Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №1

Вариант 13

Выполнил:

Бармичев Григорий Андреевич

Группа P3110

Проверил:

Мустафаева Айнур Вугар Кызы

Преподаватель практик

**Содержание**

[Задание 3](#_Toc181905401)

[Комментарии 3](#_Toc181905402)

[Ваши покемоны: 5](#_Toc181905403)

[UML-диаграмма классов 5](#_Toc181905404)

[Исходный код программы 5](#_Toc181905405)

[Move 5](#_Toc181905406)

[**Rock Tomb** 5](#_Toc181905407)

[**Aerial Ace** 6](#_Toc181905408)

[**Draco Meteor** 6](#_Toc181905409)

[**Dream Eater** 6](#_Toc181905410)

[**Fire Blast** 6](#_Toc181905411)

[**Ice Beam** 7](#_Toc181905412)

[**Night Shade** 7](#_Toc181905413)

[**Overheat** 7](#_Toc181905414)

[**Sludge Bomb** 8](#_Toc181905415)

[**Dragon Dance** 8](#_Toc181905416)

[**Sand Attack** 8](#_Toc181905417)

[**Swagger** 9](#_Toc181905418)

[Pokémon 9](#_Toc181905419)

[**Chandelure** 9](#_Toc181905420)

[**Lampent** 9](#_Toc181905421)

[**Latios** 10](#_Toc181905422)

[**Litwick** 10](#_Toc181905423)

[**Palossand** 10](#_Toc181905424)

[**Sandygast** 11](#_Toc181905425)

[Main 11](#_Toc181905426)

[Результат работы программы 12](#_Toc181905427)

[Вывод 13](#_Toc181905428)

# 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. Все разработанные классы, не имеющие наследников, должны быть реализованы таким образом, чтобы от них нельзя было наследоваться.

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.

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:

Изображение выглядит как текст, снимок экрана, мультфильм

Автоматически созданное описание

# UML-диаграмма классов

Изображение выглядит как снимок экрана, текст, прямоугольный, дизайн

Автоматически созданное описание

# Исходный код программы

## Move

### **Rock Tomb**

|  |
| --- |
| **package move.physical; import ru.ifmo.se.pokemon.\*; public class RockTomb extends PhysicalMove {  public RockTomb(){  super(Type.*ROCK*,60.0,95.0);  }  protected void applyOppEffects(Pokemon p2) {  p2.setMod(Stat.*SPEED*, -1);  }  protected String describe() {  return "использует Rock Tomb";  } }** |

### **Aerial Ace**

|  |
| --- |
| **package move.physical; import ru.ifmo.se.pokemon.\*; public class AerialAce extends SpecialMove {  public AerialAce() {  super(Type.*FLYING*, 60.0, 99999.0);  }  protected String describe() {  return "использует Aerial Ace";  } }** |

### **Draco Meteor**

|  |
| --- |
| **package move.special; import ru.ifmo.se.pokemon.\*; public class DracoMeteor extends SpecialMove {  public DracoMeteor() {  super(Type.*DRAGON*, 130.0, 90.0);  }  public void applySelfEffects(Pokemon p1) {  p1.setMod(Stat.*SPECIAL\_ATTACK*, -2);  }  protected String describe() {  return "использует Draco Meteor";  } }** |

### **Dream Eater**

|  |
| --- |
| **package move.special; import ru.ifmo.se.pokemon.\*; public class DreamEater extends SpecialMove {  public DreamEater() {  super(Type.*PSYCHIC*, 100.0, 100.0);  }  public void applySelfEffects(Pokemon p1,Pokemon p2) {  p1.setMod(Stat.*HP*, -(int)((p2.getStat(Stat.*HP*)-p2.getHP())/2));  }  protected String describe() {  return "использует Dream Eater";  } }** |

### **Fire Blast**

|  |
| --- |
| **package move.special; import ru.ifmo.se.pokemon.\*; public class FireBlast extends SpecialMove{  public FireBlast(){  super(Type.*FIRE*, 110.0,85.0);  }  private boolean flag;  public void applyOppEffects(Pokemon p2) {  if (Math.*random*() <= 0.3) {  flag = true;  Effect.*burn*(p2);  }  }  protected String describe(){  if(flag) return "использует Fire Blast и поджигает";  else return "использует Fire Blast";  } }** |

