Санкт-Петербургский

Национальный исследовательский университет

информационных технологий,

механики и оптики

(НИУ ИТМО)

Факультет программной инженерии и компьютерной техники

**Дисциплина** Программирование

**Отчёт** по Лабораторной работе № 2

**Вариант** 928656

**Преподаватель** Пашнин А.Д.

Выполнил

Студент группы P3113

Бритков А.А.

« 24 » октября 2020 г.

Санкт-Петербург 2020

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/).

Информацию о покемонах, цепочках эволюции и атаках можно найти на сайтах [http://poke-universe.ru](http://poke-universe.ru/), [http://pokemondb.net](http://pokemondb.net/),<http://veekun.com/dex/pokemon>

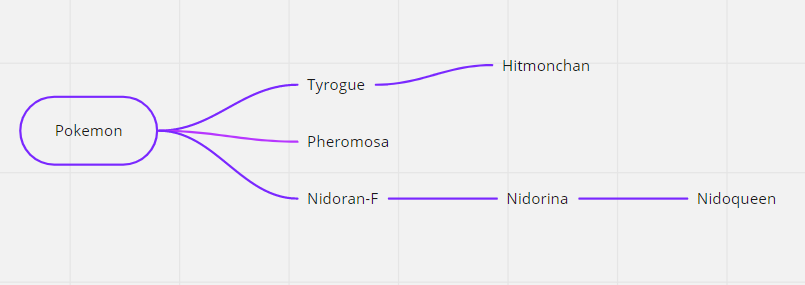
Ваши покемоны:

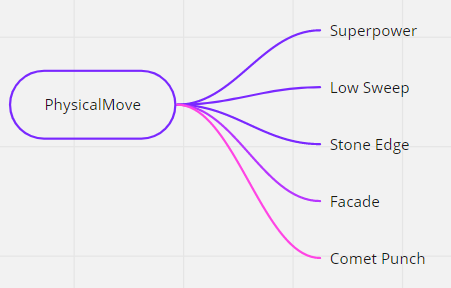
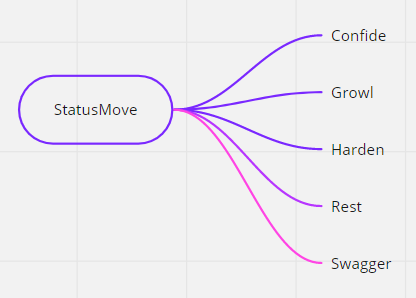


**Вопросы к защите лабораторной работы:**

1. Объектно-ориентированное программирование. Основные понятия: объекты, наследование, полиморфизм, инкапсуляция.
2. Понятие класса. Классы и объекты в Java.
3. Члены класса. Модификаторы доступа.
4. Создание и инициализация объектов. Вызов методов.
5. Области видимости переменных.
6. Модификаторы final и static.
7. Пакеты, инструкция import.

Диаграмма классов реализованной объектной модели:



Исходный код программы:

Main:

import com.pokemons.\*;  
import com.attacks.\*;  
import ru.ifmo.se.pokemon.\*;  
  
public class Main {  
  
 public static void main(String[] args) {  
 Battle b = new Battle();  
 Pokemon p1 = new Pheromosa("1", 10);  
 Pokemon p2 = new Nidorina("2", 10);  
 Pokemon p3 = new Hitmonchan("3", 10);  
 Pokemon p4 = new Nidoqueen("4", 10);  
 Pokemon p5 = new NidoranF("5", 10);  
 Pokemon p6 = new Tyrogue("6", 10);  
 b.addAlly(p1);  
 b.addAlly(p2);  
 b.addAlly(p3);  
 b.addFoe(p4);  
 b.addFoe(p5);  
 b.addFoe(p6);  
 b.go();  
 }  
}

Pheromosa:

package com.pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import com.attacks.\*;  
  
public class Pheromosa extends Pokemon{  
 public Pheromosa(String name, int level){  
 super(name, level);  
 setStats(71, 137, 37, 137, 37, 151);  
 setType(Type.*BUG*, Type.*FIGHTING*);  
 setMove(new Harden(), new Superpower(), new Confide(), new Rest());  
 }  
}

