Национальный исследовательский университет информационных технологий, механики и оптики

Факультет программной инженерии и компьютерной техники

Программирование

Лабораторная работа №2

Работу выполнил:

Бавыкин Роман

Группа: Р3110

Вариант 10039

Санкт-Петербург

2020 г.

1. **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 (обновлен 9.10.2018, исправлен баг с добавлением атак и кодировкой). Документация в формате javadoc - здесь.

Информацию о покемонах, цепочках эволюции и атаках можно найти на сайтах http://poke-universe.ru, 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, 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.

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

Pokemon p1 = **new** Pokemon("Чужой", **1**);

Pokemon p2 = **new** Pokemon("Хищник", **1**);

b.addAlly(p1);

b.addFoe(p2);

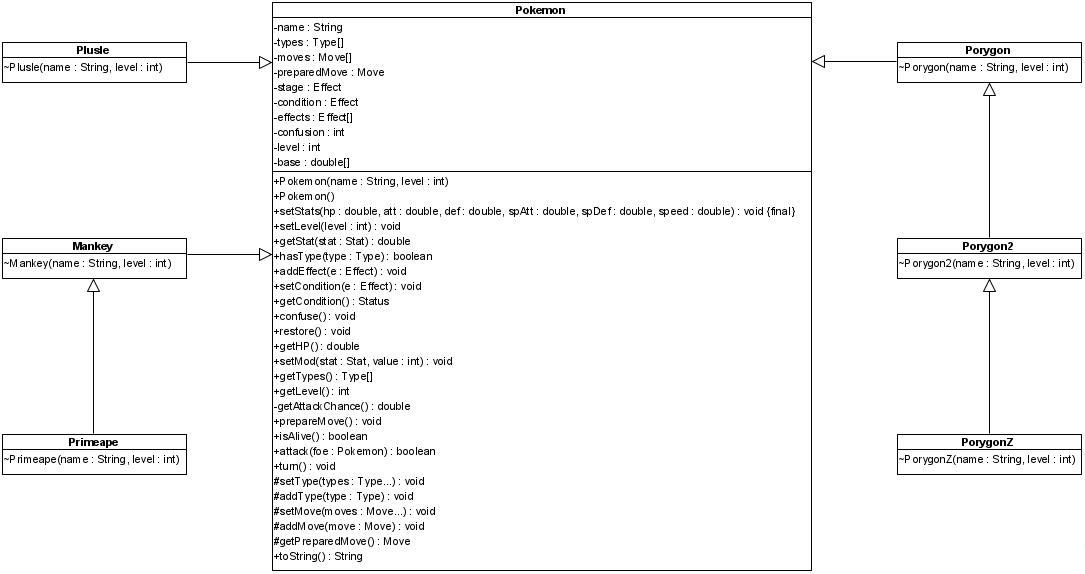
b.go();