### **Ice Beam**

|  |
| --- |
| **package move.special; import ru.ifmo.se.pokemon.\*; public class IceBeam extends SpecialMove{  public IceBeam(){  super(Type.*ICE*, 90.0,100.0);  }  private boolean flag;  public void applyOppEffects(Pokemon p2) {  if (Math.*random*() <= 0.1) {  flag = true;  Effect.*freeze*(p2);  }  }  protected String describe(){  if(flag) return "использует Ice Beam и замораживает";  else return "использует Ice Beam";  } }** |

### **Night Shade**

|  |
| --- |
| **package move.special; import ru.ifmo.se.pokemon.\*; public class NightShade extends SpecialMove {  public NightShade() {  super(Type.*GHOST*, 0.0, 100.0);  };  protected double calcBaseDamage(Pokemon p1, Pokemon p2) {  return p1.getLevel();  }  protected String describe() {  return "использует Night Shade";  } }** |

|  |
| --- |
| **package move.special; import ru.ifmo.se.pokemon.\*; public class Overheat extends SpecialMove {  public Overheat() {  super(Type.*FIRE*, 130.0, 90.0);  };  public void applySelfEffects(Pokemon p1) {  p1.setMod(Stat.*SPECIAL\_ATTACK*, -2);  }  protected String describe() {  return "использует Overheat";  } }** |

### **Overheat**

|  |
| --- |
| **package move.special; import ru.ifmo.se.pokemon.\*; public class Overheat extends SpecialMove {  public Overheat() {  super(Type.*FIRE*, 130.0, 90.0);  };  public void applySelfEffects(Pokemon p1) {  p1.setMod(Stat.*SPECIAL\_ATTACK*, -2);  }  protected String describe() {  return "использует Overheat";  } }** |

### **Sludge Bomb**

|  |
| --- |
| **package move.special; import ru.ifmo.se.pokemon.\*; public class SludgeBomb extends SpecialMove{  public SludgeBomb(){  super(Type.*FIRE*, 90.0,100.0);  }  private boolean flag;  public void applyOppEffects(Pokemon p2) {  if (Math.*random*() <= 0.3) {  flag = true;  Effect.*poison*(p2);  }  }  protected String describe(){  if(flag) return "использует Sludge Bomb и отравляет";  else return "использует Sludge Bomb";  } }** |

### **Dragon Dance**

|  |
| --- |
| **package move.special; import ru.ifmo.se.pokemon.\*; public class SludgeBomb extends SpecialMove{  public SludgeBomb(){  super(Type.*FIRE*, 90.0,100.0);  }  private boolean flag;  public void applyOppEffects(Pokemon p2) {  if (Math.*random*() <= 0.3) {  flag = true;  Effect.*poison*(p2);  }  }  protected String describe(){  if(flag) return "использует Sludge Bomb и отравляет";  else return "использует Sludge Bomb";  } }** |

|  |
| --- |
| **package move.status; import ru.ifmo.se.pokemon.\*; public class DragonDance extends StatusMove{  public DragonDance() {  super(Type.*DRAGON*, 0.0, 0.0);  }  public void applySelfEffects(Pokemon p1) {  p1.setMod(Stat.*ATTACK*, 1);  p1.setMod(Stat.*SPEED*, 1);  }  protected String describe() {  return "использует Dragon Dance";  } }** |

### **Sand Attack**

|  |
| --- |
| **package move.status; import ru.ifmo.se.pokemon.\*; public class SandAttack extends StatusMove {  public SandAttack(){  super(Type.*GROUND*,0.0,100.0);  }  protected void applyOppEffects(Pokemon p2) {  p2.setMod(Stat.*ACCURACY*, -1);  }  protected String describe() {  return "использует Sand Attack";  } }** |

