САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ

ИССЛЕДРОВАТЕЛЬСКИЙ

УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,

МЕХАНИКИ И ОПТИКИ

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.01 Программная инженерия

Дисциплина «Программирование»

**Отчет**

По лабораторной работе №2

Вариант 11110

Студент

*Абделсамад Марван*

*P3130*

Преподаватель

Чупанов Аликылыч

Санкт-Петербург, 2023 г.

Текст задания:

На основе базового класса написать свои классы для заданных видов покемонов. Каждый вид покемона должен иметь один или два типа и стандартные базовые характеристики:Pokemon

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

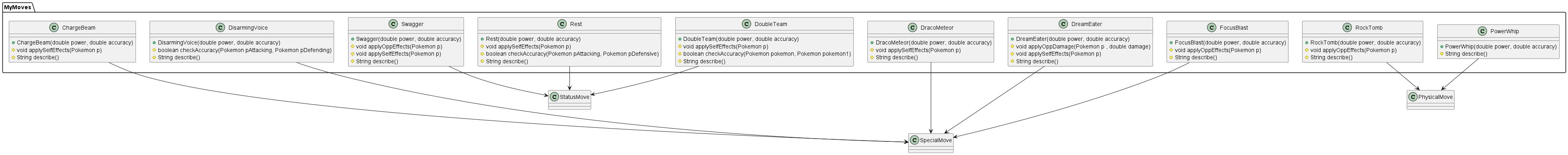
Классы покемонов должны наследоваться в соответствии с цепочкой эволюции покемонов. На основе базовых классов PhysicalMove, SpecialMove и StatusMove реализовать свои классы для заданных видов атак.

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.

A screenshot of a computer

Description automatically generatedUML-диаграмма классов



Код программы:

Атаки:

1. ChargeBeam

*package* MyMoves;  
  
*import* ru.ifmo.se.pokemon.\*;  
  
*public class* ChargeBeam *extends* SpecialMove {  
  
 *public* ChargeBeam(*double* power, *double* accuracy){  
 *super*(Type.ELECTRIC, power, accuracy);  
 }  
  
  
 *@Override  
 protected void* applySelfEffects(Pokemon p){  
 *super*.applySelfEffects(p);  
  
 Effect effect = *new* Effect().chance(0.7).stat(Stat.SPECIAL\_ATTACK, 1);  
 p.addEffect(effect);  
  
 }  
  
  
 *@Override  
 protected* String describe(){  
 String[] pieces = *this*.getClass().toString().split("\\.");  
 *return* "is doing " + pieces[pieces.length - 1];  
 }  
  
}

1. DisarmingVoice

*package* MyMoves;  
  
*import* ru.ifmo.se.pokemon.Pokemon;  
*import* ru.ifmo.se.pokemon.SpecialMove;  
*import* ru.ifmo.se.pokemon.Type;  
  
*public class* DisarmingVoice *extends* SpecialMove {  
  
 *public* DisarmingVoice(*double* power, *double* accuracy){  
 *super*(Type.FAIRY,power,accuracy);  
 }  
  
  
 *@Override  
 protected boolean* checkAccuracy(Pokemon pAttacking, Pokemon pDefending){  
 *return true*;  
 }  
  
 *@Override  
 protected* String describe(){  
 String[] pieces = *this*.getClass().toString().split("\\.");  
 *return* "is doing " +pieces[pieces.length - 1] ;  
 }  
  
}

1. DoubleTeam

*package* MyMoves;  
  
*import* ru.ifmo.se.pokemon.\*;  
  
*public class* DoubleTeam *extends* StatusMove {  
  
 *public* DoubleTeam(*double* power, *double* accuracy){  
 *super*(Type.NORMAL, power, accuracy);  
 }  
  
 *protected void* applySelfEffects(Pokemon p){  
  
 *super*.applySelfEffects(p);  
  
 Effect effect = *new* Effect().stat(Stat.EVASION, 1);  
 p.addEffect(effect);  
 }  
  
 *@Override  
 protected boolean* checkAccuracy(Pokemon pokemon, Pokemon pokemon1) {  
 *return super*.checkAccuracy(pokemon, pokemon1);  
 }  
  
