Министерство образования и науки РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

**факультет программной инженерии и компьютерной техники**

**ЛАБОРАТОРНАЯ РАБОТА № 2**

по дисциплине

‘ПРОГРАММИРОВАНИЕ’

Вариант №21541

*Выполнил:Манжиков Никита Сергеевич*

Студент группы P3115

*Преподаватель:Усков Иван Владимирович*



Санкт-Петербург, 2021

Диаграмма Классов

# Покемоны

****

# Исходный код

#### Lab1.java

import pokemons.\*;

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

public class lab2{

public static void main(String[] args){

Battle b = new Battle();

Anorith p1 = new Anorith("amogus", 3);

Armaldo p2 = new Armaldo("kito", 3);

Bayleef p3 = new Bayleef("bob", 2);

Chikorita p4 = new Chikorita("Victor", 1);

Delibird p5 = new Delibird("Evgeniy", 2);

Meganium p6 = new Meganium("Anton", 2);

b.addAlly(p1);

b.addAlly(p2);

b.addAlly(p3);

b.addFoe(p4);

b.addFoe(p5);

b.addFoe(p6);

b.go();

}

}

package pokemons;

# Anorith

import moves.\*;

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

public class Anorith extends Pokemon {

public Anorith(String name, int level) {

super(name, level);

setStats(40, 40, 55, 40, 70, 55);

setType(Type.ROCK, Type.BUG);

setMove(new RockPolish(),new Facade(),new Screech());

}

}

package pokemons;

# Armaldo

import moves.\*;

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

public class Armaldo extends Anorith {

public Armaldo(String name, int level) {

super(name, level);

setStats(75, 125, 100, 70, 80, 45);

setType(Type.ROCK, Type.BUG);

addMove(new Blizzard());

}

}

package pokemons;

# Bayleef

import moves.\*;

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

public class Bayleef extends Chikorita {

public Bayleef(String name, int level) {

super(name, level);

setStats(55, 75, 50, 40, 40, 80);

setType(Type.GRASS);

addMove(new WorkUp());

}

}

# Chirikota

package pokemons;

import moves.\*;

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

public class Chikorita extends Pokemon {

public Chikorita(String name, int level) {

super(name, level);

setStats(40, 55, 30, 30, 30, 60);

setType(Type.NORMAL, Type.FLYING);

setMove(new Facade(), new DoubleTeam());

}

}

package pokemons;

import moves.\*;

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

# DeLibird

public class Delibird extends Pokemon {

public Delibird(String name, int level) {

super(name, level);

setStats(45, 55, 45, 65, 45, 75);

setType(Type.ICE, Type.FLYING);

setMove(new IceShard(), new HydroPump(), new Waterfall(), new Confide());

}

}

package pokemons;

# Meganium

import moves.\*;

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

public class Meganium extends Bayleef {

public Meganium(String name, int level) {

super(name, level);

setStats(85, 120, 70, 50, 60, 100);

setType(Type.NORMAL, Type.FLYING);

addMove(new SignalBeam());

}

}

# Moves

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 Blizzard extends SpecialMove {

public Blizzard() {

super(Type.ICE, 70, 110);

}

@Override

protected boolean checkAccuracy(Pokemon att, Pokemon def) {

return true;

}

@Override

protected void applyOppEffects(Pokemon p) {

if (Math.random() < 0.1)

Effect.freeze(p);

}

@Override

protected String describe() {

return "замораживает

";

}

}

package moves;

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

public class Confide extends StatusMove {

public Confide() {

super(Type.NORMAL, 0, 100);

}

@Override

protected void applyOppEffects(Pokemon p) {

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

}

@Override

protected String describe() {

return "Использует Confide";

}

}

package moves;

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

public class DoubleTeam extends StatusMove {

public DoubleTeam() {

super(Type.NORMAL, 0, 100);

}

@Override

protected void applySelfEffects(Pokemon p) {

p.setMod(Stat.EVASION, 1);

}

@Override

protected String describe() {

return " Использует Double Team";

}

}

package moves;

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

public class Facade extends PhysicalMove {

public Facade() {

super(Type.NORMAL, 70, 100);

}

@Override

protected void applyOppDamage(Pokemon def, double damage) {

Status p\_stat = def.getCondition();

if (p\_stat.equals(Status.BURN) || p\_stat.equals(Status.POISON) || p\_stat.equals(Status.PARALYZE)) {

super.applyOppDamage(def, damage\*2);

}

else {

super.applyOppDamage(def, damage);

}

}

@Override

protected String describe() {

return " Использует Facade";

}

}

package moves;

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

public class HydroPump extends SpecialMove {

public HydroPump() {

super(Type.WATER, 80, 110);

}

@Override

protected void applyOppEffects(Pokemon p) {

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

}

@Override

protected String describe() {

return " Использует Hydro Pump";

}

}

package moves;

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

public class IceShard extends PhysicalMove{

public IceShard(){

super(Type.ICE, 40, 100);

}

@Override

protected void applySelfEffects(Pokemon p) {

p.setMod(Stat.SPECIAL\_ATTACK, 4);

}

@Override

protected String describe() {

return “Использует Ice Shard";

}

}

package moves;