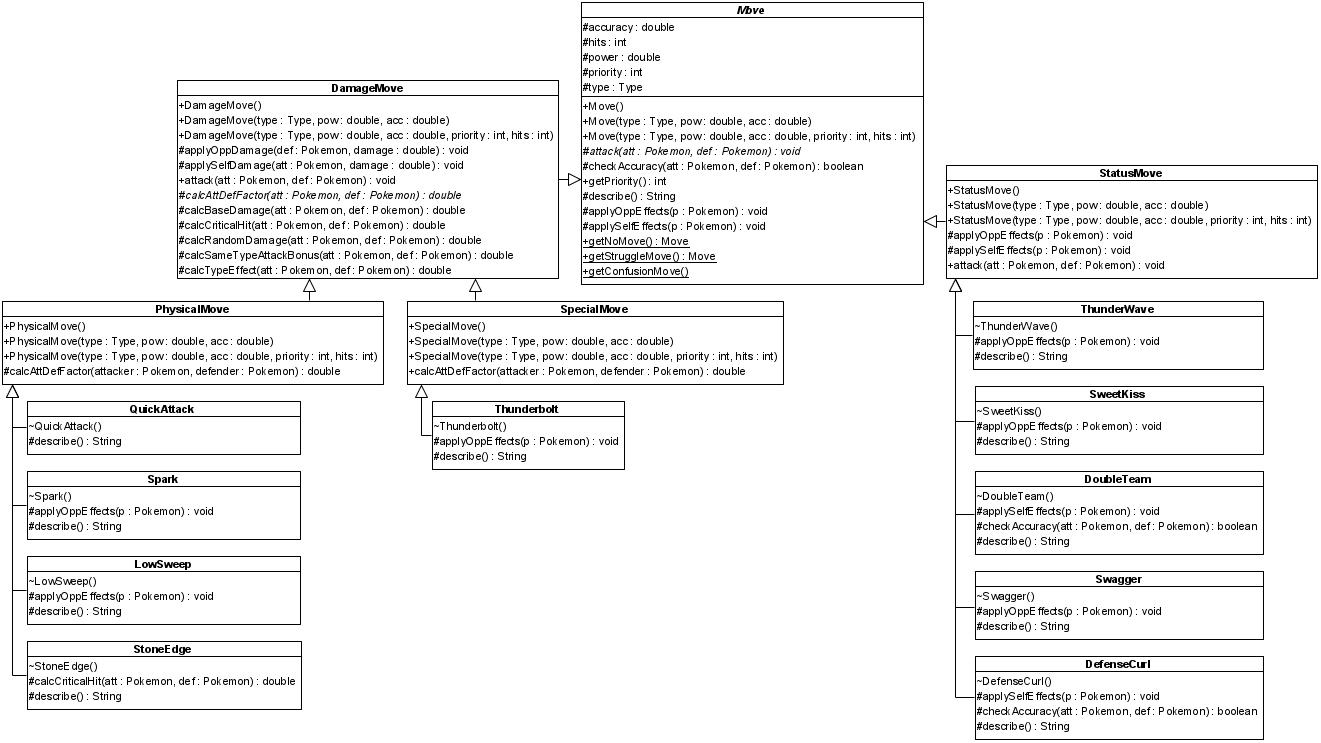
1. Создать один из классов покемонов для своего варианта. Класс должен наследоваться от базового класса Pokemon. В конструкторе нужно будет задать типы покемона и его базовые характеристики. После этого попробуйте добавить покемона в сражение.
2. Создать один из классов атак для своего варианта (лучше 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.
3. 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.
4. Доделать все необходимые атаки и всех покемонов, распределить покемонов по командам, запустить сражение.

Ваши покемоны:



1. **Диаграмма классов реализованной объектной модели:**

****

****

1. **Исходный код программы:**

Main.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | **package** lab2;  **import** **ru.ifmo.se.pokemon.\***;  **public** **class** **Main** {  **public** **static** **void** **main**(String[] args) {  Battle b = **new** Battle();  Pokemon a1 = **new** Plusle("Java", **10**);  Pokemon a2 = **new** Mankey("C", **10**);  Pokemon a3 = **new** Primeape("C++", **10**);  Pokemon f1 = **new** Porygon("Python", **10**);  Pokemon f2 = **new** Porygon2("Python 2", **10**);  Pokemon f3 = **new** PorygonZ("Python 3", **10**);  b.addAlly(a1);  b.addAlly(a2);  b.addAlly(a3);  b.addFoe(f1);  b.addFoe(f2);  b.addFoe(f3);  b.go();  }  } |

Plusle.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57 | **package** lab2;  **import** **ru.ifmo.se.pokemon.\***;  **public** **class** **Plusle** **extends** Pokemon {  Plusle(String name, **int** level){  **super**(name, level);  addType(Type.ELECTRIC);  setStats(**60**, **50**, **40**, **85**, **75**, **95**);  setMove(**new** QuickAttack(), **new** ThunderWave(), **new** SweetKiss(), **new** Spark());  }  }  **class** **QuickAttack** **extends** PhysicalMove{  QuickAttack(){  **super**(Type.NORMAL, **40**, **1.0**);  }  **protected** String **describe**(){  **return** "использует Quick Attack";  }  }  **class** **ThunderWave** **extends** StatusMove{  ThunderWave(){  **super**(Type.ELECTRIC, **0**, **0.9**);  }  **protected** **void** **applyOppEffects**(Pokemon p){  Effect.paralyze(p);  }  **protected** String **describe**(){  **return** "использует Thunder Wave";  }  }  **class** **SweetKiss** **extends** StatusMove{  SweetKiss(){  **super**(Type.FAIRY, **0**, **0.75**);  }  **protected** **void** **applyOppEffects**(Pokemon p){  Effect.confuse(p);  }  **protected** String **describe**(){  **return** "использует Sweet Kiss";  }  }  **class** **Spark** **extends** PhysicalMove{  Spark(){  **super**(Type.ELECTRIC, **65**, **1.0**);  }  **protected** **void** **applyOppEffects**(Pokemon p){  **if**(Math.random() < **0.3**) Effect.paralyze(p);  }  **protected** String **describe**(){  **return** "использует Spark";  }  } |

