Университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №2

по программированию

Вариант № 1234455

Выполнил:

Степанов Артур

Группа: P3112

Приняла:

Сорокин Роман Борисович

???

?

г. Санкт-Петербург

2021 г.

# Текст задания:

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)

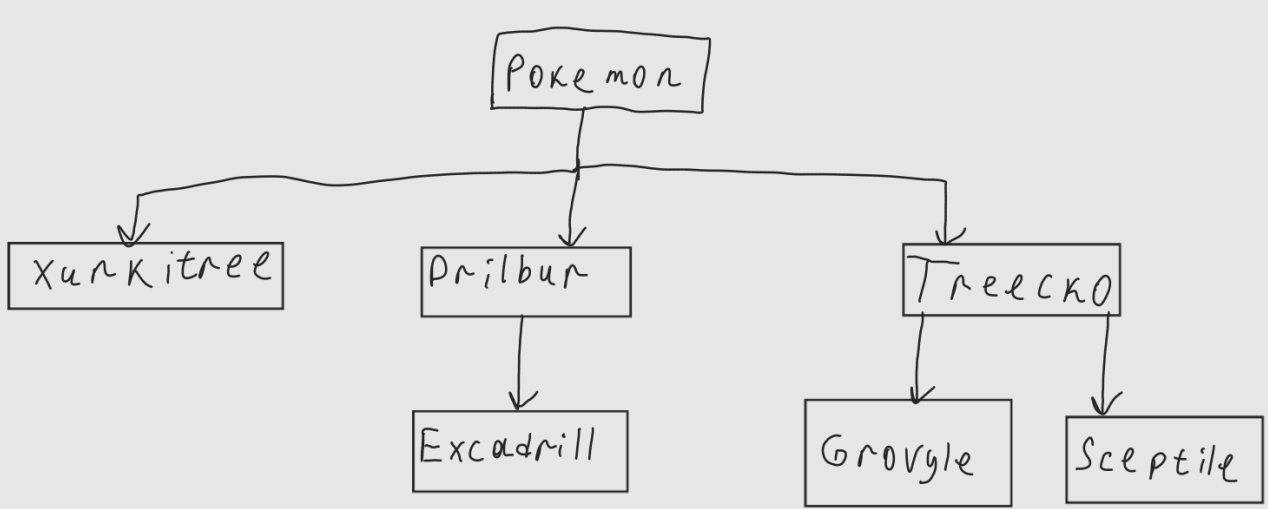
Классы покемонов должны наследоваться в соответствии с цепочкой эволюции покемонов. На основе базовых классов PhysicalMove, SpecialMove и StatusMove реализовать свои классы для заданных видов атак.

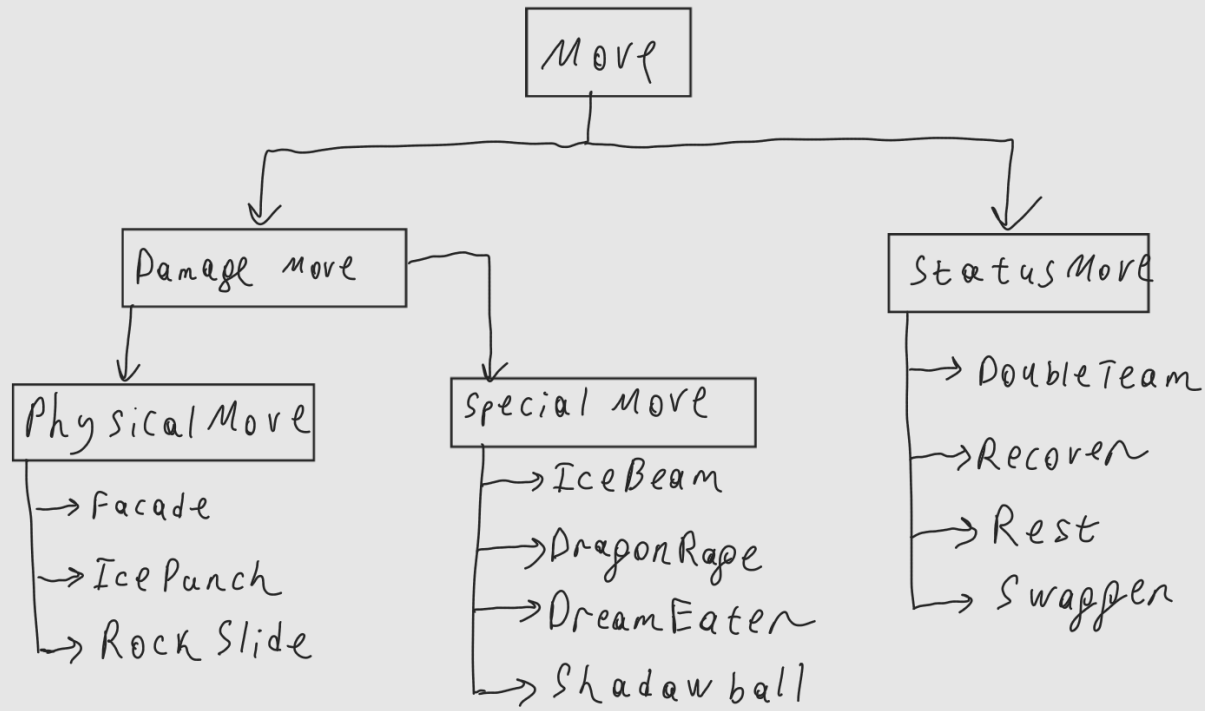
Атака должна иметь стандартные тип, силу (power) и точность (accuracy). Должны быть реализованы стандартные эффекты атаки. Назначить каждому виду покемонов атаки в соответствии с вариантом. Уровень покемона выбирается минимально необходимым для всех реализованных атак.