 *@Override  
 protected* String describe(){  
 String[] pieces = *this*.getClass().toString().split("\\.");  
 *return* "does " + pieces[pieces.length-1] + " Now Evasiveness is raised by one stage";  
 }  
  
}

1. DracoMeteor

*package* MyMoves;  
  
  
*import* ru.ifmo.se.pokemon.\*;  
  
*public class* DracoMeteor *extends* SpecialMove {  
  
 *public* DracoMeteor(*double* power, *double* accuracy){  
 *super*(Type.DRAGON, power, accuracy);  
 }  
  
 *@Override  
 protected void* applySelfEffects(Pokemon p){  
  
 *super*.applySelfEffects(p);  
  
 Effect effect = *new* Effect().stat(Stat.SPECIAL\_ATTACK, -2);  
 p.addEffect(effect);  
 }

*@Override  
 protected* String describe(){  
 String[] pieces = *this*.getClass().toString().split("\\.");  
 *return* "does " + pieces[pieces.length-1] ;  
 }  
  
}

1. DreamEater

*package* MyMoves;  
  
*import* ru.ifmo.se.pokemon.\*;  
  
*public class* DreamEater *extends* SpecialMove {  
  
 *public* DreamEater(*double* power, *double* accuracy){  
 *super*(Type.PSYCHIC ,power, accuracy);  
 }  
  
 *@Override  
 protected void* applyOppDamage(Pokemon p , *double* damage){  
 *super*.applyOppDamage(p, damage);  
 *if* (p.getCondition() == Status.SLEEP){  
 *super*.applyOppDamage(p, damage);  
 }  
 }  
  
 *@Override  
 protected void* applySelfEffects(Pokemon p){  
 *super*.applySelfEffects(p);  
  
 *double* amountToRecoverBack = (p.getStat(Stat.HP) - p.getHP()) / 2;  
  
 p.setMod(Stat.HP, (*int*)amountToRecoverBack);  
 }  
  
 *@Override  
 protected* String describe(){  
 String[] pieces = *this*.getClass().toString().split("\\.");  
 *return* "does " + pieces[pieces.length-1];  
 }  
  
}

1. FocusBlast

*package* MyMoves;  
  
*import* ru.ifmo.se.pokemon.\*;  
  
*public class* FocusBlast *extends* SpecialMove {  
  
 *public* FocusBlast(*double* power, *double* accuracy){  
  
 *super*(Type.FIGHTING, power, accuracy);  
  
 }

*@Override  
 protected void* applyOppEffects(Pokemon p){  
 *super*.applyOppEffects(p);  
  
 Effect effect = *new* Effect().chance(0.1).stat(Stat.SPECIAL\_DEFENSE, -1);  
  
 p.addEffect(effect);  
 }  
  
 *@Override  
 protected* String describe(){  
 String[] pieces = *this*.getClass().toString().split("\\.");  
 *return* "is doing " + pieces[pieces.length - 1];  
 }  
  
}

1. PowerWhip

*package* MyMoves;  
  
*import* ru.ifmo.se.pokemon.PhysicalMove;  
*import* ru.ifmo.se.pokemon.Type;  
  
*public class* PowerWhip *extends* PhysicalMove {  
 *public* PowerWhip(*double* power, *double* accuracy){  
 *super*(Type.GRASS, power, accuracy);  
 }  
  
  
 *@Override  
 protected* String describe(){  
 String[] pieces = *this*.getClass().toString().split("\\.");  
 *return* "does " + pieces[pieces.length-1];  
 }  
}

1. Rest

*package* MyMoves;  
  
  
*import* ru.ifmo.se.pokemon.\*;  
  
*public class* Rest *extends* StatusMove {  
  
 *public* Rest(*double* power, *double* accuracy){  
 *super*(Type.PSYCHIC, power, accuracy);  
 }  
  
 *@Override  
 protected void* applySelfEffects(Pokemon p) {  
  
 *super*.applySelfEffects(p);  
  