Mankey.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51 | **package** lab2;  **import** **ru.ifmo.se.pokemon.\***;  **public** **class** **Mankey** **extends** Pokemon {  Mankey(String name, **int** level){  **super**(name, level);  addType(Type.FIGHTING);  setStats(**40**, **80**, **35**, **35**, **45**, **70**);  setMove(**new** Thunderbolt(), **new** DoubleTeam(), **new** LowSweep());  }  }  **class** **Thunderbolt** **extends** SpecialMove{  Thunderbolt(){  **super**(Type.ELECTRIC, **90**, **1.0**);  }  **protected** **void** **applyOppEffects**(Pokemon p){  **if**(Math.random() < **0.1**) Effect.paralyze(p);  }  **protected** String **describe**(){  **return** "использует Thunderbolt";  }  }  **class** **DoubleTeam** **extends** StatusMove{  DoubleTeam(){  **super**(Type.NORMAL, **0**, **0**);  }  **protected** **void** **applySelfEffects**(Pokemon p){  p.setMod(Stat.EVASION, **1**);  }  **protected** **boolean** **checkAccuracy**(Pokemon att, Pokemon def) {  **return** **true**;  }  **protected** String **describe**(){  **return** "использует Double Team";  }  }  **class** **LowSweep** **extends** PhysicalMove{  LowSweep(){  **super**(Type.FIGHTING, **65**, **1.0**);  }  **protected** **void** **applyOppEffects**(Pokemon p){  p.setMod(Stat.SPEED, -**1**);  }  **protected** String **describe**(){  **return** "использует Low Sweep";  }  } |

Primeape.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | **package** lab2;  **import** **ru.ifmo.se.pokemon.\***;  **public** **class** **Primeape** **extends** Mankey{  Primeape(String name, **int** level) {  **super**(name, level);  setStats(**65**, **105**, **60**, **60**, **70**, **95**);  addMove(**new** StoneEdge());  }  }  **class** **StoneEdge** **extends** PhysicalMove{  StoneEdge(){  **super**(Type.ROCK, **100**, **0.8**);  }  **protected** **double** **calcCriticalHit**(Pokemon att, Pokemon def){  **if** (Math.random() < **1.0**/**8.0**){  **return** **2**;  }  **else**{  **return** **1**;  }  }  **protected** String **describe**(){  **return** "использует Stone Edge";  }  } |

Porygon.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | **package** lab2;  **import** **ru.ifmo.se.pokemon.\***;  **public** **class** **Porygon** **extends** Pokemon {  Porygon(String name, **int** level){  **super**(name, level);  addType(Type.NORMAL);  setStats(**65**, **60**, **70**, **85**, **75**, **40**);  setMove(**new** DoubleTeam(), **new** Swagger());  }  }  **class** **Swagger** **extends** StatusMove{  Swagger(){  **super**(Type.NORMAL, **0**, **85**);  }  **protected** **void** **applyOppEffects**(Pokemon p){  Effect.confuse(p);  p.setMod(Stat.ATTACK, **2**);  }  **protected** String **describe**(){  **return** "использует Swagger";  }  } |

Porygon2.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | **package** lab2;  **import** **ru.ifmo.se.pokemon.\***;  **public** **class** **Porygon2** **extends** Porygon{  Porygon2(String name, **int** level) {  **super**(name, level);  setStats(**85**, **80**, **90**, **105**, **95**, **60**);  addMove(**new** DefenseCurl());  }  }  **class** **DefenseCurl** **extends** StatusMove{  DefenseCurl(){  **super**(Type.NORMAL, **0**, **0**);  }  **protected** **void** **applySelfEffects**(Pokemon p){  p.setMod(Stat.DEFENSE, **1**);  }  **protected** **boolean** **checkAccuracy**(Pokemon att, Pokemon def) {  **return** **true**;  }  **protected** String **describe**(){  **return** "использует Defense Curl";  }  } |

PorygonZ.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | **package** lab2;  **public** **class** **PorygonZ** **extends** Porygon2{  PorygonZ(String name, **int** level) {  **super**(name, level);  setStats(**85**, **80**, **70**, **135**, **75**, **90**);  addMove(**new** ThunderWave());  }  } |

1. **Результат работы программы:**

Plusle Java из команды желтых вступает в бой!