Tyroque:

package com.pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import com.attacks.\*;  
  
public class Tyrogue extends Pokemon{  
 public Tyrogue(String name, int lvl){  
 super(name, lvl);  
 setStats(35, 35, 35, 35, 35, 35);  
 setType(Type.*FIGHTING*);  
 setMove(new LowSweep(), new Confide(), new WorkUp());  
 }  
}

Hitmonchan:

package com.pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import com.attacks.\*;  
  
public class Hitmonchan extends Tyrogue {  
 public Hitmonchan(String name, int lvl){  
 super(name, lvl);  
 setType(Type.*FIGHTING*);  
 setStats(50, 105, 79, 35, 110, 76);  
 setMove(new LowSweep(), new Confide(), new WorkUp(), new CometPunch());  
 }  
}

Nidoran-F:

package com.pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import com.attacks.\*;  
  
public class NidoranF extends Pokemon{  
 public NidoranF(String name, int lvl){  
 super(name, lvl);  
 setStats(55, 47, 52, 40, 40, 41);  
 setType(Type.*POISON*);  
 setMove(new Facade(), new Swagger());  
 }  
}

Nidorina:

package com.pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import com.attacks.\*;  
  
public class Nidorina extends NidoranF {  
 public Nidorina(String name, int lvl) {  
 super(name, lvl);  
 setType(Type.*POISON*);  
 setStats(70, 62, 67, 55, 55, 56);  
 setMove(new Facade(), new Swagger(), new Growl());  
 }  
  
}

Nidoqueen:

package com.pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import com.attacks.\*;  
  
public class Nidoqueen extends Nidorina {  
 public Nidoqueen(String name, int lvl) {  
 super(name, lvl);  
 setType(Type.*POISON*);  
 setStats(90, 92, 87, 75, 85, 76);  
 setMove(new Facade(), new Swagger(), new Growl(), new StoneEdge());  
 }  
}

Attacks:

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class CometPunch extends PhysicalMove {  
 /\*protected com.attacks.CometPunch(Pokemon p){  
 super(Type.NORMAL, 18, 85);  
 }\*/  
 public CometPunch() {  
 super(Type.*NORMAL*, 18, 85);  
 double rnd = Math.*random*();  
 int x;  
 if (rnd < 3.0 / 8) {  
 x = 2;  
 }  
 else if (rnd < 6.0 / 8) {  
 x = 3;  
 }  
 else if (rnd < 7.0 / 8) {  
 x = 4;  
 }  
 else {  
 x = 5;  
 }  
 this.hits = x;  
 }  
  
 @Override  
 protected double calcCriticalHit(Pokemon atk, Pokemon def) {  
 if (Math.*random*() <= 0.0625) {  
 return 2.0D;  
 }  
 else {  
 return 1.0D;  
 }  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Confide extends StatusMove{  
 public Confide(){  
 super(Type.*NORMAL*, 0, 0);  
 }  
 @Override  
 protected boolean checkAccuracy(Pokemon atk, Pokemon def){  
 return true;  
 }  
 protected void applyOppEffects(Pokemon def){  
 def.setMod(Stat.*SPECIAL\_ATTACK*, -1);  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Facade extends PhysicalMove {  
 public Facade() {  
 super(Type.*NORMAL*, 70, 100);  
 }  
  
 @Override  
 protected double calcCriticalHit(Pokemon atk, Pokemon def){  
 Status stt = atk.getCondition();  
 if (stt == Status.*BURN* || stt == Status.*POISON* || stt == Status.*PARALYZE*){  
 return 2.0D;  
 }  
 else{  
 return 1.0D;  
 }  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Growl extends StatusMove {  
 public Growl() {  
 super(Type.*NORMAL*, 0, 100);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon def){  
 def.setMod(Stat.*ATTACK*, -1);  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Harden extends StatusMove {  
 public Harden(){  
 super(Type.*NORMAL*, 0, 0);  
 }  
 @Override  
 protected boolean checkAccuracy(Pokemon atk, Pokemon def){  
 return true;  
 }  
 protected void applySelfEffects(Pokemon atk){  
 atk.setMod(Stat.*DEFENSE*, 1);  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class LowSweep extends PhysicalMove {  
 public LowSweep(){  
 super(Type.*PSYCHIC*, 65, 100);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon def){  
 def.setMod(Stat.*SPEED*, -1);  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Rest extends StatusMove {  
 public Rest(){  
 super(Type.*PSYCHIC*, 0, 0);  
 }  
 @Override  
 protected boolean checkAccuracy(Pokemon atk, Pokemon def){  
 return true;  
 }  
  
