НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет Нейротехнологии и программирование

Дисциплина

Лабораторная работа № 2

Выполнил студент

Торениязов Арман Ажинияз улы

Группа № 3.6

Преподаватель: Байрамова Хумай Бахруз Кызы

г. Санкт-Петербург

2023

Оглавление

[Задание 2](#_Toc153026048)

[Структура проекта 4](#_Toc153026049)

[Main.java 4](#_Toc153026050)

[mypokemons 5](#_Toc153026051)

[Gloom.java 5](#_Toc153026052)

[Lapras.java 5](#_Toc153026053)

[Mantine.java 5](#_Toc153026054)

[Mantyke.java 6](#_Toc153026055)

[Oddish.java 6](#_Toc153026056)

[Vileplume.java 6](#_Toc153026057)

[mymoves 7](#_Toc153026058)

[Bubble.java 7](#_Toc153026059)

[ConfuseRay.java 7](#_Toc153026060)

[Haze.java 8](#_Toc153026061)

[IseShard.java 8](#_Toc153026062)

[MegaDrain.java 8](#_Toc153026063)

[Rest.java 9](#_Toc153026064)

[RockTomb.java 9](#_Toc153026065)

[SleepPowder.java 9](#_Toc153026066)

[SludgeBomb.java 10](#_Toc153026067)

[Tackle.java 10](#_Toc153026068)

[Waterfall.java 11](#_Toc153026069)

[Результат программы: 11](#_Toc153026070)

[Вывод 12](#_Toc153026071)

**Вариант:** 348274

# 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:

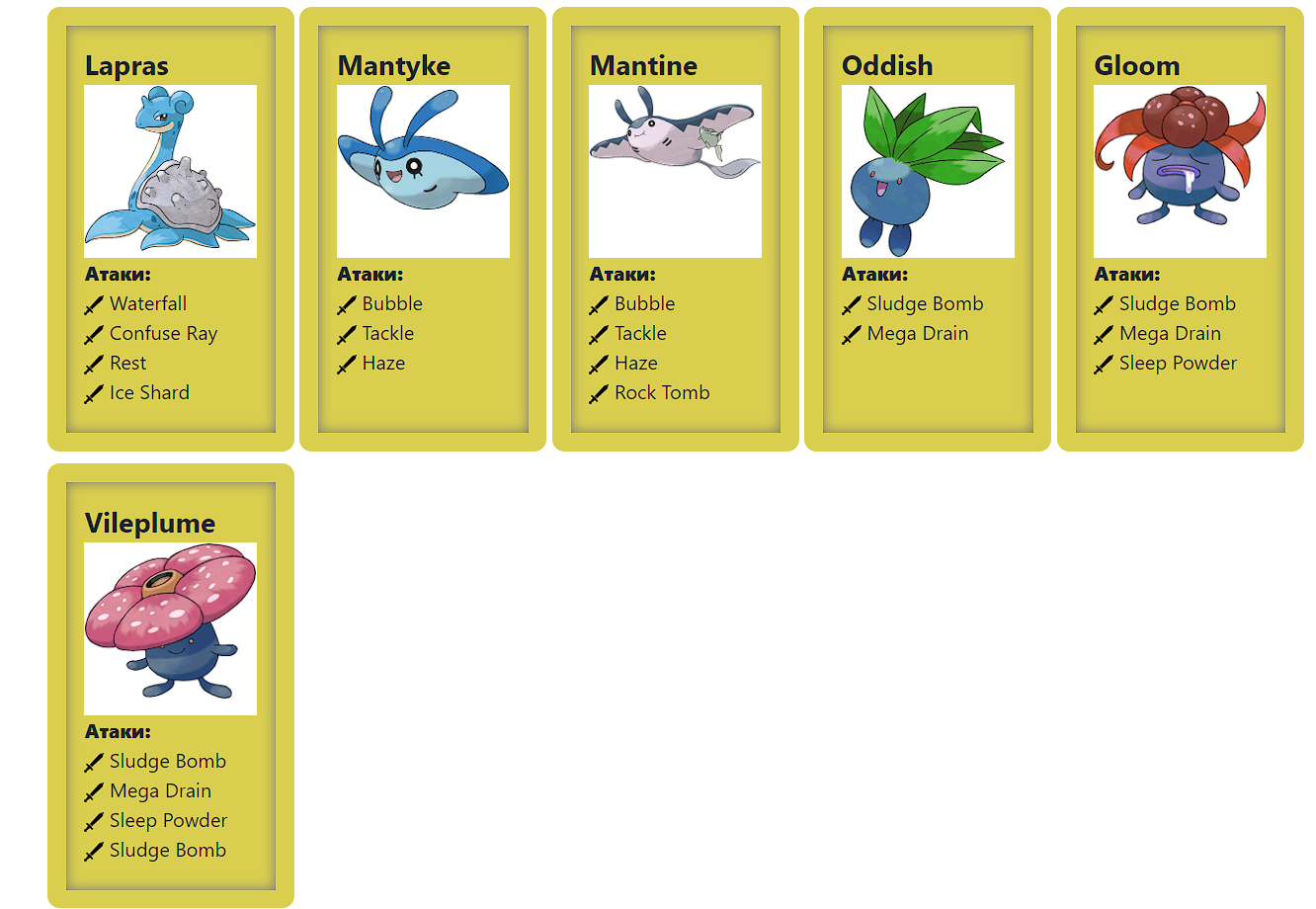
* очки здоровья (HP)
* атака (attack)
* защита (defense)
* специальная атака (special attack)
* специальная защита (special defense)
* скорость (speed)