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



# Диаграмма классов реализованной объектной модели





# Исходный код программы

**package** com.company;

**import** **com.company.pokemons.\***;

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

**public** **class** **Main** {

**public** **static** **void** **main**(String[] args) {

Battle b = **new** Battle();

Pokemon p1 = **new** Xurkitree("ВМК", **1**);

Pokemon p2 = **new** Drilbur("МФТИ", **1**);

Pokemon p3 = **new** Excadrill("КТ", **1**);

Pokemon p4 = **new** Treecko("СППО", **1**);

Pokemon p5 = **new** Grovyle("ИСИТ", **1**);

Pokemon p6 = **new** Sceptile("ВШЭ", **1**);

b.addAlly(p1);

b.addAlly(p3);

b.addAlly(p5);

b.addFoe(p2);

b.addFoe(p4);

b.addFoe(p6);

b.go();

}

}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class DoubleTeam extends StatusMove {  
 public DoubleTeam(){  
 super(Type.*NORMAL*, 0, 0);  
 }  
 @Override  
 protected String describe(){  
 return "использует DoubleTeam";  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon satret){  
 satret.setMod(Stat.*EVASION*, 1);  
 }  
}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class DragonRage extends SpecialMove {  
 public DragonRage(){  
 super(Type.*DRAGON*, 0, 100);  
 }  
 @Override  
 protected String describe(){  
 return "использует DragonRage";  
 }  
  
 @Override  
 protected void applySelfEffects(Pokemon satret){  
 satret.setMod(Stat.*ATTACK*, 40);//поправь  
 }  
}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class DreamEater extends SpecialMove {  
 public DreamEater(){  
 super(Type.*PSYCHIC*, 100, 100);  
 }  
 @Override  
 protected String describe(){  
 return "использует DreamEater";  
 }  
  
 @Override  
 protected void applySelfEffects(Pokemon satret){  
 satret.setMod(Stat.*HP*, 50);//поправь  
 }  
}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Facade extends PhysicalMove {  
 public Facade() {  
 super(Type.*NORMAL*, 70, 100);  
 }  
  
 @Override  
 protected String describe() {  
 return "использует Facade";  
 }  
  
 @Override  
 protected void applyOppDamage(Pokemon satret, double damage) {  
 Status p\_stat = satret.getCondition();  
 if (p\_stat.equals(Status.*BURN*) || p\_stat.equals(Status.*POISON*) || p\_stat.equals(Status.*PARALYZE*)) {  
 super.applyOppDamage(satret, damage \* 2);  
 } else {  
 super.applyOppDamage(satret, damage);  
 }  
 }  
}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class IceBeam extends SpecialMove {  
 public IceBeam(){  
 super(Type.*ICE*, 90, 100);  
 }  
 @Override  
 protected String describe(){  
 return "использует IceBeam";  
 }  
  
 @Override  
 protected void applyOppDamage(Pokemon satret, double damage){  
 satret.setMod(Stat.*HP*, (int) Math.*round*(damage));  
 if (Math.*random*() <= 0.1) {  
 Effect.*freeze*(satret);  
 }  
 satret.setMod(Stat.*ATTACK*, 40);//поправь  
 }  
}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class IcePunch extends PhysicalMove {  
 public IcePunch(){  
 super(Type.*ICE*, 75, 100);  
 }  
 @Override  
 protected String describe(){  
 return "использует IcePunch";  
 }  
  
