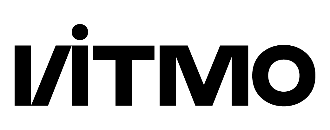
Национальный исследовательский университет ИТМО



Вариант 23789 Лабораторная работа №1

# По дисциплине «Программирование»

**Выполнила:**

Студент группы R3135

Возжаева Д.А.

# Преподаватель:

Иманзаде Ф.Р.

Санкт-Петербург 2022 г.

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.

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>

#### **Комментарии**

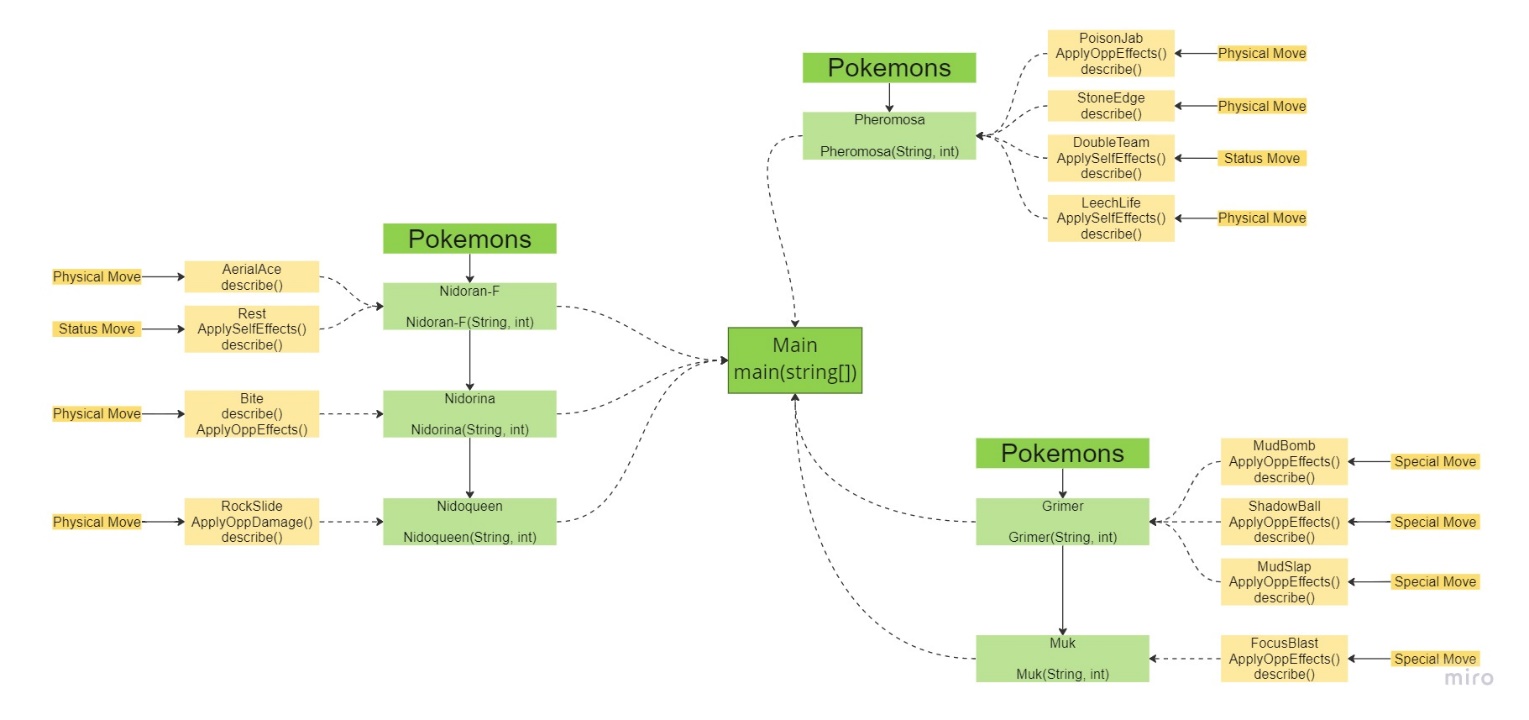
Цель работы: на простом примере разобраться с основными концепциями ООП и научиться использовать их в программах.