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.

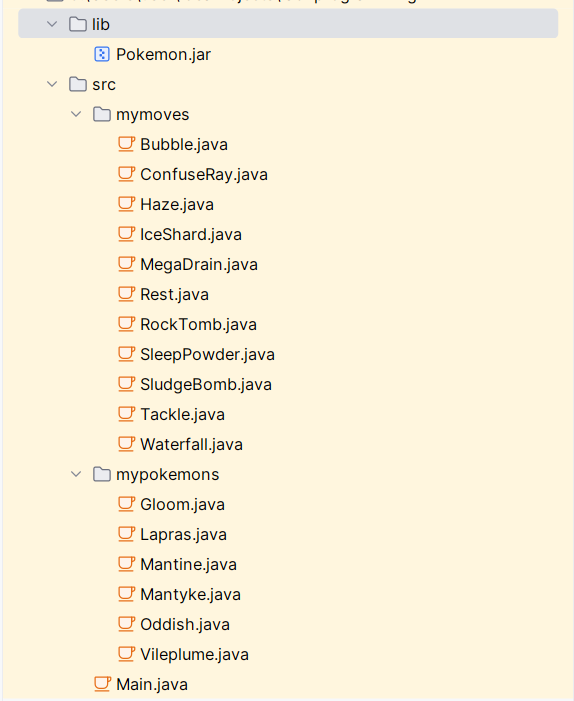
Атака должна иметь стандартные тип, силу (power) и точность (accuracy). Должны быть реализованы стандартные эффекты атаки. Назначить каждому виду покемонов атаки в соответствии с вариантом. Уровень покемона выбирается минимально необходимым для всех реализованных атак.

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/).



# Структура проекта

****

# Main.java

import mypokemons.\*;  
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 a1 = new Lapras("Lapras", 1);  
 Pokemon a2 = new Mantyke("Mantyke", 1);  
 Pokemon a3 = new Mantine("Mantine", 1);  
  
 Pokemon b1 = new Oddish("Oddish", 1);  
 Pokemon b2 = new Gloom("Gloom", 1);  
 Pokemon b3 = new Vileplume("Vileplume", 1);  
  
 b.addAlly(a1);  
 b.addAlly(a2);  
 b.addAlly(a3);  
  
 b.addFoe(b1);  
 b.addFoe(b2);  
 b.addFoe(b3);  
 b.go();  
 }  
  
 public static boolean chance(double c) {  
 return (1 - c) >= Math.random();  
 }  
}

# mypokemons

## Gloom.java

package mypokemons;  
  
import mymoves.MegaDrain;  
import mymoves.SleepPowder;  
import mymoves.SludgeBomb;  
import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
public class Gloom extends Oddish {  
 public Gloom(String name, int level) {  
 super(name, level);  
 super.setStats(60, 65, 70, 85, 75, 40);  
 super.setType(Type.GRASS, Type.POISON);  
  
 setMove(new SludgeBomb(), new MegaDrain(), new SleepPowder());  
 }  
}

## Lapras.java

package mypokemons;  
  
import mymoves.ConfuseRay;  
import mymoves.Rest;  
import mymoves.Waterfall;  
import mymoves.Waterfall;  
import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
public class Lapras extends Pokemon {  
 public Lapras(String name, int level) {  
 super(name, level);  
 setStats(130, 85, 80,85, 95, 60);  
 setType(Type.WATER, Type.ICE);  
  