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

public class RockPolish extends StatusMove {

public RockPolish() {

super(Type.ROCK, 0, 0);

}

@Override

protected void applySelfEffects(Pokemon p) {

Effect e1 = new Effect().turns(0).stat(Stat.SPEED, 2);

p.addEffect(e1);

}

@Override

protected String describe() {

return " Использует Rock Polish";

}

}

package moves;

import ru.ifmo.se.pokemon.PhysicalMove;

import ru.ifmo.se.pokemon.StatusMove;

import ru.ifmo.se.pokemon.Type;

public class Screech extends StatusMove {

public Screech() {

super(Type.NORMAL, 18, 80);

}

@Override

protected String describe() {

return "–љ–∞–љ–Њ—Б–Є—В —П—А–Њ—Б—В–љ—Л–µ —Г–і–∞—А—Л";

}

}

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 SignalBeam extends SpecialMove {

public SignalBeam() {

super(Type.BUG, 75, 100);

}

@Override

protected boolean checkAccuracy(Pokemon att, Pokemon def) {

return true;

}

@Override

protected void applyOppEffects(Pokemon p) {

if (Math.random() < 0.1)

Effect.confuse(p);

}

@Override

protected String describe() {

return “замораживает";

}

}

package moves;

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

public class Waterfall extends PhysicalMove {

public Waterfall() {

super(Type.WATER, 80, 100);

}

@Override

protected void applyOppEffects(Pokemon p) {

if (Math.random() < 0.2) {

Effect.flinch(p);

}

}

@Override

protected String describe() {

return "Использует Waterfall";

}

}

package moves;

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

public class WorkUp extends StatusMove {

public WorkUp() {

super(Type.NORMAL, 0, 0);

}

@Override

protected String describe() {

return super.describe() + " [ Power: " + this.power + ", " + "Accuracy: " +

this.accuracy + ", " + "Type: " + this.type + " ] ";

}

@Override

protected void applySelfEffects(Pokemon pokemon) {

super.applySelfEffects(pokemon);

pokemon.setMod(Stat.ATTACK, +1);

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

}

}

# Результат работы

Anorith amogus из команды полосатых вступает в бой!

Chikorita Victor из команды красных вступает в бой!

Anorith amogus промахивается

Chikorita Victor использует Double Team.

Chikorita Victor увеличивает уклоняемость.

Anorith amogus промахивается

Chikorita Victor использует Facade.

Anorith amogus теряет 3 здоровья.

Anorith amogus промахивается

Chikorita Victor использует Facade.

Anorith amogus теряет 4 здоровья.

Anorith amogus наносит яростные удары.

Chikorita Victor использует Double Team.

Chikorita Victor увеличивает уклоняемость.

Anorith amogus наносит яростные удары.

Chikorita Victor использует Double Team.

Chikorita Victor увеличивает уклоняемость.

Anorith amogus наносит яростные удары.

Chikorita Victor использует Facade.

Anorith amogus теряет 2 здоровья.

Anorith amogus использует Facade.

Chikorita Victor теряет 7 здоровья.

Chikorita Victor использует Double Team.

Chikorita Victor увеличивает уклоняемость.

Anorith amogus использует Facade.

Chikorita Victor теряет 7 здоровья.

Chikorita Victor теряет сознание.

Delibird Evgeniy из команды красных вступает в бой!

Anorith amogus наносит яростные удары.

Delibird Evgeniy использует Confide.

Anorith amogus уменьшает специальную атаку.

Anorith amogus использует Facade.

Delibird Evgeniy теряет 5 здоровья.

Delibird Evgeniy использует Hydro Pump.

Anorith amogus теряет 8 здоровья.

Anorith amogus уменьшает специальную атаку.

Anorith amogus теряет сознание.

Armaldo kito из команды полосатых вступает в бой!

Delibird Evgeniy использует Waterfall.

Armaldo kito теряет 12 здоровья.

Armaldo kito замораживает.

Delibird Evgeniy теряет 7 здоровья.

Delibird Evgeniy использует Ice Shard.

Armaldo kito теряет 6 здоровья.

Delibird Evgeniy увеличивает специальную атаку.

Armaldo kito теряет сознание.

Bayleef bob из команды полосатых вступает в бой!

Bayleef bob использует Double Team.

Bayleef bob увеличивает уклоняемость.

Delibird Evgeniy использует Confide.

Bayleef bob уменьшает специальную атаку.

Bayleef bob промахивается

Delibird Evgeniy использует Confide.

Bayleef bob уменьшает специальную атаку.

Bayleef bob использует Facade.

Delibird Evgeniy теряет 4 здоровья.

Delibird Evgeniy теряет сознание.

Meganium Anton из команды красных вступает в бой!

Meganium Anton использует Facade.

Bayleef bob теряет 11 здоровья.

Bayleef bob использует Facade.

Meganium Anton теряет 3 здоровья.

Meganium Anton использует Facade.

Bayleef bob теряет 7 здоровья.

Bayleef bob теряет сознание.

В команде полосатых не осталось покемонов.

Команда красных побеждает в этом бою!

# Вывод

Во время выполнения данной лабораторной работы я изучил основные принципы ООП, научился работать с методами, классами, модификаторами доступами и сторонними библиотеками.