Porygon Python из команды синих вступает в бой!

Plusle Java использует Spark.

Porygon Python теряет 7 здоровья.

Porygon Python использует Double Team.

Porygon Python увеличивает уклоняемость.

Plusle Java использует Thunder Wave.

Porygon Python парализован

Porygon Python использует Double Team.

Porygon Python увеличивает уклоняемость.

Porygon Python использует Swagger.

Plusle Java увеличивает атаку.

Plusle Java использует Thunder Wave.

Plusle Java промахивается

Plusle Java промахивается

Plusle Java промахивается

Porygon Python использует Swagger.

Plusle Java увеличивает атаку.

Plusle Java промахивается

Porygon Python использует Double Team.

Porygon Python увеличивает уклоняемость.

Plusle Java использует Quick Attack.

Porygon Python теряет 6 здоровья.

Porygon Python использует Swagger.

Plusle Java увеличивает атаку.

Plusle Java промахивается

Porygon Python использует Double Team.

Porygon Python увеличивает уклоняемость.

Plusle Java растерянно попадает по себе.

Plusle Java теряет 10 здоровья.

Porygon Python использует Swagger.

Plusle Java увеличивает атаку.

Plusle Java использует Spark.

Porygon Python теряет 18 здоровья.

Porygon Python использует Double Team.

Porygon Python увеличивает уклоняемость.

Plusle Java промахивается

Porygon Python использует Swagger.

Plusle Java увеличивает атаку.

Plusle Java промахивается

Plusle Java промахивается

Porygon Python использует Swagger.

Plusle Java увеличивает атаку.

Plusle Java растерянно попадает по себе.

Plusle Java теряет 10 здоровья.

Porygon Python использует Double Team.

Porygon Python увеличивает уклоняемость.

Plusle Java промахивается

Porygon Python использует Swagger.

Plusle Java увеличивает атаку.

Plusle Java растерянно попадает по себе.

Plusle Java теряет 6 здоровья.

Porygon Python использует Double Team.

Porygon Python увеличивает уклоняемость.

Plusle Java промахивается

Porygon Python использует Swagger.

Plusle Java увеличивает атаку.

Plusle Java растерянно попадает по себе.

Plusle Java теряет 8 здоровья.

Plusle Java теряет сознание.

Mankey C из команды желтых вступает в бой!

Porygon Python использует Double Team.

Porygon Python увеличивает уклоняемость.

Mankey C промахивается

Porygon Python использует Swagger.

Mankey C увеличивает атаку.

Mankey C использует Low Sweep.

Porygon Python теряет 25 здоровья.

Porygon Python уменьшает скорость.

Porygon Python теряет сознание.

Porygon2 Python 2 из команды синих вступает в бой!

Mankey C использует Low Sweep.

Porygon2 Python 2 теряет 29 здоровья.

Porygon2 Python 2 уменьшает скорость.

Porygon2 Python 2 использует Double Team.

Porygon2 Python 2 увеличивает уклоняемость.

Porygon2 Python 2 использует Defense Curl.

Porygon2 Python 2 увеличивает защиту.

Mankey C использует Thunderbolt.

Porygon2 Python 2 теряет 5 здоровья.

Porygon2 Python 2 использует Swagger.

Mankey C увеличивает атаку.

Mankey C промахивается

Porygon2 Python 2 использует Defense Curl.

Porygon2 Python 2 увеличивает защиту.

Mankey C использует Low Sweep.

Porygon2 Python 2 теряет 32 здоровья.

Porygon2 Python 2 уменьшает скорость.

Porygon2 Python 2 теряет сознание.

PorygonZ Python 3 из команды синих вступает в бой!

PorygonZ Python 3 использует Swagger.

Mankey C увеличивает атаку.

Mankey C использует Low Sweep.

PorygonZ Python 3 теряет 64 здоровья.

PorygonZ Python 3 уменьшает скорость.

PorygonZ Python 3 теряет сознание.

В команде синих не осталось покемонов.

Команда желтых побеждает в этом бою!

**Вывод:** во время выполнения лабораторной работы изучил основные принципы ООП, научился создавать классы и объекты и работать с ними.