**Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»**

**Факультет Программной Инженерии и Компьютерной Техники**

**Отчет по лабораторной работе №2:**

**Вариант 2153**

Студент:

Иванов Илья Андреевич

Р3117

Преподаватель:

Карасёва Мария Александровна

Санкт-Петербург, 2023

# 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.

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/).

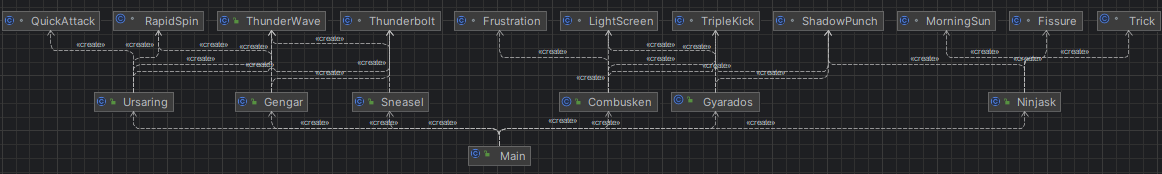
Информацию о покемонах, цепочках эволюции и атаках можно найти на сайтах [http://poke-universe.ru](http://poke-universe.ru/), [http://pokemondb.net](http://pokemondb.net/),<http://veekun.com/dex/pokemon>

Цель работы: на простом примере разобраться с основными концепциями ООП и научиться использовать их в программах.

Что надо сделать (краткое описание)

1. Ознакомиться с [документацией](https://se.ifmo.ru/~tony/doc/), обращая особое внимание на классы Pokemon и Move. При дальнейшем выполнении лабораторной работы читать документацию еще несколько раз.
2. Скачать файл Pokemon.jar. Его необходимо будет использовать как для компиляции, так и для запуска программы. Распаковывать его не надо! Нужно научиться подключать внешние jar-файлы к своей программе.
3. Написать минимально работающую программу и посмотреть как она работает.
4. Battle b = new Battle();
5. Pokemon p1 = new Pokemon("Чужой", 1);
6. Pokemon p2 = new Pokemon("Хищник", 1);
7. b.addAlly(p1);
8. b.addFoe(p2);
9. b.go();
10. Создать один из классов покемонов для своего варианта. Класс должен наследоваться от базового класса Pokemon. В конструкторе нужно будет задать типы покемона и его базовые характеристики. После этого попробуйте добавить покемона в сражение.
11. Создать один из классов атак для своего варианта (лучше всего начать с физической или специальной атаки). Класс должен наследоваться от класса PhysicalMove или SpecialMove. В конструкторе нужно будет задать тип атаки, ее силу и точность. После этого добавить атаку покемону и проверить ее действие в сражении. Не забудьте переопределить метод describe, чтобы выводилось нужное сообщение.
12. Если действие атаки отличается от стандартного, например, покемон не промахивается, либо атакующий покемон также получает повреждение, то в классе атаки нужно дополнительно переопределить соответствующие методы (см. документацию). При реализации атак, которые меняют статус покемона (наследники StatusMove), скорее всего придется разобраться с классом Effect. Он позволяет на один или несколько ходов изменить состояние покемона или модификатор его базовых характеристик.
13. Доделать все необходимые атаки и всех покемонов, распределить покемонов по командам, запустить сражение.

# Диаграмма классов реализованной объектной модели



# Коды программ

import ru.ifmo.se.pokemon.\***;**public class Main {  
 public static void main(String[] args) {  
 Battle b = new Battle()**;** Pokemon p1 = new Ninjask(""**, 1**)**;** Pokemon p2 = new Gengar(""**, 1**)**;** Pokemon p3 = new Sneasel(""**, 1**)**;** Pokemon p4 = new Gyarados(""**, 1**)**;** Pokemon p5 = new Combusken(""**, 1**)**;** Pokemon p6 = new Ursaring(""**, 1**)**;** b.addAlly(p1)**;** b.addAlly(p3)**;** b.addAlly(p5)**;** b.addFoe(p2)**;** b.addFoe(p4)**;** b.addFoe(p6)**;** b.go()**;** }  
}

## Покемоны

import ru.ifmo.se.pokemon.\***;**public class Ninjask extends Pokemon {  
 public Ninjask(String name**,** int level) {  
 super(name**,** level)**;** setType(Type.*BUG***,** Type.*FLYING*)**;** setStats(**61, 90, 45, 50, 50, 160**)**;** setMove(new Trick()**,** new Fissure()**,** new ShadowPunch()**,** new MorningSun())**;** }  
}

import ru.ifmo.se.pokemon.\***;**public class Gengar extends Pokemon {  
 public Gengar(String name**,** int level) {  
 super(name**,** level)**;** setType(Type.*GHOST***,** Type.*POISON*)**;** setStats(**60, 65, 60, 130, 75, 110**)**;** setMove(new ThunderWave()**,** new Thunderbolt()**,** new RapidSpin())**;** }  
}

