Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Направление подготовки: 09.03.01 -Информатика и вычислительная техника, Компьютерные системы и технологии

Отчет

**по лабораторной работе №2**

по дисциплине

«Программирование»

**Вариант 777111**

Выполнила: Кручинина Дарья Сергеевна

Группа: P3131

Преподаватель: Николаев Владимир Вячеславович

г. Санкт-Петербург

2022 г

Оглавление

[Задание 3](#_Toc116826709)

[Покемоны 3](#_Toc116826710)

[UML-диаграмма классов 4](#_Toc116826711)

[Выполнение задания 4](#_Toc116826712)

[Результат работы 8](#_Toc116826713)

[Выводы 10](#_Toc116826714)

# 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:

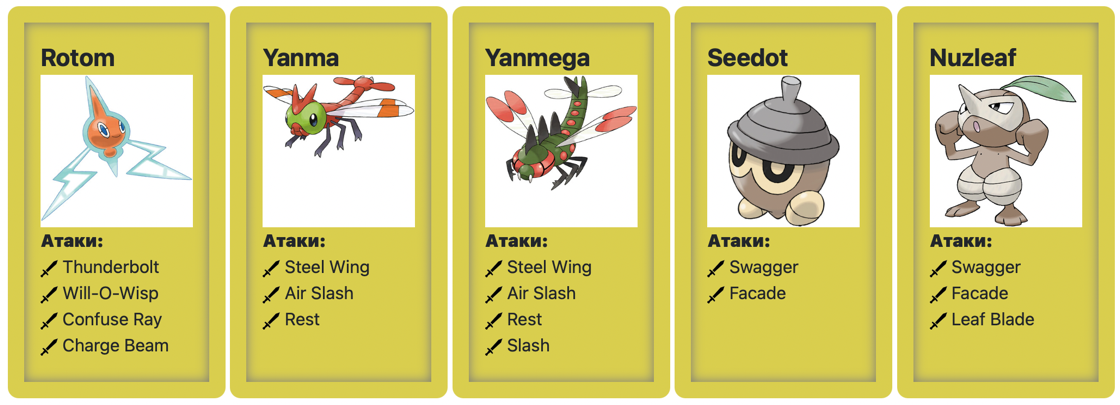
* очки здоровья (HP)
* атака (attack)
* защита (defense)
* специальная атака (special attack)
* специальная защита (special defense)
* скорость (speed)

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.

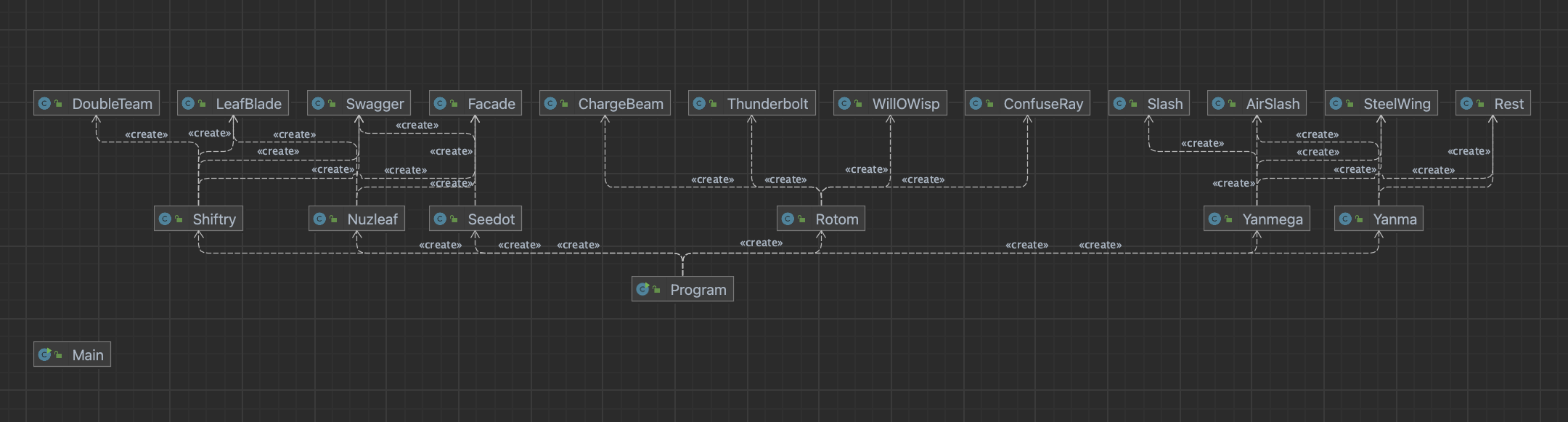
Атака должна иметь стандартные тип, силу (power) и точность (accuracy). Должны быть реализованы стандартные эффекты атаки. Назначить каждому виду покемонов атаки в соответствии с вариантом. Уровень покемона выбирается минимально необходимым для всех реализованных атак.

