Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский университет ИТМО»
Факультет программной инженерии и компьютерной техники

Лабораторная работа по программированию №2

Вариант 31015

Выполнил
Пчелкин Илья Игоревич
P3106

Проверил
Вербовой А. А.

Санкт-Петербург 2024

ОГЛАВЛЕНИЕ

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 (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.

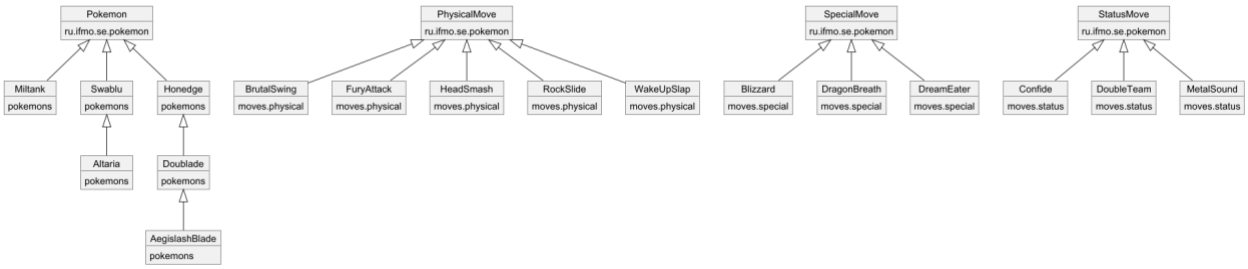
Что надо сделать (краткое описание)

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.
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.

UML – Диаграмма



Код программы

moves

physical

```
package moves.physical;

import ru.ifmo.se.pokemon.PhysicalMove;
import ru.ifmo.se.pokemon.Type;


public class BrutalSwing extends PhysicalMove {
    public BrutalSwing() {
        super(Type.DARK, 60, 100); // type, power, accuracy
    }
    @Override
    public String describe() {
        return "использует Brutal Swing";
    }
}



package moves.physical;

import ru.ifmo.se.pokemon.*;

public class FuryAttack extends PhysicalMove {
    public FuryAttack() {
        super(Type.NORMAL, 15, 85);
    }

    @Override
    protected void applySelfEffects(Pokemon pokemon){
        Effect powerX = new Effect().chance((double) 3/8).stat(Stat.ATTACK, 2).chance((double)
3/8).stat(Stat.ATTACK, 3).chance((double) 1/8).stat(Stat.ATTACK, 4).chance((double) 1/8).stat(Stat.ATTACK, 5);
        pokemon.addEffect(powerX);
    }

    @Override
    public String describe(){
        return "использует Fury Attack";
    }
}


package moves.physical;

import ru.ifmo.se.pokemon.*;

public class HeadSmash extends PhysicalMove {
    public HeadSmash(){
        super(Type.ROCK, 150, 80);
    }


    @Override
    protected void applySelfDamage(Pokemon pokemon, double damage){
        pokemon.setMod(Stat.HP, (int) (0.5 * Math.round(damage)));
    }
    @Override
    public String describe(){
        return "использует Head Smash";
    }
}



package moves.physical;

import ru.ifmo.se.pokemon.*;

public class RockSlide extends PhysicalMove {
    public RockSlide(){
        super(Type.ROCK, 75, 90);
    }

    private boolean isFlinched = false;
    @Override
    protected void applyOppEffects(Pokemon pokemon){
        if ((Math.random() < 0.3) & (isFlinched)){
            Effect.flinch(pokemon);
            isFlinched = true;
        }
    }

    @Override
```

```
    protected String describe(){
        return "использует Rock Slide";
    }
}
```

```
package moves.physical;

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


public class WakeUpSlap extends PhysicalMove {
    public WakeUpSlap() {
        super(Type.FIGHTING, 70, 100);  // type, power, accuracy
    }

    @Override
    protected void applyOppEffects(Pokemon pokemon) {
        Effect normalCondition = new Effect().condition(Status.NORMAL);
        if (pokemon.getCondition() == Status.SLEEP){
            pokemon.setMod(Stat.ATTACK, 2);
            pokemon.setCondition(normalCondition);
        }
    }
    @Override
    protected String describe() {
        return "использует Wake-Up Slap";
    }
}
```