 Effect effect = *new* Effect().turns(2).condition(Status.SLEEP);  
 p.restore();  
 p.addEffect(effect);  
  
 }  
  
 *@Override  
 protected boolean* checkAccuracy(Pokemon pAttacking, Pokemon pDefensive){  
 *return true*;  
 }  
  
 *@Override  
 protected* String describe(){  
 String[] pieces = *this*.getClass().toString().split("\\.");  
 *return* "does " + pieces[pieces.length-1] + " and restored";  
 }  
  
}

1. RockTomb

*package* MyMoves;  
  
*import* ru.ifmo.se.pokemon.\*;  
  
*public class* RockTomb *extends* PhysicalMove {  
  
 *public* RockTomb(*double* power, *double* accuracy) {  
 *super*(Type.ROCK, power, accuracy);  
 }  
  
 *@Override  
 protected void* applyOppEffects(Pokemon p){  
 *super*.applyOppEffects(p);  
  
 Effect effect = *new* Effect().stat(Stat.SPEED, -1);  
 p.addEffect(effect);  
  
 }  
  
  
  
 *@Override  
 protected* String describe() {  
 String[] pieces = *this*.getClass().toString().split("\\.");  
 *return* "does " + pieces[pieces.length - 1] + " Damage to target and lowering it's speed by one stage";  
  
 }  
}

1. Swagger
2. *package* MyMoves;  
     
   *import* ru.ifmo.se.pokemon.\*;  
     
   *public class* Swagger *extends* StatusMove {  
     
    *public* Swagger(*double* power, *double* accuracy){  
    *super*(Type.NORMAL ,power ,accuracy);  
    }  
     
     
    *@Override  
    protected void* applyOppEffects(Pokemon p){  
    *super*.applyOppEffects(p);  
    Effect.*confuse*(p);  
    }  
     
     
    *@Override  
    protected void* applySelfEffects(Pokemon p){  
    *super*.applySelfEffects(p);  
     
    Effect effect = *new* Effect().stat(Stat.ATTACK, 2);  
     
    p.addEffect(effect);  
    }  
     
     
    *@Override  
    protected* String describe(){  
    String[] pieces = *this*.getClass().toString().split("\\.");  
    *return* "is doing " + pieces[pieces.length - 1];  
    }  
     
   }

Покемоны:

1. Arceus

*package* MyPokemons;  
  
*import* MyMoves.DracoMeteor;  
*import* MyMoves.Rest;  
*import* MyMoves.DoubleTeam;  
*import* MyMoves.RockTomb;  
  
*import* ru.ifmo.se.pokemon.Pokemon;  
*import* ru.ifmo.se.pokemon.Type;  
  
*public class* Arceus *extends* Pokemon{  
  
 *public* Arceus(String name, *int* level){  
 *super*(name, level);  
  
 *super*.setType(Type.NORMAL);  
  
 *super*.setStats(120,120,120,120,120,120);  
  
  
 *super*.setMove(  
 *new* DracoMeteor(130,90),  
 *new* Rest(0,100),  
 *new* DoubleTeam(0,100),  
 *new* RockTomb( 60, 95)  
 );  
 }  
  
}

1. Igglybuff

*package* MyPokemons;  
  
*import* MyMoves.DreamEater;  
*import* MyMoves.Swagger;  
  
*import* ru.ifmo.se.pokemon.Pokemon;  
*import* ru.ifmo.se.pokemon.Type;  
  
*public class* Igglybuff *extends* Pokemon {  
  
 *public* Igglybuff(String name, *int* level){  
  
 *super*(name ,level);  
  
 *super*.setType(Type.NORMAL, Type.FAIRY);  
  
 *super*.setStats(90, 30,15, 40, 20, 15);  
  
 *super*.setMove(  
 *new* DreamEater(100,100),  
 *new* Swagger(0, 85)  
 );  
  
 }  
  
  
}

1. Jigglypuff

*package* MyPokemons;  
  
*import* MyMoves.DreamEater;  
*import* MyMoves.Swagger;  
*import* MyMoves.DisarmingVoice;  
  
  
*import* ru.ifmo.se.pokemon.Pokemon;  
*import* ru.ifmo.se.pokemon.Type;  
  