 @Override  
 protected void applyOppDamage(Pokemon satret, double damage){  
 satret.setMod(Stat.*HP*, (int) Math.*round*(damage));  
 if (Math.*random*() <= 0.1) {  
 Effect.*freeze*(satret);  
 }  
 }  
}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Recover extends StatusMove {  
 public Recover(){  
 super(Type.*NORMAL*, 0, 0);  
 }  
 @Override  
 protected String describe(){  
 return "использует Recover";  
 }  
  
 @Override  
 protected void applySelfEffects(Pokemon satret){  
 satret.setMod(Stat.*HP*, 50);//поправь  
 }  
}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Rest extends StatusMove {  
 public Rest(){  
 super(Type.*PSYCHIC*, 0, 0);  
 }  
 @Override  
 protected String describe(){  
 return "использует Recover";  
 }  
 @Override  
 protected void applySelfEffects(Pokemon p){  
 Effect e2 = new Effect().turns(2).condition(Status.*SLEEP*);  
 p.restore();  
 }  
}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class RockSlide extends PhysicalMove {  
 public RockSlide(){  
 super(Type.*ROCK*, 75, 90);  
 }  
 @Override  
 protected String describe(){  
 return "использует RockSlide";  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon satret){  
 if (Math.*random*() <= 0.3) {  
 Effect.*flinch*(satret);  
 }  
 }  
 }

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class ShadowBall extends SpecialMove {  
 public ShadowBall(){  
 super(Type.GHOST, 80, 100);  
 }  
 @Override  
 protected String describe(){  
 return "использует ShadowBall";  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon satret){  
 if (Math.random() <= 0.2) {  
 satret.setMod(Stat.SPECIAL\_ATTACK, -1);  
 }  
 }  
}

package com.company.moves;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Swagger extends StatusMove {  
 public Swagger(){  
 super(Type.*NORMAL*, 0, 85);  
 }  
 @Override  
 protected String describe(){  
 return "использует Swagger";  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon satret){  
 satret.setMod(Stat.*ATTACK*, 2);  
 Effect.*confuse*(satret);  
 }  
}

package com.company.pokemons;  
  
import com.company.moves.\*;  
import ru.ifmo.se.pokemon.\*;  
  
public class Drilbur extends Pokemon {  
 public Drilbur(String name, int level){  
 super(name, level);  
  
 setLevel(level <= 30 ? level : 30);  
  
 setType(Type.*GROUND*);  
  
 double hp = 60,  
 attack = 85,  
 defense = 40,  
 specialAttack = 30,  
 specialDefense = 45,  
 speed = 68;  
 setStats(hp, attack, defense, specialAttack, specialDefense, speed);  
  
 setMove(new DoubleTeam(), new Swagger(), new Rest());  
 }  
}

package com.company.pokemons;  
  
import com.company.moves.\*;  
import ru.ifmo.se.pokemon.\*;  
  
public class Excadrill extends Drilbur {  
 public Excadrill(String name, int level){  
 super(name, level);  
  
 setLevel(level <= 31 ? level : 31);  
  
 setType(Type.GROUND, Type.STEEL);  
  
 double hp = 110,  
 attack = 135,  
 defense = 60,  
 specialAttack = 50,  
 specialDefense = 65,  
 speed = 88;  
 setStats(hp, attack, defense, specialAttack, specialDefense, speed);  
  
 setMove(new DoubleTeam(), new Swagger(), new Rest(), new DreamEater());  
 }  
}

package com.company.pokemons;  
  
import com.company.moves.\*;  
import ru.ifmo.se.pokemon.\*;  
  
public class Grovyle extends Treecko {  
 public Grovyle(String name, int level){  
 super(name, level);  
  
 setLevel(level <= 16 ? level : 16);  
  
 setType(Type.*GRASS*);  
  
 double hp = 50,  
 attack = 65,  
 defense = 45,  
 specialAttack = 85,  
 specialDefense = 65,  
 speed = 95;  
 setStats(hp, attack, defense, specialAttack, specialDefense, speed);  
  
 setMove(new Facade(), new Rest(), new DragonRage());  
 }  
}

package com.company.pokemons;  
  
import com.company.moves.\*;  
import ru.ifmo.se.pokemon.\*;  
  
public class Sceptile extends Grovyle {  
 public Sceptile(String name, int level){  
 super(name, level);  
  
 setLevel(level <= 36 ? level : 36);  
  
 setType(Type.*GRASS*);  
  
 double hp = 70,  
 attack = 85,  
 defense = 65,  
 specialAttack = 105,  
 specialDefense = 85,  
 speed = 120;  
 setStats(hp, attack, defense, specialAttack, specialDefense, speed);  
  
 setMove(new Facade(), new Rest(), new DragonRage(), new IceBeam());  
 }  
}

package com.company.pokemons;  
  