### **Swagger**

|  |
| --- |
| **package move.status; import ru.ifmo.se.pokemon.\*; public class Swagger extends StatusMove {  public Swagger(){  super(Type.*NORMAL*,0.0,85.0);  }  protected void applyOppEffects(Pokemon p2) {  p2.setMod(Stat.*ATTACK*, 2);  Effect.*confuse*(p2);  }  protected String describe() {  return "использует Swagger";  } }** |

## Pokémon

|  |
| --- |
| **package move.status; import ru.ifmo.se.pokemon.\*; public class Swagger extends StatusMove {  public Swagger(){  super(Type.*NORMAL*,0.0,85.0);  }  protected void applyOppEffects(Pokemon p2) {  p2.setMod(Stat.*ATTACK*, 2);  Effect.*confuse*(p2);  }  protected String describe() {  return "использует Swagger";  } }** |

### **Chandelure**

|  |
| --- |
| **package pokemon; import move.special.DreamEater; import move.special.NightShade; import move.special.FireBlast; import move.special.Overheat; import ru.ifmo.se.pokemon.\*; public class Chandelure extends Pokemon {  public Chandelure(String name, int level){  super(name,level);  setType(new Type[]{Type.*GHOST*, Type.*FIRE*});  setStats(60.0, 55.0, 90.0, 145.0, 90.0, 80.0);  setMove(new Move[]{new DreamEater(), new Overheat(), new NightShade(), new FireBlast()});  } }** |

### **Lampent**

|  |
| --- |
| **package pokemon; import move.special.DreamEater; import move.special.NightShade; import move.special.Overheat; import ru.ifmo.se.pokemon.\*; public class Lampent extends Pokemon {  public Lampent(String name, int level){  super(name,level);  setType(new Type[]{Type.*GHOST*, Type.*FIRE*});  setStats(60.0, 40.0, 60.0, 95.0, 60.0, 55.0);  setMove(new Move[]{new DreamEater(), new Overheat(), new NightShade()});  } }** |

### **Latios**

|  |
| --- |
| **package pokemon; import move.status.DragonDance; import move.special.DracoMeteor; import move.physical.AerialAce; import move.special.IceBeam; import ru.ifmo.se.pokemon.\*; public class Latios extends Pokemon {  public Latios(String name, int level){  super(name,level);  setType(new Type[]{Type.*DRAGON*, Type.*PSYCHIC*});  setStats(80.0, 90.0, 80.0, 130.0, 110.0, 110.0);  setMove(new Move[]{new DragonDance(), new DracoMeteor(), new AerialAce(), new IceBeam()});  } }** |

### **Litwick**

|  |
| --- |
| **package pokemon; import move.special.DreamEater; import move.special.Overheat; import ru.ifmo.se.pokemon.\*; public class Litwick extends Pokemon {  public Litwick(String name, int level){  super(name,level);  setType(new Type[]{Type.*GHOST*, Type.*FIRE*});  setStats(50.0, 30.0, 55.0, 65.0, 55.0, 20.0);  setMove(new Move[]{new DreamEater(), new Overheat()});  } }** |

### **Palossand**

|  |
| --- |
| **package pokemon; import move.physical.RockTomb; import move.status.SandAttack; import move.status.Swagger; import move.special.SludgeBomb; import ru.ifmo.se.pokemon.\*; public class Palossand extends Pokemon {  public Palossand(String name, int level){  super(name,level);  setType(new Type[]{Type.*GHOST*, Type.*FIRE*});  setStats(60.0, 55.0, 90.0, 145.0, 90.0, 80.0);  setMove(new Move[]{new Swagger(), new SandAttack(), new RockTomb(), new SludgeBomb()});  } }** |

### **Sandygast**

|  |
| --- |
| **package pokemon; import move.status.Swagger; import move.status.SandAttack; import move.physical.RockTomb; import ru.ifmo.se.pokemon.\*; public class Sandygast extends Pokemon {  public Sandygast(String name, int level){  super(name,level);  setType(new Type[]{Type.*GHOST*, Type.*GROUND*});  setStats(55.0, 55.0, 80.0, 70.0, 45.0, 15.0);  setMove(new Move[]{new Swagger(), new SandAttack(), new RockTomb()});  } }** |