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:



Исходный код программы:

AerialAce.java:

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class AerialAce extends PhysicalMove {  
 public AerialAce(){  
 super(Type.FLYING, 60.0, 1.0);  
 }  
 protected String describe() {  
  
 return "использует Aerial Ace";  
 }  
}

Bite.java:

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Bite extends PhysicalMove {  
 public Bite(){  
 super(Type.*DARK*, 60.0, 1.0);  
 }  
 protected void applyOppDamage(Pokemon pokemon){  
 if (Math.*random*() <= 0.3) {  
 Effect.*flinch*(pokemon);  
 }  
 }  
 protected String describe() {  
  
 return "использует Attacs.Bite";  
 }  
}

DoubleTeam.java

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class DoubleTeam extends StatusMove {  
 public DoubleTeam() {  
 super(Type.*NORMAL*, 0.0, 1.0);  
 }  
  
 protected void applySelfEffects(Pokemon pokemon) {  
 pokemon.setMod(Stat.*EVASION*, 1);  
 }  
  
 protected String describe() {  
 return "использует Attacs.DoubleTeam";  
 }  
  
  
}

FocusBlast.java

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class FocusBlast extends SpecialMove {  
  
 public FocusBlast(){  
 super(Type.*FIGHTING*, 120.0, 0.7);  
 }  
 public void applyOppEffects(Pokemon pokemon) {  
  
 if (Math.*random*() <= 0.1) pokemon.setMod(Stat.*SPECIAL\_DEFENSE*, -1);  
 }  
 protected String describe() {  
  
 return "использует Focus Blast";  
 }  
}

LeechLife.java:

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class LeechLife extends PhysicalMove{  
 public LeechLife(){  
 super(Type.*BUG*, 80.0, 1.0);  
 }  
  
 public void applySelfEffects(Pokemon pokemon) {  
 pokemon.setMod(Stat.*HP*, (int) (calcBaseDamage(pokemon, pokemon) \* (-0.5)));  
 }  
  
 protected String describe() {  
 return "использует Attacs.LeechLife";  
 }  
}

MudBomb.java

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class MudBomb extends SpecialMove {  
  
 public MudBomb(){  
 super(Type.*GROUND*, 65.0, 0.8);  
 }  
 public void applyOppEffects(Pokemon pokemon) {  
  
 if (Math.*random*() <= 0.3) pokemon.setMod(Stat.*ACCURACY*, -1);  
 }  
 protected String describe() {  
  
 return "использует Mud Bomb";  
 }  
}

MudSlap.java

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class MudSlap extends SpecialMove {  
 public MudSlap(){  
 super(Type.*GROUND*, 20.0, 1.0);  
 }  
 public void applyOppEffects(Pokemon pokemon) {  
  
 pokemon.setMod(Stat.*ACCURACY*, -1);  
 }  
 protected String describe() {  
  
 return "использует Mud-Slap";  
 }  
}

PoisonJab.java

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class PoisonJab extends PhysicalMove {  
 public PoisonJab() {  
  
 super(Type.*POISON*, 80, 1);  
 }  
  
 public void applyOppEffects(Pokemon pokemon) {  
  
 if (Math.*random*() <= 0.3) Effect.*poison*(pokemon);  
 }  
  
 protected String describe() {  
  
 return "использует Attacs.PoisonJab";  
 }  
}

Rest.java

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Rest extends StatusMove {  
 protected double HP;  
 public Rest(double hp){  
 super(Type.*PSYCHIC*, 0.0, 1.0);  
 HP = hp;  
 }  
 protected void applySelfEffects(Pokemon pokemon){  
 pokemon.addEffect(new Effect().condition(Status.*SLEEP*).turns(2));  
 pokemon.setStats(HP, pokemon.getStat(Stat.*ATTACK*), pokemon.getStat(Stat.*DEFENSE*), pokemon.getStat(Stat.*SPECIAL\_ATTACK*), pokemon.getStat(Stat.*SPECIAL\_DEFENSE*), pokemon.getStat(Stat.*SPEED*));  
 }  
 protected String describe() {  
  
 return "использует Attacs.Rest";  
 }  
}

RockSlide.java:

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class RockSlide extends PhysicalMove {  
 public RockSlide(){  
 super(Type.*ROCK*, 75.0, 0.9);  
 }  
 protected void applyOppDamage(Pokemon pokemon){  
 if (Math.*random*() <= 0.3) {  
 Effect.*flinch*(pokemon);  
 }  
 }  
 protected String describe() {  
  
 return "использует Rock SLide";  
 }  
}

ShadowBall.java:

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class ShadowBall extends SpecialMove {  
 public ShadowBall(){  
 super(Type.*GHOST*, 80.0, 1.0);  
 }  
 public void applyOppEffects(Pokemon pokemon) {  
  
 if (Math.*random*() <= 0.2) pokemon.setMod(Stat.*SPECIAL\_DEFENSE*, -1);  
 }  
 protected String describe() {  
  
 return "использует Shadow Ball";  
 }  
}

StoneEdge.java:

package Attacs;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class StoneEdge extends PhysicalMove{  
 public StoneEdge(){  
 super(Type.*ROCK*, 100, 0.8);  
  
 }  
  
 protected String describe() {  
 return "использует Attacs.StoneEdge";  
 }  
}

Grimer.java:

package Pokemons;  
  
import Attacs.MudBomb;  
import Attacs.MudSlap;  
import Attacs.ShadowBall;  
import ru.ifmo.se.pokemon.\*;  
  
public class Grimer extends Pokemon {  
 public Grimer(String name,int lvl) throws WrongLevelException {  
 super(name, lvl);  
 if (lvl < 0) {  
 throw new WrongLevelException(name, lvl);  
 }  
 setStats(80.0, 80.0, 50.0, 40.0, 50.0, 25.0);  
 setType(Type.*POISON*);  
 setMove(new MudBomb(), new ShadowBall(), new MudSlap());  
 }  
}

Muk.java:

package Pokemons;  
  
import Attacs.FocusBlast;  
  
public class Muk extends Grimer{  
 public Muk(String name, int lvl) throws WrongLevelException {  
 super(name, lvl);  
 if (lvl < 0) {  
 throw new WrongLevelException(name, lvl);  
 }  
 setStats(105.0, 105.0, 75.0, 65.0, 100.0, 50.0);  
 addMove(new FocusBlast());  
 }  
  
}

Nidoqueen.java:

package Pokemons;  
  
import Attacs.RockSlide;  
import ru.ifmo.se.pokemon.\*;  
public class Nidoqueen extends Nidorina{  
  
 public Nidoqueen(String name, int lvl) throws WrongLevelException {  
 super(name, lvl);  
 if (lvl < 0) {  
 throw new WrongLevelException(name, lvl);  
 }  
 addType(Type.*GROUND*);  
 setStats(90.0, 92.0, 87.0, 75.0, 85.0, 76.0);  
 addMove(new RockSlide());  
 }  
}

NidoranF.java:

package Pokemons;  
  
import Attacs.AerialAce;  
import Attacs.Rest;  
import ru.ifmo.se.pokemon.\*;  
  
public class NidoranF extends Pokemon {  
 public NidoranF(String name, int lvl) throws WrongLevelException {  
 super(name, lvl);  
 if (lvl < 0) {  
 throw new WrongLevelException(name, lvl);  
 }  
 setType(Type.*POISON*);  
 setStats(55.0, 47.0, 52.0, 40.0, 40.0, 41.0);  
 setMove(new AerialAce(), new Rest(55.0));  
 }  
}