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

# Покемоны



# UML-диаграмма классов



# Выполнение задания

**Атаки**

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

public class AirSlash extends SpecialMove {  
 public AirSlash(){  
 super(Type.*FLYING*,75,95);  
 }  
 @Override  
 public void applyOppEffects(Pokemon p) {  
 if (Math.*random*() <= 0.3) {  
 Effect.*flinch*(p);  
 }  
 }  
 @Override  
 protected String describe() {  
 return "does AirSlash";  
 }  
}

public class ChargeBeam extends SpecialMove {  
 public ChargeBeam(){  
 super(Type.*ELECTRIC*,50,90);  
 }  
 @Override  
 protected void applySelfEffects(Pokemon p) {  
 p.addEffect(new Effect().chance(0.7).turns(1).stat(Stat.*SPECIAL\_ATTACK*, 3));  
 }  
 @Override  
 protected String describe() {  
 return "does ChargeBeam";  
 }  
}

public class ConfuseRay extends StatusMove {  
 public ConfuseRay(){  
 super(Type.*GHOST*,0,100);  
 }  
 @Override  
 protected void applySelfEffects (Pokemon p) {  
 Effect.*confuse*(p);  
 if (Math.*random*() < 0.33) {  
 p.setMod(Stat.*HP*, -40);  
 }  
 }  
 @Override  
 protected String describe() {  
 return "does ConfuseRay";  
 }  
}

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 "does DoubleTeam";  
 }  
}

public class Facade extends PhysicalMove {  
 public Facade(){  
 super(Type.*NORMAL*,70,100);  
 }  
 @Override  
 public void applyOppDamage(Pokemon p, double damage){  
 Status cond = p.getCondition();  
 if (cond.equals(Status.*POISON*) || cond.equals(Status.*BURN*) || cond.equals(Status.*PARALYZE*)) {  
 p.setMod(Stat.*HP*, 2\*(int)Math.*round*(damage));  
 }  
 }  
 @Override  
 protected String describe() {  
 return "does Facade";  
 }  
}

public class LeafBlade extends PhysicalMove {  
 public LeafBlade(){  
 super(Type.*GRASS*,90,100);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p) {  
 p.addEffect(new Effect().chance(0.1).turns(-1).stat(Stat.*SPECIAL\_DEFENSE*, -1));  
 }  
 @Override  
 protected String describe() {  
 return "does LeafBlade";  
 }  
}

public class Rest extends StatusMove {  
 public Rest(){  
 super(Type.*PSYCHIC*,0,0);  
 }  
 @Override  
 protected void applySelfEffects(Pokemon p){  
 Effect e1 = new Effect().turns(2).condition(Status.*SLEEP*);  
 Effect e = new Effect().turns(0).stat(Stat.*HP*,-99999);  
 p.addEffect(e);  
 p.setCondition(e1);  
 }  
 @Override  
 protected String describe() {  
 return "does Rest";  
 }  
}

public class Slash extends PhysicalMove {  
 public Slash(){  
 super(Type.*NORMAL*,70,100);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p) {  
 p.addEffect(new Effect().chance(0.1).turns(-1).stat(Stat.*SPECIAL\_DEFENSE*, -1));  
 }  
 @Override  
 protected String describe() {  
 return "does Slash";  
 }  
}

