Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования  
**"Национальная научно-образовательная корпорация ИТМО"**

Факультет Систем Управления и Робототехники

**Лабораторная работа №2**по дисциплине **«Программирование»**

Вариант №79091

Выполнил:  
Студент группы R3141  
Афанасьев Кирилл Александрович

Преподаватель:  
Иманзаде Фахри Рашидович

Санкт-Петербург, 2022

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:

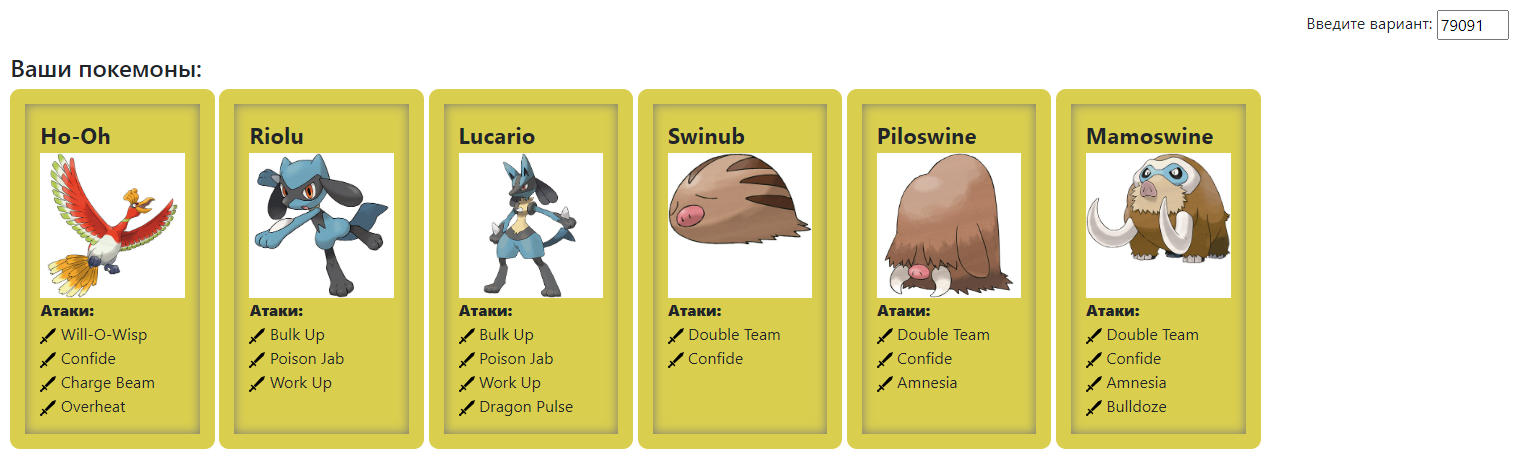
* очки здоровья (HP)
* атака (attack)
* защита (defense)
* специальная атака (special attack)
* специальная защита (special defense)
* скорость (speed)

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. Информацию о покемонах, цепочках эволюции и атаках можно найти на сайтах.



Исходный код:

Репозиторий: <https://github.com/Zerumi/no2_140922_79091>

**Class: Program (Program.java):**

import pokemons.\*;

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

public class Program {

public static void main(String[] args)

{

Battle b = new Battle();

Pokemon p1 = new HoOh("Птица Феникс",100);

Pokemon p2 = new Riolu("Абелева полугруппа", 100);

Pokemon p3 = new Lucario("Злой второкурсник", 100);

Pokemon p4 = new Swinub("Евангелион", 100);

Pokemon p5 = new Piloswine("Ктостер лимонойд", 100);

Pokemon p6 = new Mamoswine("Паймон", 100);

b.addAlly(p1);

b.addAlly(p5);

b.addAlly(p6);

b.addFoe(p2);

b.addFoe(p3);

b.addFoe(p4);

b.go(); // мясцо

}

}

**Блок покемонов (package pokemons):**

**Class HoOh (/pokemons/HoOh.java)**

package pokemons;

import attacks.specialAttacks.ChargeBeam;

import attacks.specialAttacks.Overheat;

import attacks.statusAttacks.Confide;

import attacks.statusAttacks.WillOWisp;