Nidorina.java:

package Pokemons;  
  
import Attacs.Bite;  
  
public class Nidorina extends NidoranF{  
 public Nidorina(String name, int lvl) throws WrongLevelException {  
 super(name, lvl);  
 if (lvl < 0) {  
 throw new WrongLevelException(name, lvl);  
 }  
 setStats(70.0, 62.0, 67.0, 55.0, 55.0, 56.0);  
 addMove(new Bite());  
 }  
}

Pheromosa.java:

package Pokemons;  
  
import Attacs.DoubleTeam;  
import Attacs.LeechLife;  
import Attacs.PoisonJab;  
import Attacs.StoneEdge;  
import ru.ifmo.se.pokemon.\*;  
  
public class Pheromosa extends Pokemon {  
 public Pheromosa(String name,int lvl) throws WrongLevelException {  
 super(name, lvl);  
 if (lvl < 0) {  
 throw new WrongLevelException(name, lvl);  
 }  
 setStats(71.0, 137.0, 37.0, 137.0, 37.0, 151.0);  
 setType(Type.*BUG*, Type.*FIGHTING*);  
 setMove(new PoisonJab(), new StoneEdge(), new DoubleTeam(), new LeechLife());  
 }  
 }

WrongLevelException.java

package Pokemons;  
  
public class WrongLevelException extends Exception {  
 public WrongLevelException(String name, int lvl) {  
 super("Некорректный уровень покемона " + name + "(" + String.*valueOf*(lvl) + ")");  
 }  
}

Main.java

import Pokemons.\*;  
import ru.ifmo.se.pokemon.\*;  
  
public class Main {  
 public static void main(String[] args) throws WrongLevelException {  
 Battle b = new Battle();  
 Pokemon p1 = new Pheromosa("Алена", 1);  
 b.addFoe(p1);  
 Pokemon p2 = new Grimer("Алиса", 1);  
 b.addFoe(p2);  
 Pokemon p3 = new Muk("Вероника", 1);  
 b.addAlly(p3);  
 Pokemon p4 = new NidoranF("Василиса", 1);  
 b.addAlly(p4);  
 Pokemon p5 = new Nidorina("Валерия", 1);  
 b.addAlly(p5);  
 Pokemon p6 = new Nidoqueen("Аделина", 1);  
 b.addFoe(p6);  
 b.go();  
 }  
}

Результат работы программы:

Muk Вероника из команды белых вступает в бой!

Pheromosa Алена из команды зеленых вступает в бой!

Pheromosa Алена использует Attacs.DoubleTeam.

Pheromosa Алена увеличивает уклоняемость.

Muk Вероника использует Mud Bomb.

Pheromosa Алена теряет 2 здоровья.

Pheromosa Алена использует Attacs.PoisonJab.

Muk Вероника теряет 2 здоровья.

Muk Вероника использует Focus Blast.

Pheromosa Алена теряет 3 здоровья.

Pheromosa Алена уменьшает специальную защиту.

Pheromosa Алена использует Attacs.DoubleTeam.

Pheromosa Алена увеличивает уклоняемость.

Muk Вероника использует Shadow Ball.

Pheromosa Алена теряет 6 здоровья.

Pheromosa Алена использует Attacs.PoisonJab.

Muk Вероника теряет 3 здоровья.

Muk Вероника использует Mud-Slap.

Pheromosa Алена теряет 2 здоровья.

Pheromosa Алена уменьшает точность.

Pheromosa Алена теряет сознание.

Grimer Алиса из команды зеленых вступает в бой!

Muk Вероника промахивается

Grimer Алиса использует Shadow Ball.

Muk Вероника теряет 5 здоровья.

Muk Вероника использует Mud-Slap.

Grimer Алиса теряет 9 здоровья.

Grimer Алиса уменьшает точность.

Grimer Алиса использует Shadow Ball.

Muk Вероника теряет 4 здоровья.

Muk Вероника теряет сознание.

NidoranF Василиса из команды белых вступает в бой!

NidoranF Василиса использует Attacs.Rest.