special

```
package moves.special;

import ru.ifmo.se.pokemon.*;

public class Blizzard extends SpecialMove {
    public Blizzard(){
        super(Type.ICE, 110, 70);
    }
    @Override
    protected void applyOppEffects(Pokemon pokemon){
        Effect freeze = new Effect().chance(0.1).condition(Status.FREEZE);
        pokemon.addEffect(freeze);
    }

    @Override
    public String describe(){
        return "использует Blizzard";
    }
}
```

```
package moves.special;

import ru.ifmo.se.pokemon.*;

public class DragonBreath extends SpecialMove {
    public DragonBreath() {
        super(Type.DRAGON, 60, 100);
    }
    @Override
    protected void applyOppEffects(Pokemon pokemon){
        Effect paralyze = new Effect().chance(0.3).condition(Status.PARALYZE);
        pokemon.addEffect(paralyze);
    }
    @Override
    public String describe(){
        return "использует Dragon Breath";
    }
}
```

```java
package moves.special;

import ru.ifmo.se.pokemon.*;

public class DreamEater extends SpecialMove {
    public DreamEater(){
        super(Type.PSYCHIC, 100, 100);
    }

    @Override
    protected void applySelfDamage(Pokemon attackingPokemon, double damage){
        attackingPokemon.setMod(Stat.HP, -((int) Math.round(damage * 0.5)));
    }
    @Override
    protected void applyOppDamage(Pokemon defendingPokemon, double damage){
        if (defendingPokemon.getCondition() == Status.SLEEP){
            defendingPokemon.setMod(Stat.HP, (int) Math.round(damage));
        }
    }
    @Override
    public String describe(){
        return "использует Dream Eater";
    }
}
```

status

```java
package moves.status;

import ru.ifmo.se.pokemon.*;

public class Confide extends StatusMove {
    public Confide(){
        super(Type.NORMAL, 0, 0);
    }


    @Override
    protected void applyOppEffects(Pokemon pokemon){
        pokemon.setMod(Stat.SPECIAL_ATTACK, -1);
    }

    @Override
    public String describe(){
        return "использует Confide";
    }
}
```

```java
package moves.status;

import ru.ifmo.se.pokemon.*;


public class DoubleTeam extends StatusMove {
    public DoubleTeam() {
        super(Type.NORMAL, 0, 0); // type, power, accuracy
    }

    private int count = 0;
    @Override
    protected void applySelfEffects(Pokemon pokemon) {
        pokemon.setMod(Stat.EVASION, 1);
        count++;
        if (count >= 6){
            pokemon.setMod(Stat.EVASION, 0);
        }
    }

    @Override
    protected String describe() {
        return "использует Double Team";
    }
}
```

```java
package moves.status;

import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Stat;
import ru.ifmo.se.pokemon.StatusMove;
import ru.ifmo.se.pokemon.Type;

public class MetalSound extends StatusMove {
   public MetalSound(){
       super(Type.STEEL, 0, 85);
   }

   private int count = 0;
   @Override
   protected void applyOppEffects(Pokemon pokemon){
       pokemon.setMod(Stat.SPECIAL_DEFENSE, -2);
       count++;
       if (count >= 3){
           pokemon.setMod(Stat.SPECIAL_DEFENSE, 0);
       }
   }


   @Override
   public String describe(){
       return "использует Metal Sound";
   }
}
```