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

public class HoOh extends Pokemon {

public HoOh(String name, int lvl)

{

super(name, lvl);

this.setStats(106,130,90,110,154,90);

this.setType(Type.FIRE, Type.FLYING);

this.addMove(new WillOWisp());

this.addMove(new Confide());

this.addMove(new ChargeBeam());

this.addMove(new Overheat());

}

}

**Class Riolu (/pokemons/Riolu.java)**

package pokemons;

import attacks.physicalAttacks.PoisonJab;

import attacks.statusAttacks.BulkUp;

import attacks.statusAttacks.WorkUp;

import ru.ifmo.se.pokemon.Pokemon;

import ru.ifmo.se.pokemon.Type;

public class Riolu extends Pokemon {

public Riolu(String name, int lvl)

{

super(name, lvl);

this.setStats(40,70,40,35,40,60);

this.setType(Type.FIGHTING);

this.addMove(new BulkUp());

this.addMove(new PoisonJab());

this.addMove(new WorkUp());

}

}

**Class Lucario (/pokemons/Lucario.java)**

package pokemons;

import attacks.specialAttacks.DragonPulse;

import ru.ifmo.se.pokemon.Type;

public class Lucario extends Riolu {

public Lucario(String name, int lvl)

{

super(name, lvl);

this.setStats(70,110,70,115,70,90);

this.setType(Type.FIGHTING,Type.STEEL);

this.addMove(new DragonPulse());

}

}

**Class Swinub (/pokemons/Swinub.java)**

package pokemons;

import attacks.statusAttacks.Confide;

import attacks.statusAttacks.DoubleTeam;

import ru.ifmo.se.pokemon.Pokemon;

import ru.ifmo.se.pokemon.Type;

public class Swinub extends Pokemon {

public Swinub(String name, int lvl)

{

super(name, lvl);

this.setStats(50,50,40,30,30,50);

this.setType(Type.ICE, Type.GROUND);

this.addMove(new DoubleTeam());

this.addMove(new Confide());

}

}

**Class Piloswine (/pokemons/Piloswine.java)**

package pokemons;

import attacks.statusAttacks.Amnesia;

public class Piloswine extends Swinub {

public Piloswine(String name, int lvl)

{

super(name, lvl);

this.setStats(100,100,80,60,60,50);

this.addMove(new Amnesia());

}

}

**Class Mamoswine (/pokemons/Mamoswine.java)**

package pokemons;

import attacks.physicalAttacks.Bulldoze;

public class Mamoswine extends Piloswine {

public Mamoswine(String name, int lvl)

{

super(name, lvl);

this.setStats(110,130,80,70,60,80);

this.addMove(new Bulldoze());

}

}

**Блок атак (package attacks):**

**Блок статусных атак (package attacks.statusAttacks)**

**Class Amnesia (/attacks/statusAttacks/Amnesia.java)**

package attacks.statusAttacks;

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

public class Amnesia extends StatusMove {

public Amnesia()

{

super(Type.PSYCHIC,0,100);

}

@Override

protected void applySelfEffects(Pokemon pokemon)

{

pokemon.setMod(Stat.SPECIAL\_DEFENSE, 2);

super.applySelfEffects(pokemon);

}

@Override

protected String describe() {

return "применяет Amnesia";

}

}

**Class BulkUp (/attacks/statusAttacks/BulkUp.java)**

package attacks.statusAttacks;

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

public class BulkUp extends StatusMove {

public BulkUp()

{

super(Type.FIGHTING,0,100);

}

@Override

protected void applySelfEffects(Pokemon pokemon)

{

pokemon.setMod(Stat.ATTACK, 1);

pokemon.setMod(Stat.DEFENSE, 1);

super.applySelfEffects(pokemon);