import com.company.moves.\*;  
import ru.ifmo.se.pokemon.\*;  
  
public class Treecko extends Pokemon {  
 public Treecko(String name, int level){  
 super(name, level);  
  
 setLevel(level <= 15 ? level : 15);  
  
 setType(Type.*GRASS*);  
  
 double hp = 83,  
 attack = 89,  
 defense = 71,  
 specialAttack = 173,  
 specialDefense = 71,  
 speed = 83;  
 setStats(hp, attack, defense, specialAttack, specialDefense, speed);  
  
 setMove(new Facade(), new Rest());  
 }  
}

package com.company.pokemons;  
  
import com.company.moves.\*;  
import ru.ifmo.se.pokemon.\*;  
  
public class Xurkitree extends Pokemon {  
 public Xurkitree(String name, int level){  
 super(name, level);  
  
 setLevel(level <= 15 ? level : 15);  
  
 setType(Type.*ELECTRIC*);  
  
 double hp = 40,  
 attack = 45,  
 defense = 35,  
 specialAttack = 65,  
 specialDefense = 55,  
 speed = 70;  
 setStats(hp, attack, defense, specialAttack, specialDefense, speed);  
  
 setMove(new ShadowBall(), new IcePunch(), new Recover(), new RockSlide());  
 }  
}

# Результат работы программы

Xurkitree ВМК из команды желтых вступает в бой!

Drilbur МФТИ из команды белых вступает в бой!

Xurkitree ВМК использует ShadowBall.

Drilbur МФТИ теряет 4 здоровья.

Drilbur МФТИ уменьшает специальную атаку.

Drilbur МФТИ промахивается

Xurkitree ВМК использует RockSlide.

Drilbur МФТИ теряет 2 здоровья.

Drilbur МФТИ использует Swagger.

Xurkitree ВМК увеличивает атаку.

Xurkitree ВМК использует ShadowBall.

Drilbur МФТИ теряет 5 здоровья.

Drilbur МФТИ уменьшает специальную атаку.

Drilbur МФТИ использует Swagger.

Xurkitree ВМК увеличивает атаку.

Xurkitree ВМК растерянно попадает по себе.

Xurkitree ВМК теряет 4 здоровья.

Drilbur МФТИ промахивается

Xurkitree ВМК использует ShadowBall.

Drilbur МФТИ теряет 6 здоровья.

Drilbur МФТИ теряет сознание.

Treecko СППО из команды белых вступает в бой!

Treecko СППО промахивается

Xurkitree ВМК использует RockSlide.

Treecko СППО теряет 6 здоровья.

Treecko СППО использует Facade.

Xurkitree ВМК теряет 5 здоровья.

Xurkitree ВМК использует RockSlide.

Treecko СППО теряет 6 здоровья.

Treecko СППО использует Facade.

Xurkitree ВМК теряет 4 здоровья.

Xurkitree ВМК теряет сознание.

Excadrill КТ из команды желтых вступает в бой!

Excadrill КТ использует Swagger.

Treecko СППО увеличивает атаку.

Treecko СППО использует Facade.

Excadrill КТ теряет 2 здоровья.

Excadrill КТ использует Swagger.

Treecko СППО увеличивает атаку.

Treecko СППО растерянно попадает по себе.

Treecko СППО теряет 5 здоровья.

Treecko СППО теряет сознание.

Sceptile ВШЭ из команды белых вступает в бой!

Sceptile ВШЭ использует DragonRage.

Excadrill КТ теряет 2 здоровья.

Sceptile ВШЭ увеличивает атаку.

Excadrill КТ промахивается

Sceptile ВШЭ использует Facade.

Excadrill КТ теряет 4 здоровья.

Excadrill КТ промахивается

Sceptile ВШЭ использует DragonRage.

Excadrill КТ теряет 1 здоровья.

Sceptile ВШЭ увеличивает атаку.

Excadrill КТ использует DreamEater.

Sceptile ВШЭ теряет 6 здоровья.

Excadrill КТ теряет 50 здоровья.

Excadrill КТ теряет сознание.

Grovyle ИСИТ из команды желтых вступает в бой!

Sceptile ВШЭ использует Facade.

Grovyle ИСИТ теряет 8 здоровья.

Grovyle ИСИТ промахивается

Sceptile ВШЭ использует Facade.

Grovyle ИСИТ теряет 8 здоровья.

Grovyle ИСИТ теряет сознание.

В команде желтых не осталось покемонов.

Команда белых побеждает в этом бою!

# Вывод

Выполнив лабораторную работу, я ознакомился с основными объектно-ориентировочного программирования, работой с классами и объектами, сторонними библиотеками и подключением их к своей программе.