 protected void applySelfEffects(Pokemon atk){  
 atk.setMod(Stat.*HP*, -((int)atk.getStat(Stat.*HP*) - (int)atk.getHP()));  
 Effect var1 = (new Effect()).condition(Status.*SLEEP*).attack(0.0D).turns(2);  
 atk.setCondition(var1);  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class StoneEdge extends PhysicalMove {  
 public StoneEdge() {  
 super(Type.*ROCK*, 100, 80);  
 }  
  
 @Override  
 protected double calcCriticalHit(Pokemon atk, Pokemon def) {  
 if (Math.*random*() <= 1.0 / 8) {  
 return 2.0D;  
 }  
 else {  
 return 1.0D;  
 }  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Superpower extends PhysicalMove {  
 public Superpower(){  
 super(Type.*NORMAL*, 120, 100);  
 }  
 @Override  
 protected void applySelfEffects(Pokemon atk){  
 atk.setMod(Stat.*ATTACK*, 1);  
 atk.setMod(Stat.*ACCURACY*, 1);  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Swagger extends StatusMove {  
 public Swagger() {  
 super(Type.*NORMAL*, 0, 85);  
 }  
 @Override  
 protected void applyOppEffects(Pokemon def){  
 def.setMod(Stat.*ATTACK*, 2);  
 Effect.*confuse*(def);  
 }  
}

package com.attacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class WorkUp extends StatusMove{  
 public WorkUp(){  
 super(Type.*NORMAL*, 0, 0);  
 }  
 @Override  
 protected boolean checkAccuracy(Pokemon atk, Pokemon def){  
 return true;  
 }  
 protected void applySelfEffects(Pokemon atk){  
 atk.setMod(Stat.*ATTACK*, 1);  
 atk.setMod(Stat.*SPECIAL\_ATTACK*, 1);  
 }  
}

Вывод программы:

Pheromosa 1 из команды зеленых вступает в бой!

Nidoqueen 4 из команды красных вступает в бой!

Pheromosa 1 атакует.

Nidoqueen 4 теряет 12 здоровья.

Pheromosa 1 увеличивает атаку.

Pheromosa 1 увеличивает точность.

Nidoqueen 4 атакует.

Pheromosa 1 теряет 8 здоровья.

Pheromosa 1 атакует.

Nidoqueen 4 уменьшает специальную атаку.

Nidoqueen 4 атакует.

Pheromosa 1 уменьшает атаку.

Pheromosa 1 атакует.

Pheromosa 1 восстанавливает 8 здоровья.

Pheromosa 1 засыпает

Nidoqueen 4 атакует.

Pheromosa 1 уменьшает атаку.

Nidoqueen 4 атакует.

Pheromosa 1 уменьшает атаку.

Pheromosa 1 атакует.

Критический удар!

Nidoqueen 4 восстанавливает 1 здоровья.

Pheromosa 1 увеличивает атаку.

Pheromosa 1 увеличивает точность.

Nidoqueen 4 атакует.

Pheromosa 1 теряет 8 здоровья.

Pheromosa 1 атакует.

Pheromosa 1 увеличивает защиту.

Nidoqueen 4 атакует.

Pheromosa 1 теряет 13 здоровья.

Pheromosa 1 атакует.

Pheromosa 1 увеличивает защиту.

Nidoqueen 4 атакует.

Pheromosa 1 увеличивает атаку.

Pheromosa 1 растерянно попадает по себе.

Pheromosa 1 теряет 8 здоровья.

Nidoqueen 4 атакует.

Pheromosa 1 увеличивает атаку.

Pheromosa 1 атакует.

Pheromosa 1 восстанавливает 29 здоровья.