}

@Override

protected String describe() {

return "применяет Bulk Up";

}

}

**Class Confide (/attacks/statusAttacks/Confide.java)**

package attacks.statusAttacks;

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

public class Confide extends StatusMove {

public Confide()

{

super(Type.NORMAL,0,100);

}

@Override

protected void applyOppEffects(Pokemon pokemon) {

pokemon.setMod(Stat.SPECIAL\_ATTACK, -1);

super.applyOppEffects(pokemon);

}

@Override

protected String describe() {

return "применяет Confide";

}

}

**Class DoubleTeam (/attacks/statusAttacks/DoubleTeam.java)**

package attacks.statusAttacks;

package attacks.statusAttacks;

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,0,100);

}

@Override

protected void applySelfEffects(Pokemon pokemon) {

pokemon.setMod(Stat.EVASION, 1);

super.applySelfEffects(pokemon);

}

@Override

protected String describe() {

return "применяет Double Team";

}

}

**Class WillOWisp (/attacks/statusAttacks/WillOWisp.java)**

package attacks.statusAttacks;

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

public class WillOWisp extends StatusMove {

public WillOWisp()

{

super(Type.FIRE,0,85);

}

@Override

protected void applyOppEffects(Pokemon pokemon) {

Effect effect = new Effect().condition(Status.BURN).turns(-1);

pokemon.addEffect(effect);

super.applyOppEffects(pokemon);

}

@Override

protected String describe() {

return "применяет Will-O-Wisp";

}

}

package attacks.statusAttacks;

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

**Class WorkUp (/attacks/statusAttacks/WorkUp.java)**

public class WorkUp extends StatusMove {

public WorkUp()

{

super(Type.NORMAL,0,100);

}

@Override

protected void applySelfEffects(Pokemon pokemon)

{

pokemon.setMod(Stat.ATTACK, 1);

pokemon.setMod(Stat.SPECIAL\_ATTACK, 1);

super.applySelfEffects(pokemon);

}

@Override

protected String describe() {

return "применяет Work Up";

}

}

**Блок физических атак (/attacks/physicalAttacks):**

**Class Bulldoze (/attacks/physicalAttacks/Bulldoze.java)**

package attacks.physicalAttacks;

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

public class Bulldoze extends PhysicalMove {

public Bulldoze()

{

super(Type.GROUND,60,100);

}

@Override

protected void applyOppEffects(Pokemon pokemon)

{

pokemon.setMod(Stat.SPEED, -1);

super.applyOppEffects(pokemon);

}

@Override

protected String describe() {

return "применяет Bulldoze";

}

}

**Class PoisonJab (/attacks/physicalAttacks/PoisonJab.java)**

package attacks.physicalAttacks;

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

public class PoisonJab extends PhysicalMove {

public PoisonJab()

{

super(Type.POISON,80,100);

}

@Override

protected void applyOppEffects(Pokemon pokemon)

{

Effect effect = new Effect().chance(0.3).condition(Status.POISON);

pokemon.addEffect(effect);

super.applyOppEffects(pokemon);

}

@Override

protected String describe() {

return "применяет PoisonJab";

}

}

**Блок специальных атак (/attacks/specialAttacks):**

**Class ChargeBeam (/attacks/specialAttacks/ChargeBeam.java):**

package attacks.specialAttacks;

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

public class ChargeBeam extends SpecialMove {

public ChargeBeam()

{

super(Type.ELECTRIC, 50, 90);

}

@Override

protected void applySelfEffects(Pokemon pokemon) {

Effect effect = new Effect().chance(0.7).stat(Stat.SPECIAL\_ATTACK, 1).turns(-1);

pokemon.addEffect(effect);

super.applySelfEffects(pokemon);

}

@Override

protected String describe() {

return "применяет Charge Beam!!";

}

}

**Class DragonPulse (/attacks/specialAttacks/DragonPulse.java):**

package attacks.specialAttacks;