import ru.ifmo.se.pokemon.\***;**public class Sneasel extends Pokemon {  
 public Sneasel(String name**,** int level) {  
 super(name**,** level)**;** setType(Type.*DARK***,** Type.*ICE*)**;** setStats(**55, 95, 55, 35, 75, 115**)**;** setMove(new ThunderWave()**,** new Thunderbolt())**;** }  
}

import ru.ifmo.se.pokemon.\***;**public class Gyarados extends Pokemon {  
 public Gyarados(String name**,** int level) {  
 super(name**,** level)**;** setType(Type.*WATER***,** Type.*FLYING*)**;** setStats(**95, 125, 79, 60, 100, 81**)**;** setMove(new TripleKick()**,** new LightScreen()**,** new ShadowPunch())**;** }  
}

import ru.ifmo.se.pokemon.\***;**public class Combusken extends Pokemon {  
 public Combusken(String name**,** int level) {  
 super(name**,** level)**;** setType(Type.*FIRE***,** Type.*FIGHTING*)**;** setStats(**60, 85, 60, 85, 60, 55**)**;** setMove(new TripleKick()**,** new LightScreen()**,** new ShadowPunch()**,** new Frustration())**;** }  
}

import ru.ifmo.se.pokemon.\***;**public class Ursaring extends Pokemon {  
 public Ursaring(String name**,** int level) {  
 super(name**,** level)**;** setType(Type.*NORMAL*)**;** setStats(**90, 130, 75, 75, 75, 55**)**;** setMove(new ThunderWave()**,** new Thunderbolt()**,** new RapidSpin()**,** new QuickAttack())**;** }  
}

## Атаки покемонов

import ru.ifmo.se.pokemon.\***;**class Trick extends StatusMove {  
 //Trick switches held items with the target  
 public Trick() {  
 super(Type.*PSYCHIC***, 0, 100**)**;** }  
  
 @Override  
 protected String describe() {  
 return "меняется предметами"**;** }  
}

import ru.ifmo.se.pokemon.\***;**import java.lang.Math**;**class Fissure extends PhysicalMove {  
 public Fissure() {  
 super(Type.*GROUND***, 0, 30**)**;** }  
  
 private boolean flag**;** @Override  
 protected void applyOppEffects(Pokemon p) {  
 if (Math.*round*(Math.*random*() \* **100**) <= **30**) {  
 Effect.*sleep*(p)**;** flag = true**;** }  
 }  
  
 @Override  
 protected String describe() {  
 if (flag) return "делает разлом"**;** else return "атакует"**;** }  
}

import ru.ifmo.se.pokemon.\***;**class ShadowPunch extends PhysicalMove {  
 /\*Shadow Punch deals damage and ignores changes to the Accuracy and Evasion stats\*/  
 public ShadowPunch() {  
 super(Type.*GHOST***, 60,** Double.*POSITIVE\_INFINITY*)**;** }  
  
 @Override  
 protected String describe() {  
 return "бьет из тени"**;** }  
}

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

class MorningSun extends StatusMove {  
 public MorningSun() {  
 super(Type.*NORMAL***, 0, 0**)**;** }  
  
 //User recovers HP.  
 @Override  
 protected void applySelfEffects(Pokemon p) {  
 p.restore()**;** }  
  
 @Override  
 protected String describe() {  
 return "восстанавливает HP и просыпается Солнце"**;** }  
  
  
}

import ru.ifmo.se.pokemon.\***;**public class ThunderWave extends StatusMove {  
 public ThunderWave() {  
 super(Type.*ELECTRIC***, 0, 100**)**;** }  
  
 private boolean flag**;** @Override  
 protected void applyOppEffects(Pokemon p) {  
 //Thunder Wave paralyzes the opponent. Paralyzed Pokémon have a 25% chance of not being able to attack, and their Speed is decreased by 50%  
 if (Math.*round*(Math.*random*() \* **100**) >= **25**) {  
 Effect.*paralyze*(p)**;** p.setMod(Stat.*SPEED***,** -(int) (**0.5** \* p.getStat(Stat.*SPEED*)))**;** flag = true**;** }  
 }  
  
 @Override  
 protected String describe() {  
 if (flag) return "парализует громовой волной"**;** else return "атакует"**;** }  
}

import ru.ifmo.se.pokemon.\***;**class Thunderbolt extends SpecialMove {  
 public Thunderbolt() {  
 super(Type.*ELECTRIC***, 90, 100**)**;** }  
  
 //Thunderbolt deals damage and has a 10% chance of paralyzing the target.  
 private boolean flag**;** @Override  
 protected void applyOppEffects(Pokemon p) {  
 if (Math.*round*(Math.*random*() \* **100**) <= **10**) {  
 Effect.*paralyze*(p)**;** flag = true**;** }  
 }  
  