 super.setMove(new Waterfall(), new ConfuseRay(), new Rest());  
 }  
}

## Mantine.java

package mypokemons;  
  
import mymoves.Bubble;  
import mymoves.Haze;  
import mymoves.RockTomb;  
import mymoves.Tackle;  
import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
public class Mantine extends Pokemon {  
 public Mantine(String name, int level) {  
 super(name, level);  
 setStats(85, 40, 70, 80, 140,70);  
 setType(Type.WATER, Type.FLYING);  
  
 super.setMove(new Bubble(), new Tackle(), new Haze(), new RockTomb());  
 }  
}

## Mantyke.java

package mypokemons;  
  
import mymoves.\*;  
import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
public class Mantyke extends Mantine {  
 public Mantyke(String name, int level) {  
 super(name, level);  
 setStats(45, 20, 50, 60, 120, 50);  
 setType(Type.WATER, Type.FLYING);  
  
 super.setMove(new Bubble(), new Tackle(), new Haze());  
 }  
}

## Oddish.java

package mypokemons;  
  
import mymoves.\*;  
import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;  
  
public class Oddish extends Pokemon {  
 public Oddish(String name, int level) {  
 super(name, level);  
 setStats(45, 50, 55, 75, 65, 30);  
 setType(Type.GRASS, Type.POISON);  
  
 super.setMove(new SludgeBomb(), new MegaDrain());  
 }  
}

## Vileplume.java

package mypokemons;  
  
import mymoves.MegaDrain;  
import mymoves.SleepPowder;  
import mymoves.SludgeBomb;  
import ru.ifmo.se.pokemon.Type;  
  
public class Vileplume extends Gloom {  
 {  
  
 }  
 public Vileplume(String name, int level) {  
 super(name, level);  
 setStats(75, 80, 85,110, 90, 50);  
 setType(Type.GRASS, Type.POISON);  
  
 setMove(new SludgeBomb(), new MegaDrain(), new SleepPowder());  
 }  
}

# mymoves

## Bubble.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Bubble extends SpecialMove {  
 public Bubble(){  
 super(Type.WATER, 40, 100 );  
 }  
  
 @Override  
 protected void applyOppDamage(Pokemon pokemon, double v) {  
 pokemon.setMod(Stat.HP, (int) v);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon p) {  
 if (Math.random() < 0.1) {  
 p.setMod(Stat.SPEED, -1);  
 Effect.flinch(p);  
 }  
 }  
  
 @Override  
 protected String describe(){  
 return "использует Bubble";  
 }  
}

## ConfuseRay.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class ConfuseRay extends StatusMove {  
 public ConfuseRay() {  
 super(Type.GHOST, 0, 100);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon pokemon) {  
 super.applyOppEffects(pokemon);  
 if (Math.random() < 0.33) {  
 Effect.confuse(pokemon);  
 }  
 }  
  
 @Override  
 protected String describe() {  
 return "использует ConfuseRay";  
 }  
}

## Haze.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Haze extends StatusMove {  
 public Haze(){  
 super(Type.ICE, 0, 0 );  
 }  
  
 @Override  
 protected void applySelfEffects(Pokemon p) {  
 p.restore();  
 }  
  
 @Override  
 protected String describe(){  
 return "использует Haze";  
 }  
}

## IseShard.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class IceShard extends PhysicalMove {  
 public IceShard() {super(Type.ICE, 40, 100);}  
  
 @Override  
 protected void applyOppDamage(Pokemon pokemon, double v) {pokemon.setMod(Stat.HP, (int) v);}  
 @Override  
 protected void applySelfEffects(Pokemon p) { *// библиотека не расчитана на выставление приоритетов атак. оставим метод пустым}* }  
 protected String describe(){  
 return "использует IceShard";  
 }  
  
}

## MegaDrain.java

package mymoves;  
  
import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.SpecialMove;  
import ru.ifmo.se.pokemon.Stat;  
import ru.ifmo.se.pokemon.Type;  
  
public class MegaDrain extends SpecialMove {  
 public MegaDrain() {super(Type.GRASS, 40, 100);}  
 @Override  
 public void applyOppDamage(Pokemon p, double damage) {  
 p.setMod(Stat.HP, (int) damage);  
 }  
 @Override  
 public String describe() {  
 return "использует MegaDrain";  
 }  
}