public class SteelWing extends PhysicalMove {  
 public SteelWing(){  
 super(Type.*STEEL*,70,90);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p) {  
 p.addEffect(new Effect().chance(0.1).turns(-1).stat(Stat.*SPECIAL\_DEFENSE*, -1));  
 }  
 @Override  
 protected String describe() {  
 return "does SteelWing";  
 }  
}

public class Swagger extends StatusMove {  
 public Swagger(){  
 super(Type.*NORMAL*,0,85);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p){  
 p.confuse();  
 p.setMod(Stat.*ATTACK*,2);  
 }  
 @Override  
 protected String describe() {  
 return "does Swagger";  
 }  
}

public class Thunderbolt extends SpecialMove {  
 public Thunderbolt() {  
 super(Type.*ELECTRIC*,90,100);  
 }  
 @Override  
 public void applyOppEffects(Pokemon p) {  
 if (Math.*random*() <= 0.1) {  
 Effect.*paralyze*(p);  
 }  
 }  
 @Override  
 protected String describe() {  
 return "does Thunderbolt";  
 }  
}

public class WillOWisp extends StatusMove {  
 public WillOWisp() {  
 super(Type.*FIRE*, 0, 85);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon p) {  
 Effect e = new Effect().stat(Stat.*ATTACK*, -3).condition(Status.*BURN*);  
 p.addEffect(e);  
 }  
 @Override  
 protected String describe() { return "does WillOWisp";}  
  
}

**Покемоны**

package mypokemons;  
  
import mymoves.\*;  
import ru.ifmo.se.pokemon.Pokemon;  
import ru.ifmo.se.pokemon.Type;

public class Nuzleaf extends Pokemon {  
 public Nuzleaf(String name, int level) {  
 super(name, level);  
 super.setType(Type.*GRASS*,Type.*DARK*);  
 super.setStats(70,70,40,60,40,60);  
 super.setMove(new Swagger(), new Facade(), new LeafBlade());  
 }  
}

public class Rotom extends Pokemon {  
 public Rotom(String name, int level) {  
 super(name, level);  
 super.setType(Type.*ELECTRIC*,Type.*GHOST*);  
 super.setStats(50, 50, 77, 95, 77,91);  
 super.setMove(new Thunderbolt(), new WillOWisp(), new ConfuseRay(), new ChargeBeam());  
 }  
}

public class Seedot extends Pokemon {  
 public Seedot(String name, int level) {  
 super(name, level);  
 super.setType(Type.*GRASS*);  
 super.setStats(40,40,50,30,30,30);  
 super.setMove(new Swagger(), new Facade());  
 }  
}

public class Shiftry extends Pokemon {  
 public Shiftry(String name, int level) {  
 super(name, level);  
 super.setType(Type.*GRASS*,Type.*DARK*);  
 super.setStats(90,100,60,90,60,80);  
 super.setMove(new Swagger(), new Facade(), new LeafBlade(), new DoubleTeam());  
 }  
}

public class Yanma extends Pokemon {  
 public Yanma(String name, int level) {  
 super(name, level);  
 super.setType(Type.*BUG*,Type.*FLYING*);  
 super.setStats(65,65,45,75,45,95);  
 super.setMove(new SteelWing(), new AirSlash(), new Rest());  
 }  
}

public class Yanmega extends Pokemon {  
 public Yanmega(String name, int level) {  
 super(name, level);  
 super.setType(Type.*BUG*,Type.*FLYING*);  
 super.setStats(86,76,86,116,56,95);  
 super.setMove(new SteelWing(), new AirSlash(), new Rest(), new Slash());  
 }  
}

**Program**

package lab2;  
import ru.ifmo.se.pokemon.\*;  
import mypokemons.\*;  
import mymoves.\*;  
  
public class Program {  
 public static void main(String[] args) {  
   
 Battle b = new Battle();

Rotom Rotom = new Rotom("Trululu", 1);  
 Nuzleaf Nuzleaf = new Nuzleaf("Trululu2", 2);  
 Seedot Seedot = new Seedot("Trululu3", 3);  
 Shiftry Shiftry = new Shiftry("Pumpum", 1);  
 Yanma Yanma = new Yanma("Pumpum2", 2);  
 Yanmega Yanmega = new Yanmega("Pumpum3", 3);  
 b.addFoe(Rotom);  
 b.addFoe(Nuzleaf);  
 b.addFoe(Seedot);  
 b.addAlly(Shiftry);  
 b.addAlly(Yanma);  
 b.addAlly(Yanmega);  
 b.go();  
 }  
}

# Результат работы

Shiftry Pumpum из команды желтых вступает в бой!