 @Override  
 protected String describe() {  
 if (flag) return "бьет молнией"**;** else return "атакует"**;** }  
}

import ru.ifmo.se.pokemon.\***;**class RapidSpin extends PhysicalMove {  
 //Rapid Spin deals damage, raises the user's Speed by one stage  
 protected RapidSpin() {  
 super(Type.*NORMAL***, 50, 100**)**;** }  
  
 @Override  
 protected void applySelfEffects(Pokemon p) {  
 p.setMod(Stat.*SPEED***,** +**1**)**;** }  
  
 @Override  
 protected String describe() {  
 return "крутится как юла"**;** }  
  
}

import ru.ifmo.se.pokemon.\***;**class TripleKick extends PhysicalMove {  
 //Triple Kick deals damage and will strike three times  
 public TripleKick() {  
 super(Type.*FIGHTING***, 20, 90,** *getStruggleMove*().getPriority()**, 3**)**;** }  
  
 @Override  
 protected String describe() {  
 return "использует тройной удар ногой"**;** }  
}

import ru.ifmo.se.pokemon.\***;**class LightScreen extends StatusMove {  
 //Light Screen reduces damage from Special attacks by 50%, for 5 turns. Its effects apply to all Pokémon on the user's side of the field.  
 public LightScreen() {  
 super(Type.*PSYCHIC***, 0, 0**)**;** }  
  
 @Override  
 protected void applyOppEffects(Pokemon p) {  
 if (Math.*round*(Math.*random*() \* **100**) <= **30**) {  
 p.addEffect((new Effect()).stat(Stat.*SPECIAL\_ATTACK***,** (int) (**0.5** \* p.getStat(Stat.*SPECIAL\_ATTACK*))).turns(**5**))**;** }  
 }  
  
 @Override  
 protected String describe() {  
 return "использует Световой экран"**;** }  
}

import ru.ifmo.se.pokemon.\***;**class Frustration extends PhysicalMove {  
 public Frustration() {  
 //The power of Frustration is higher when the Pokémon likes its trainer less  
 super(Type.*NORMAL***,** (int) ((**255** - (int) (Math.*random*() \* **101** + **1.1**)) / **2.5**)**, 100**)**;** }  
  
 @Override  
 protected String describe() {  
 return "разочаровывается"**;** }  
}

import ru.ifmo.se.pokemon.\***;**class QuickAttack extends PhysicalMove {  
 //Quick Attack deals damage and has a priority of +1.  
 public QuickAttack() {  
 super(Type.*NORMAL***, 40, 100,** *getStruggleMove*().getPriority() + **1, 1**)**;** }  
  
 @Override  
 protected String describe() {  
 return "использует быструю атаку"**;** }  
}

# Результат работы программы

Ninjask из команды полосатых вступает в бой!

Gengar из команды синих вступает в бой!

Ninjask бьет из тени.

Gengar теряет 11 здоровья.

Gengar крутится как юла.

Ninjask теряет 5 здоровья.

Ninjask атакует.

Gengar теряет 5 здоровья.

Gengar засыпает

Gengar теряет сознание.

Gyarados из команды синих вступает в бой!

Ninjask бьет из тени.

Gyarados теряет 5 здоровья.

Gyarados использует тройной удар ногой.

Ninjask теряет 1 здоровья.

Gyarados использует тройной удар ногой.

Ninjask теряет 1 здоровья.

Gyarados использует тройной удар ногой.

Ninjask теряет 1 здоровья.

Ninjask делает разлом.

Gyarados теряет 1 здоровья.

Gyarados не замечает воздействие типа GROUND

Gyarados бьет из тени.

Ninjask теряет 5 здоровья.

Ninjask теряет сознание.

Sneasel из команды полосатых вступает в бой!

Sneasel атакует.

Gyarados теряет 18 здоровья.

Gyarados теряет сознание.

Ursaring из команды синих вступает в бой!

Ursaring использует быструю атаку.

Sneasel теряет 6 здоровья.

Sneasel атакует.

Ursaring теряет 4 здоровья.

Ursaring использует быструю атаку.

Sneasel теряет 6 здоровья.

Sneasel атакует.

Ursaring парализован

Ursaring уменьшает скорость.

Ursaring использует быструю атаку.

Sneasel теряет 5 здоровья.

Sneasel теряет сознание.

Combusken из команды полосатых вступает в бой!

Combusken промахивается

Ursaring использует быструю атаку.

Combusken теряет 4 здоровья.

Combusken использует тройной удар ногой.

Ursaring теряет 12 здоровья.

Combusken использует тройной удар ногой.

Ursaring теряет 9 здоровья.

Combusken использует тройной удар ногой.

Ursaring теряет 11 здоровья.

Ursaring теряет сознание.

В команде синих не осталось покемонов.

Команда полосатых побеждает в этом бою!

# Вывод

Я научился основам ООП языка программирования Java и научился их применять на практике.