NidoranF Василиса засыпает

Grimer Алиса использует Mud-Slap.

NidoranF Василиса теряет 9 здоровья.

NidoranF Василиса уменьшает точность.

Grimer Алиса использует Mud-Slap.

NidoranF Василиса теряет 5 здоровья.

NidoranF Василиса уменьшает точность.

NidoranF Василиса теряет сознание.

Nidorina Валерия из команды белых вступает в бой!

Nidorina Валерия использует Attacs.Bite.

Grimer Алиса теряет 3 здоровья.

Grimer Алиса использует Mud-Slap.

Nidorina Валерия теряет 6 здоровья.

Nidorina Валерия уменьшает точность.

Nidorina Валерия использует Aerial Ace.

Grimer Алиса теряет 5 здоровья.

Grimer Алиса теряет сознание.

Nidoqueen Аделина из команды зеленых вступает в бой!

Nidoqueen Аделина использует Rock SLide.

Nidorina Валерия теряет 5 здоровья.

Nidorina Валерия использует Attacs.Rest.

Nidorina Валерия засыпает

Nidoqueen Аделина использует Rock SLide.

Nidorina Валерия теряет 4 здоровья.

Nidorina Валерия теряет сознание.

В команде белых не осталось покемонов.

Команда зеленых побеждает в этом бою!

Вывод:

Во время выполнения данной лабораторной работы я познакомилась с объектно-ориентированным программированием, а также его принципами: Абстракцией, Инкапсуляцией, Наследованием и Полиморфизмом. В ходе работы я научилась создавать классы-наследники, добавлять им новые методы и переопределять родительские. Также я научилась работать с документацией и Jar архивами, научилась строить UML диаграммы.

Дополнительное задание:

1. Форматированный вывод (без пустых строк)

Причина: метод attack(Pokemon var1) родительского класса Pokemon

public final boolean attack(Pokemon var1) {  
 try {  
 Thread.*sleep*(1000L);  
 } catch (InterruptedException var3) {  
 }  
  
 this.preparedMove.attack(this, var1);  
 if (this.isAlive() && var1.isAlive()) {  
 System.*out*.println();  
 return false;  
 } else {  
 if (!this.isAlive() && !var1.isAlive()) {  
 System.*out*.println(Messages.*get*("bothFaint"));  
 } else {  
 System.*out*.println((this.isAlive() ? var1 : this) + " " + Messages.*get*("faint"));  
 }  
  
 return true;  
 }  
}

Невозможно устранить, т.к. метод является final, соответственно его невозможно переопределить.

1. Исключение в случае отрицательного значения уровня покемона

Базовый класс Pokemon не вызывает исключения, т.к. в метод присваивает значению единицу, в случае отрицательного уровня:

public final void setLevel(int var1) {  
 if (var1 < 1) {  
 var1 = 1;  
 }  
  
 if (var1 > 100) {  
 var1 = 100;  
 }  
  
 this.level = var1;  
}

Исключение, возникающее при попытки задать отрицательный уровень, является наследником базового класса Exception:

package Pokemons;  
  
public class WrongLevelException extends Exception {  
 public WrongLevelException(String name, int lvl) {  
 super("Некорректный уровень покемона " + name + "(" + String.*valueOf*(lvl) + ")");  
 }  
}

Пример применения в классах:

package Pokemons;  
  
import Attacs.AerialAce;  
import Attacs.Rest;  
import ru.ifmo.se.pokemon.\*;  
  
public class NidoranF extends Pokemon {  
 public NidoranF(String name, int lvl) throws WrongLevelException {  
 super(name, lvl);  
 if (lvl < 0) {  
 throw new WrongLevelException(name, lvl);  
 }  
 setType(Type.*POISON*);  
 setStats(55.0, 47.0, 52.0, 40.0, 40.0, 41.0);  
 setMove(new AerialAce(), new Rest(55.0));  
 }  
}

Пример вывода:

Изображение выглядит как текст

Автоматически созданное описание