## Rest.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Rest extends StatusMove {  
 public Rest(){  
 super(Type.PSYCHIC, 0, 0 );  
 }  
 @Override  
 protected void applySelfEffects(Pokemon p){  
 Effect effect = (new Effect()).condition(Status.SLEEP).turns(2);  
 p.setMod(Stat.HP, (int) -73.00);  
 p.setCondition(effect);  
 }  
  
 @Override  
 protected String describe(){  
 return "использует Rest";  
 }  
}

## RockTomb.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class RockTomb extends PhysicalMove {  
 public RockTomb(){  
 super(Type.ROCK, 60, 95);  
 }  
  
 @Override  
 protected void applyOppDamage(Pokemon p, double damage) {  
 p.setMod(Stat.HP, (int) damage);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p){  
 p.setMod(Stat.SPEED, -1);  
 }  
  
 @Override  
 protected String describe(){  
 return "использует RockTomb";  
 }  
}

## SleepPowder.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class SleepPowder extends StatusMove {  
 public SleepPowder(){  
 super(Type.GRASS, 0, 75 );  
 }  
  
 @Override  
 public void applyOppEffects(Pokemon p) {  
 var turns = (int) (Math.random() \* 2 + 1);  
 Effect e = (new Effect()).condition(Status.SLEEP).turns(turns);  
 p.setCondition(e);  
 }  
 @Override  
 protected String describe(){  
 return "использует SleepPowder";  
 }  
}

## SludgeBomb.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class SludgeBomb extends SpecialMove {  
 public SludgeBomb(){  
 super(Type.POISON, 90, 100);  
 }  
  
 @Override  
 protected void applyOppDamage(Pokemon p, double damage) {  
 p.setMod(Stat.HP, (int) damage);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p){  
 if (Math.random() < 0.3) {  
 Effect.poison(p);  
 }  
 }  
  
 @Override  
 protected String describe(){  
 return "использует SludgeBomb";  
 }  
}

## Tackle.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Tackle extends PhysicalMove {  
 public Tackle(){  
 super(Type.NORMAL, 40, 100 );  
 }  
 @Override  
 protected void applyOppDamage(Pokemon pokemon, double v) {  
 pokemon.setMod(Stat.HP, (int) v);  
 }  
  
 @Override  
 protected String describe(){  
 return "использует Tackle";  
 }  
}

## Waterfall.java

package mymoves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Waterfall extends PhysicalMove {  
 public Waterfall(){  
 super(Type.WATER, 80, 100);  
 }  
 protected void applyOppDamage(Pokemon pokemon, double v) {  
 pokemon.setMod(Stat.HP, (int) v);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p) {  
 if (Math.random() < 0.2) {  
 Effect.flinch(p);  
 }  
 }  
 @Override  
 public String describe(){  
 return "использует Waterfall";  
 }  
}

# Результат программы:

Lapras Lapras from the team Yellow enters the battle!

Oddish Oddish from the team White enters the battle!

Lapras Lapras использует Waterfall.

Oddish Oddish loses 5 hit points.

Oddish Oddish использует MegaDrain.

Lapras Lapras loses 12 hit points.

Lapras Lapras использует ConfuseRay.

Oddish Oddish использует SludgeBomb.

Lapras Lapras loses 10 hit points.

Lapras Lapras is poisoned

Lapras Lapras faints.

Mantyke Mantyke from the team Yellow enters the battle!

Mantyke Mantyke использует Tackle.

Oddish Oddish loses 4 hit points.

Oddish Oddish использует MegaDrain.

Mantyke Mantyke loses 4 hit points.

Mantyke Mantyke misses

Oddish Oddish использует MegaDrain.

Mantyke Mantyke loses 5 hit points.

Mantyke Mantyke misses

Oddish Oddish использует MegaDrain.

Mantyke Mantyke loses 7 hit points.

Mantyke Mantyke faints.

Mantine Mantine from the team Yellow enters the battle!

Mantine Mantine misses

Oddish Oddish использует SludgeBomb.

Mantine Mantine loses 6 hit points.

Mantine Mantine misses

Oddish Oddish использует SludgeBomb.

Mantine Mantine loses 9 hit points.

Mantine Mantine is poisoned

Mantine Mantine faints.

Team Yellow loses its last Pokemon.

The team White wins the battle!

# Вывод

В процессе выполнения данной лабораторной работы я изучил 3 принципа ООП и научился их применять: наследование, полиморфизм, инкапсуляцию. Также узнал про модификаторы доступа, модификаторы final и static, области видимости переменных