Rotom Trululu из команды полосатых вступает в бой!

Rotom Trululu does WillOWisp.

Shiftry Pumpum воспламеняется

Shiftry Pumpum промахивается

Rotom Trululu does WillOWisp.

Shiftry Pumpum does Swagger.

Rotom Trululu не замечает воздействие типа NORMAL

Rotom Trululu does ChargeBeam.

Shiftry Pumpum теряет 3 здоровья.

Shiftry Pumpum does LeafBlade.

Rotom Trululu теряет 6 здоровья.

Rotom Trululu does ConfuseRay.

Rotom Trululu восстанавливает 40 здоровья.

Shiftry Pumpum does Facade.

Rotom Trululu не замечает воздействие типа NORMAL

Rotom Trululu растерянно попадает по себе.

Rotom Trululu теряет 4 здоровья.

Shiftry Pumpum промахивается

Rotom Trululu does ConfuseRay.

Rotom Trululu восстанавливает 40 здоровья.

Shiftry Pumpum does LeafBlade.

Rotom Trululu теряет 7 здоровья.

Rotom Trululu does Thunderbolt.

Shiftry Pumpum теряет 4 здоровья.

Shiftry Pumpum does Facade.

Rotom Trululu не замечает воздействие типа NORMAL

Rotom Trululu does ChargeBeam.

Shiftry Pumpum теряет 3 здоровья.

Shiftry Pumpum does Swagger.

Rotom Trululu не замечает воздействие типа NORMAL

Rotom Trululu does WillOWisp.

Shiftry Pumpum does Facade.

Rotom Trululu не замечает воздействие типа NORMAL

Rotom Trululu does WillOWisp.

Shiftry Pumpum промахивается

Rotom Trululu does ChargeBeam.

Shiftry Pumpum теряет 3 здоровья.

Shiftry Pumpum теряет сознание.

Yanma Pumpum2 из команды желтых вступает в бой!

Yanma Pumpum2 does AirSlash.

Rotom Trululu теряет 4 здоровья.

Rotom Trululu does Thunderbolt.

Yanma Pumpum2 теряет 14 здоровья.

Yanma Pumpum2 промахивается

Yanma Pumpum2 промахивается

Rotom Trululu does Thunderbolt.

Yanma Pumpum2 теряет 18 здоровья.

Yanma Pumpum2 теряет сознание.

Yanmega Pumpum3 из команды желтых вступает в бой!

Yanmega Pumpum3 does SteelWing.

Rotom Trululu теряет 3 здоровья.

Rotom Trululu does Thunderbolt.

Yanmega Pumpum3 теряет 18 здоровья.

Yanmega Pumpum3 does AirSlash.

Rotom Trululu теряет 6 здоровья.

Rotom Trululu does WillOWisp.

Yanmega Pumpum3 воспламеняется

Yanmega Pumpum3 промахивается

Rotom Trululu does WillOWisp.

Yanmega Pumpum3 does Slash.

Rotom Trululu теряет 1 здоровья.

Rotom Trululu не замечает воздействие типа NORMAL

Rotom Trululu does ChargeBeam.

Yanmega Pumpum3 теряет 8 здоровья.

Yanmega Pumpum3 теряет сознание.

В команде желтых не осталось покемонов.

Команда полосатых побеждает в этом бою!

# Выводы

В процессе выполнения лабораторной работы я получила навыки использования

объектно-ориентированного подхода программирования при использовании языка Java.

Попробовала научиться работать с классами, конструкторами, полями и модификаторами доступа.