep*(pokemon);  
 pokemon.setMod(Stat.*HP* , (int) (pokemon.getStat(Stat.*HP*) - pokemon.getHP()));  
 }  
 @Override  
 protected String describe(){  
 return "восстонавливает здоровье и засыпает";  
 }  
}

SeedBombMove.java

import ru.ifmo.se.pokemon.PhysicalMove;  
import ru.ifmo.se.pokemon.Type;  
  
public class SeedBombMove extends PhysicalMove {  
 public SeedBombMove(){  
 super(Type.*GRASS*, 80, 100);  
 }  
 @Override  
 protected String describe(){  
 return "никаких дополнительных эфектов";  
 }  
}

ShadowBallMove.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;  
  
import java.util.Random;  
  
public class ShadowBallMove extends StatusMove {  
 public ShadowBallMove(){  
 super(Type.*NORMAL*, 80, 100);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p){  
 if (new Random().nextInt(100)<20)p.setMod(Stat.*SPECIAL\_DEFENSE*, -1);  
 }  
 @Override  
 protected String describe(){  
 return "Имеет 20% вероятность понизить Спец. Защиту цели на одну ступень.";  
 }  
}

StompMove.java

import ru.ifmo.se.pokemon.Effect;  
import ru.ifmo.se.pokemon.PhysicalMove;  
import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
import java.util.Random;  
  
public class StompMove extends PhysicalMove {  
 public StompMove(){  
 super(Type.*NORMAL*, 60, 100);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p){  
 if (new Random().nextInt(100)<30) Effect.*flinch*(p);  
 }  
 @Override  
 protected String describe(){  
 return "имеет 30% вероятность заставить цель дрогнуть";  
 }  
}

SwaggerMove.java

import ru.ifmo.se.pokemon.\*;  
  
public class SwaggerMove extends StatusMove {  
 public SwaggerMove(){  
 super(Type.*NORMAL*, 0, 85);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p){  
 p.setMod(Stat.*ATTACK*, 2);  
 Effect.*confuse*(p);  
 }  
 @Override  
 protected String describe(){  
 return "Повышает Атаку цели на две ступени и сбивает её с толку";  
 }

TailWhipMove.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 TailWhipMove extends StatusMove {  
 public TailWhipMove(){  
 super(Type.*NORMAL*, 0, 100);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p){  
 p.setMod(Stat.*DEFENSE*, -1);  
 }  
 @Override  
 protected String describe(){  
 return "Снижает защиту цели на один уровень";  
 }  
}

WaterPulseMove.java

import ru.ifmo.se.pokemon.Effect;  
import ru.ifmo.se.pokemon.PhysicalMove;  
import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
import java.util.Random;  
  
public class WaterPulseMove extends PhysicalMove {  
 public WaterPulseMove(){  
 super(Type.*WATER*, 60, 100);  
 }  
 protected void applyOppEffects(Pokemon p){  
 if (new Random().nextInt(100)<30) Effect.*confuse*(p);  
 }  
 @Override  
 protected String describe(){  
 return "Имеет 30% шанс сбить цель с толку";  
 }  
}

Результат работы:

ArticunoPokemon Articuno из команды фиолетовых вступает в бой!

WailmerPokemon Waimer из команды черных вступает в бой!

ArticunoPokemon Articuno имеет 30% вероятность заставить цель дрогнуть.

WailmerPokemon Waimer теряет 6 здоровья.

WailmerPokemon Waimer промахивается

ArticunoPokemon Articuno имеет 30% вероятность заставить цель дрогнуть.

WailmerPokemon Waimer теряет 4 здоровья.

WailmerPokemon Waimer промахивается

ArticunoPokemon Articuno имеет 30% вероятность заставить цель дрогнуть.

WailmerPokemon Waimer теряет 5 здоровья.

WailmerPokemon Waimer теряет сознание.

WailordPokemon Wailord из команды черных вступает в бой!

WailordPokemon Wailord промахивается

ArticunoPokemon Articuno Понижает атаку цели на одну ступень.

WailordPokemon Wailord теряет 3 здоровья.

WailordPokemon Wailord уменьшает атаку.

WailordPokemon Wailord промахивается

ArticunoPokemon Articuno Имеет 10% шанс ожечь цель.

WailordPokemon Wailord теряет 3 здоровья.

WailordPokemon Wailord промахивается

ArticunoPokemon Articuno Имеет 10% шанс ожечь цель.

WailordPokemon Wailord теряет 3 здоровья.

WailordPokemon Wailord промахивается

ArticunoPokemon Articuno имеет 30% вероятность заставить цель дрогнуть.

WailordPokemon Wailord теряет 3 здоровья.

WailordPokemon Wailord промахивается

ArticunoPokemon Articuno имеет 30% вероятность заставить цель дрогнуть.

WailordPokemon Wailord теряет 4 здоровья.

ArticunoPokemon Articuno Понижает атаку цели на одну ступень.

WailordPokemon Wailord теряет 3 здоровья.

WailordPokemon Wailord уменьшает атаку.

ArticunoPokemon Articuno Имеет 30% шанс сбить цель с толку.

WailordPokemon Wailord теряет 2 здоровья.

WailordPokemon Wailord теряет сознание.

WhismurPokemon Whismur из команды черных вступает в бой!

WhismurPokemon Whismur Повышает Атаку цели на две ступени и сбивает её с толку.

ArticunoPokemon Articuno увеличивает атаку.

ArticunoPokemon Articuno Имеет 10% шанс ожечь цель.

WhismurPokemon Whismur теряет 7 здоровья.

WhismurPokemon Whismur промахивается

ArticunoPokemon Articuno Имеет 30% шанс сбить цель с толку.

WhismurPokemon Whismur теряет 4 здоровья.

WhismurPokemon Whismur промахивается

ArticunoPokemon Articuno имеет 30% вероятность заставить цель дрогнуть.

WhismurPokemon Whismur теряет 3 здоровья.

WhismurPokemon Whismur Повышает Атаку цели на две ступени и сбивает её с толку.

ArticunoPokemon Articuno увеличивает атаку.

ArticunoPokemon Articuno Имеет 10% шанс ожечь цель.

WhismurPokemon Whismur теряет 5 здоровья.

WhismurPokemon Whismur теряет сознание.

В команде черных не осталось покемонов.

Команда фиолетовых побеждает в этом бою!

Вывод:

В процессе выполнения лабораторной работы я получила навыки использования объектно-ориентированного подхода программирования в Java.