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

public class DragonPulse extends SpecialMove {

public DragonPulse()

{

super(Type.DRAGON, 85, 100);

}

@Override

protected String describe() {

return "применяет Dragon Pulse!!";

}

}

**Class Overheat (/attacks/specialAttacks/Overheat.java):**

package attacks.specialAttacks;

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

public class Overheat extends SpecialMove {

public Overheat()

{

super(Type.FIRE, 130, 90);

}

@Override

protected void applySelfEffects(Pokemon pokemon) {

pokemon.setMod(Stat.SPECIAL\_ATTACK, -2);

super.applySelfEffects(pokemon);

}

@Override

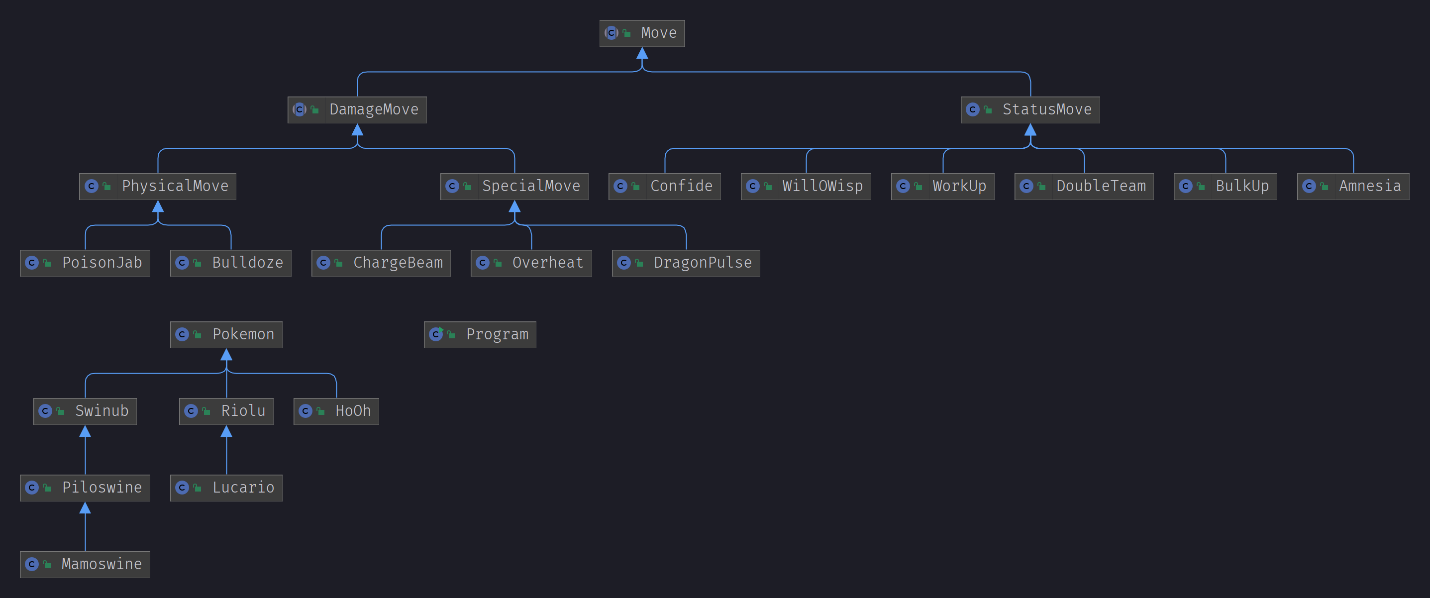
protected String describe() {

return "применяет Overheat!!";