*public class* Jigglypuff *extends* Pokemon {  
  
 *public* Jigglypuff(String name, *int* level){  
  
 *super*(name, level);  
  
 *super*.setType(Type.NORMAL, Type.FAIRY);  
  
 *super*.setStats(115, 45, 20, 45, 25, 20);  
  
 *super*.setMove(  
 *new* DreamEater(100, 100),  
 *new* Swagger(0, 85),  
 *new* DisarmingVoice(40, 100)  
 );  
  
 }  
  
}

1. Lickilicky

*package* MyPokemons;  
  
*import* MyMoves.Rest;  
*import* MyMoves.PowerWhip;  
*import* MyMoves.DreamEater;  
*import* MyMoves.FocusBlast;  
  
*import* ru.ifmo.se.pokemon.Pokemon;  
*import* ru.ifmo.se.pokemon.Type;  
  
*public class* Lickilicky *extends* Pokemon {  
  
 *public* Lickilicky(String name, *int* level){  
  
 *super*(name, level);  
  
 *super*.setType(Type.NORMAL);  
  
 *super*.setStats(110, 85, 95, 80, 95, 50);  
  
 *super*.setMove(  
 *new* Rest(0, 100),  
 *new* PowerWhip(120, 85),  
 *new* DreamEater(100, 100),  
 *new* FocusBlast(120, 70)  
 );  
  
  
 }  
}

1. Lickitung

*package* MyPokemons;  
  
*import* MyMoves.Rest;  
*import* MyMoves.PowerWhip;  
*import* MyMoves.DreamEater;  
  
*import* ru.ifmo.se.pokemon.Pokemon;  
*import* ru.ifmo.se.pokemon.Type;  
  
*public class* Lickitung *extends* Pokemon {  
  
 *public* Lickitung(String name, *int* level){  
  
 *super*(name, level);  
  
 *super*.setType(Type.NORMAL);  
  
 *super*.setStats(90,55,75,60,75,30);  
  
 *super*.setMove(  
 *new* Rest(0,100),  
 *new* PowerWhip(120,85),  
 *new* DreamEater(100, 100)  
 );  
 }  
  
}

1. Wigglytuff

*package* MyPokemons;  
  
*import* MyMoves.Swagger;  
*import* MyMoves.DreamEater;  
*import* MyMoves.DisarmingVoice;  
*import* MyMoves.ChargeBeam;  
  
*import* ru.ifmo.se.pokemon.Pokemon;  
*import* ru.ifmo.se.pokemon.Type;  
  
*public class* Wigglytuff *extends* Pokemon {  
  
 *public* Wigglytuff(String name, *int* level){  
  
 *super*(name,level);  
  
 *super*.setType(Type.NORMAL, Type.FAIRY);  
  
 *super*.setStats(140, 70, 45, 85, 50,45);  
  
 *super*.setMove(  
 *new* Swagger(0, 85),  
 *new* DreamEater(100 , 100),  
 *new* DisarmingVoice(40, 100),  
 *new* ChargeBeam(50 , 90)  
 );  
  
 }  
  
  
}

*main.java*

*import* ru.ifmo.se.pokemon.Pokemon;  
*import* ru.ifmo.se.pokemon.Battle;  
*import* MyPokemons.\*;  
  
  
  
*public class* Main {  
  
 // https://pokemondb.net/pokedex/arceus  
 // https://pokemondb.net/pokedex/lickitung  
 // https://pokemondb.net/pokedex/lickilicky  
 // https://pokemondb.net/pokedex/igglybuff  
 // https://pokemondb.net/pokedex/jigglypuff  
 // https://pokemondb.net/pokedex/wigglytuff  
  
 *public static void* main(String[] args){  
 Battle b = *new* Battle();  
 Arceus arceus = *new* Arceus("ARCEUS", 1);  
 Lickitung lickitung = *new* Lickitung("LICKITUNG", 1);  
 Lickilicky lickilicky = *new* Lickilicky("LICKILICKY", 1);  
  