Pheromosa 1 засыпает

Nidoqueen 4 атакует.

Pheromosa 1 увеличивает атаку.

Nidoqueen 4 атакует.

Pheromosa 1 уменьшает атаку.

Pheromosa 1 растерянно попадает по себе.

Pheromosa 1 теряет 13 здоровья.

Nidoqueen 4 атакует.

Pheromosa 1 увеличивает атаку.

Pheromosa 1 атакует.

Pheromosa 1 увеличивает защиту.

Nidoqueen 4 атакует.

Pheromosa 1 увеличивает атаку.

Pheromosa 1 атакует.

Pheromosa 1 увеличивает защиту.

Nidoqueen 4 атакует.

Pheromosa 1 уменьшает атаку.

Pheromosa 1 растерянно попадает по себе.

Pheromosa 1 теряет 15 здоровья.

Nidoqueen 4 атакует.

Pheromosa 1 уменьшает атаку.

Pheromosa 1 атакует.

Pheromosa 1 увеличивает защиту.

Nidoqueen 4 атакует.

Pheromosa 1 теряет 8 здоровья.

Pheromosa 1 теряет сознание.

Nidorina 2 из команды зеленых вступает в бой!

Nidoqueen 4 атакует.

Nidorina 2 уменьшает атаку.

Nidorina 2 атакует.

Nidoqueen 4 уменьшает атаку.

Nidoqueen 4 атакует.

Nidorina 2 теряет 12 здоровья.

Nidorina 2 атакует.

Nidoqueen 4 теряет 8 здоровья.

Nidoqueen 4 атакует.

Nidorina 2 увеличивает атаку.

Nidorina 2 атакует.

Nidoqueen 4 теряет 6 здоровья.

Nidoqueen 4 атакует.

Nidorina 2 увеличивает атаку.

Nidorina 2 атакует.

Nidoqueen 4 уменьшает атаку.

Nidoqueen 4 атакует.

Nidorina 2 уменьшает атаку.

Nidorina 2 растерянно попадает по себе.

Nidorina 2 теряет 8 здоровья.

Nidoqueen 4 атакует.

Nidorina 2 уменьшает атаку.

Nidorina 2 атакует.

Nidoqueen 4 уменьшает атаку.

Nidoqueen 4 атакует.

Nidorina 2 увеличивает атаку.

Nidorina 2 растерянно попадает по себе.

Nidorina 2 теряет 7 здоровья.

Nidoqueen 4 атакует.

Nidorina 2 увеличивает атаку.

Nidorina 2 растерянно попадает по себе.

Nidorina 2 теряет 7 здоровья.

Nidoqueen 4 атакует.

Nidorina 2 увеличивает атаку.

Nidorina 2 атакует.

Nidoqueen 4 уменьшает атаку.

Nidoqueen 4 атакует.

Nidorina 2 увеличивает атаку.

Nidorina 2 атакует.

Nidoqueen 4 увеличивает атаку.

Nidoqueen 4 атакует.

Nidorina 2 восстанавливает 1 здоровья.

Nidorina 2 атакует.

Nidoqueen 4 теряет 9 здоровья.

Nidoqueen 4 атакует.

Nidorina 2 увеличивает атаку.

Nidorina 2 растерянно попадает по себе.

Nidorina 2 теряет 8 здоровья.

Nidorina 2 теряет сознание.

Hitmonchan 3 из команды зеленых вступает в бой!

Hitmonchan 3 атакует.

Hitmonchan 3 теряет 5 здоровья.

Hitmonchan 3 атакует.

Hitmonchan 3 теряет 11 здоровья.

Hitmonchan 3 атакует.

Hitmonchan 3 теряет 5 здоровья.

Hitmonchan 3 атакует.

Hitmonchan 3 теряет 9 здоровья.

Hitmonchan 3 атакует.

Hitmonchan 3 теряет 17 здоровья.

Hitmonchan 3 уменьшает скорость.

Оба покемона теряют сознание.

В команде зеленых не осталось покемонов.

Команда красных побеждает в этом бою!

Вывод: Во время выполнения лабораторной работы разобрался с работой с ООП в Java и научился использовать внешние jar архивы в своих проектах.