## Main

|  |
| --- |
| **import pokemon.\*; import ru.ifmo.se.pokemon.Battle; public class Main {  public Main() {  }  public static void main(String[] var0) {  Battle b = new Battle();  Latios p1 = new Latios("Латиос", 2);  Sandygast p2 = new Sandygast("Сандигаст", 2);  Palossand p3 = new Palossand("Палоссанд", 2);  Litwick p4 = new Litwick("Литвик", 2);  Lampent p5 = new Lampent("Лампент", 2);  Chandelure p6 = new Chandelure("Чанделур", 2);  b.addAlly(p1);  b.addAlly(p3);  b.addAlly(p5);  b.addFoe(p2);  b.addFoe(p4);  b.addFoe(p6);  b.go();  }**  **}** |

# Результат работы программы

|  |
| --- |
| **Latios Латиос из команды красных вступает в бой!**  **Sandygast Сандигаст из команды синих вступает в бой!**  **Latios Латиос промахивается**  **Sandygast Сандигаст использует Rock Tomb.**  **Latios Латиос теряет 6 здоровья.**  **Latios Латиос уменьшает скорость.**  **Latios Латиос использует Draco Meteor.**  **Критический удар!**  **Sandygast Сандигаст теряет 31 здоровья.**  **Latios Латиос уменьшает специальную атаку.**  **Sandygast Сандигаст теряет сознание.**  **Litwick Литвик из команды синих вступает в бой!**  **Latios Латиос использует Aerial Ace.**  **Litwick Литвик восстанавливает 1 здоровья.**  **Litwick Литвик использует Overheat.**  **Latios Латиос теряет 4 здоровья.**  **Litwick Литвик уменьшает специальную атаку.**  **Latios Латиос использует Ice Beam.**  **Litwick Литвик восстанавливает 1 здоровья.**  **Litwick Литвик замерзает**  **Litwick Литвик использует Overheat.**  **Latios Латиос восстанавливает 1 здоровья.**  **Litwick Литвик уменьшает специальную атаку.**  **Latios Латиос использует Draco Meteor.**  **Litwick Литвик восстанавливает 1 здоровья.**  **Latios Латиос уменьшает специальную атаку.**  **Litwick Литвик оттаивает**  **Latios Латиос использует Aerial Ace.**  **Litwick Литвик теряет 4 здоровья.**  **Litwick Литвик использует Overheat.**  **Latios Латиос теряет 4 здоровья.**  **Litwick Литвик уменьшает специальную атаку.**  **Latios Латиос использует Aerial Ace.**  **Litwick Литвик теряет 2 здоровья.**  **Litwick Литвик использует Overheat.**  **Latios Латиос теряет 2 здоровья.**  **Litwick Литвик уменьшает специальную атаку.**  **Latios Латиос промахивается**  **Litwick Литвик использует Dream Eater.**  **Latios Латиос теряет 1 здоровья.**  **Latios Латиос теряет сознание.**  **Palossand Палоссанд из команды красных вступает в бой!**  **Palossand Палоссанд использует Rock Tomb.**  **Litwick Литвик теряет 7 здоровья.**  **Litwick Литвик уменьшает скорость.**  **Litwick Литвик использует Dream Eater.**  **Palossand Палоссанд теряет 3 здоровья.**  **Palossand Палоссанд использует Rock Tomb.**  **Litwick Литвик теряет 10 здоровья.**  **Litwick Литвик уменьшает скорость.**  **Litwick Литвик теряет сознание.**  **Chandelure Чанделур из команды синих вступает в бой!**  **Chandelure Чанделур использует Night Shade.**  **Chandelure Чанделур теряет 25 здоровья.**  **Оба покемона теряют сознание.**  **В команде синих не осталось покемонов.**  **Команда красных побеждает в этом бою!** |

# Вывод

В ходе данной работы я научился подключать внешние jar файлы и использовать их. Научился работать с документацией, изучил объектно-ориентированный подход на языке Java. Научился работать с классами, конструкторами, полями и модификаторами доступа.