 Igglybuff igglybuff = *new* Igglybuff("IGGLYBUFF", 1);  
 Jigglypuff jigglypuff = *new* Jigglypuff("JIGGLYPUFF", 1);  
 Wigglytuff wigglytuff = *new* Wigglytuff("WIGGLYTUFF", 1);  
  
 b.addAlly(arceus);  
 b.addAlly(lickitung);  
 b.addAlly(lickilicky);  
  
 b.addFoe(igglybuff);  
 b.addFoe(jigglypuff);  
 b.addFoe(wigglytuff);  
  
 b.go();  
 }  
}

*Результат работы программы:*

Arceus ARCEUS from the team Black enters the battle!

Igglybuff IGGLYBUFF from the team Greren enters the battle!

Arceus ARCEUS does DoubleTeam Now Evasiveness is raised by one stage.

Igglybuff IGGLYBUFF is doing Swagger.

Arceus ARCEUS does DoubleTeam Now Evasiveness is raised by one stage.

Igglybuff IGGLYBUFF is doing Swagger.

Arceus ARCEUS does DracoMeteor.

Igglybuff IGGLYBUFF loses 1 hit points.

Igglybuff IGGLYBUFF isn't affected by DRAGON

Igglybuff IGGLYBUFF is doing Swagger.

Arceus ARCEUS does DoubleTeam Now Evasiveness is raised by one stage.

Igglybuff IGGLYBUFF does DreamEater.

Arceus ARCEUS loses 3 hit points.

Arceus ARCEUS does DoubleTeam Now Evasiveness is raised by one stage.

Igglybuff IGGLYBUFF is doing Swagger.

Arceus ARCEUS hits himself in confusion.

Arceus ARCEUS loses 4 hit points.

Igglybuff IGGLYBUFF does DreamEater.

Arceus ARCEUS loses 5 hit points.

Arceus ARCEUS does DracoMeteor.

Igglybuff IGGLYBUFF isn't affected by DRAGON

Igglybuff IGGLYBUFF does DreamEater.

Arceus ARCEUS loses 5 hit points.

Arceus ARCEUS faints.

Lickitung LICKITUNG from the team Black enters the battle!

Lickitung LICKITUNG does PowerWhip.

Igglybuff IGGLYBUFF loses 5 hit points.

Igglybuff IGGLYBUFF does DreamEater.

Lickitung LICKITUNG loses 4 hit points.

Igglybuff IGGLYBUFF loses 3 hit points.

Lickitung LICKITUNG does DreamEater.

Igglybuff IGGLYBUFF loses 4 hit points.

Lickitung LICKITUNG loses 2 hit points.

Igglybuff IGGLYBUFF faints.

Jigglypuff JIGGLYPUFF from the team Greren enters the battle!

Lickitung LICKITUNG does Rest and restored.

Lickitung LICKITUNG is sleeping

Jigglypuff JIGGLYPUFF does DreamEater.

Lickitung LICKITUNG loses 7 hit points.

Lickitung LICKITUNG loses 7 hit points.

Lickitung LICKITUNG faints.

Lickilicky LICKILICKY from the team Black enters the battle!

Lickilicky LICKILICKY does PowerWhip.

Jigglypuff JIGGLYPUFF loses 5 hit points.

Jigglypuff JIGGLYPUFF is doing Swagger.

Lickilicky LICKILICKY hits himself in confusion.

Lickilicky LICKILICKY loses 4 hit points.

Jigglypuff JIGGLYPUFF is doing DisarmingVoice.

Lickilicky LICKILICKY loses 7 hit points.

Lickilicky LICKILICKY is doing FocusBlast.

Jigglypuff JIGGLYPUFF loses 7 hit points.

Jigglypuff JIGGLYPUFF is doing DisarmingVoice.

Lickilicky LICKILICKY loses 7 hit points.

Lickilicky LICKILICKY faints.

Team Black loses its last Pokemon.

The team Greren wins the battle!

Выводы:

В процессе выполнения лабораторной работы я получил навыки использования

объектно-ориентированного подхода программирования при использовании языка Java.

Научился работать с классами, конструкторами, полями и модификаторами доступа.