}

}

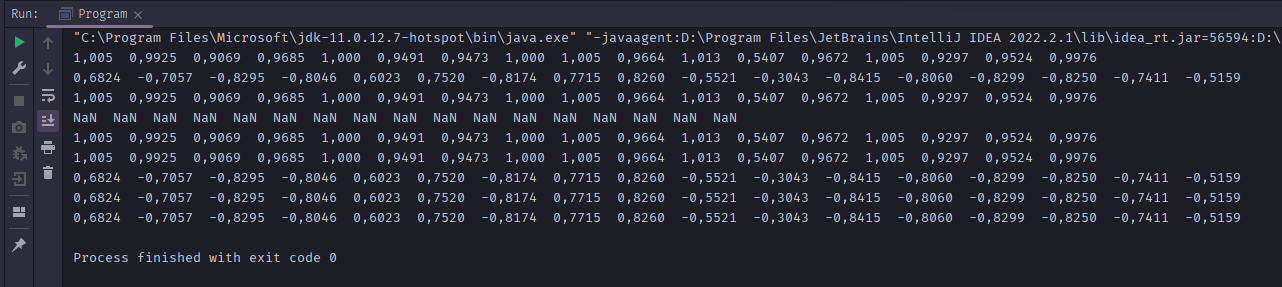
Диаграмма классов реализованной объектной модели. Иерархия объектов.



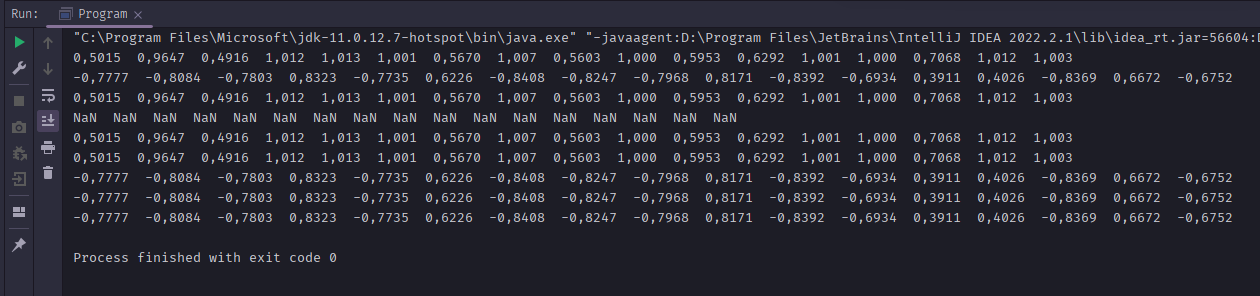
Вывод в .graphml см. в репозитории.

Результаты работы программы

Результат #1:



Результат #2:



Вывод

Во время выполнения данной лабораторной работы я

* Ознакомился с базовым синтаксисом языка Java
* Ознакомился с интегрированной средой разработки JetBrains IntelliJ IDEA Ultimate и ее некоторыми возможностями
* Работал с примитивными типами данных, а также одномерными и двумерными массивами
* Ознакомился с классом Random для генерации псевдослучайных чисел
* Использовал циклы (в том числе и вложенные) для перебора элементов массива
* Использовал оператор switch для ветвления кода
* Ознакомился с классом Math для базовых математических вычисления, а также изучил теорию похожего класса StrictMath
* Использовал форматированный вывод данных для более удобного представления результатов работы программы
* Ознакомился с командами терминала javac, java и jar для компиляции, исполнения и упаковки Java кода
* Научился компилировать, запускать и отлаживать программу в JetBrains IntelliJ IDEA IDE, а также создавать .jar архивы
* Научился использовать систему контроля версии в IDE JetBrains
* Активно использовал комментирование для повышения читаемости кода

По окончании данной лабораторной работы я

* Умею пользоваться основными средствами разработки на Java, в том числе понимаю основное устройство языка, базовый синтаксис, написание кода в IDE/текстовом редакторе, компиляцию в IDE/терминале, исполнение в IDE/терминале, а также отладку в IDE.
* Умею оперировать примитивными данными в рамках определенной задачи на языке Java

Полученные мною знания будут использоваться для дальнейшего изучения языка и обучения в целом.