pokemons

```java
package pokemons;

import moves.physical.HeadSmash;

public class AegislashBlade extends Doublade{
    public AegislashBlade(String name, int level){
        super(name, level);
        setStats(60, 140, 50, 140, 50, 60);
        addMove(new HeadSmash());
    }
}


package pokemons;

import moves.special.DragonBreath;
import ru.ifmo.se.pokemon.Type;

public class Altaria extends Swablu{
    public Altaria(String name, int level) {
        super(name, level);
        setStats(75, 70, 90, 70, 105, 80);
        addType(Type.DRAGON);
        addMove(new DragonBreath());
    }
}



package pokemons;
import moves.status.MetalSound;


public class Doublade extends Honedge {
    public Doublade(String name, int level){
        super(name, level);
        setStats(59, 110, 150, 45, 49, 35);
        addMove(new MetalSound());
    }
}



package pokemons;

import moves.physical.BrutalSwing;
import moves.status.DoubleTeam;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class Honedge extends Pokemon
{
    public Honedge(String name, int level)
    {
        super(name, level);
        setStats(45, 80, 100, 35, 37, 28);
        setType(Type.STEEL, Type.GHOST);
        setMove(new BrutalSwing(), new DoubleTeam());
    }
}
```

```java
package pokemons;

import moves.special.Blizzard;
import moves.status.DoubleTeam;
import moves.physical.RockSlide;
import moves.physical.WakeUpSlap;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class Miltank extends Pokemon {
    public Miltank(String name, int level){
        super(name, level);
        setStats(90, 80, 105, 40, 70, 100);
        setType(Type.NORMAL);
        setMove(new WakeUpSlap(), new RockSlide(), new Blizzard(), new DoubleTeam());
    }
}
```

```java
package pokemons;

import moves.status.Confide;
import moves.special.DreamEater;
import moves.physical.FuryAttack;
import ru.ifmo.se.pokemon.Pokemon;
import ru.ifmo.se.pokemon.Type;

public class Swablu extends Pokemon {
    public Swablu(String name, int level) {
        super(name, level);
        setType(Type.NORMAL, Type.FLYING);
        setStats(45, 40, 60, 40, 75, 50);
        setMove(new Confide(), new FuryAttack(), new DreamEater());
    }
}
```

Main

```java
import pokemons.*;
import ru.ifmo.se.pokemon.Battle;

public class Main {
    public static void main(String[] args) {
        Battle b = new Battle();
        Miltank p1 = new Miltank("Молоко", 40);
        Swablu p2 = new Swablu("Говорун", 12);
        Altaria p3 = new Altaria("Летун", 20);
        Honedge p4 = new Honedge("Спичка", 16);
        Doublade p5 = new Doublade("Нож", 16);
        AegislashBlade p6 = new AegislashBlade("Меч", 16);
        b.addAlly(p1);
        b.addAlly(p2);
        b.addAlly(p3);
        b.addFoe(p4);
        b.addFoe(p5);
        b.addFoe(p6);
        b.go();
    }
}
```

Результат работы программы

Miltank Мазурова из команды полосатых вступает в бой!

Honedge Ножик из команды фиолетовых вступает в бой!

Miltank Мазурова использует Rock Slide.

Honedge Ножик теряет 14 здоровья.

Honedge Ножик промахивается

Miltank Мазурова промахивается

Honedge Ножик использует Brutal Swing.

Miltank Мазурова теряет 6 здоровья.

Miltank Мазурова использует Rock Slide.

Honedge Ножик теряет 17 здоровья.

Honedge Ножик использует Brutal Swing.

Miltank Мазурова теряет 3 здоровья.

Miltank Мазурова использует Blizzard.

Honedge Ножик теряет 16 здоровья.

Honedge Ножик замерзает

Honedge Ножик теряет сознание.

Doublade Нож из команды фиолетовых вступает в бой!

Miltank Мазурова использует Blizzard.

Doublade Нож теряет 19 здоровья.

Doublade Нож промахивается

Miltank Мазурова использует Wake-Up Slap.

Критический удар!

Doublade Нож теряет 1 здоровья.

Doublade Нож не замечает воздействие типа FIGHTING

Doublade Нож использует Brutal Swing.

Miltank Мазурова теряет 5 здоровья.

Miltank Мазурова использует Rock Slide.

Doublade Нож теряет 13 здоровья.

Doublade Нож промахивается

Miltank Мазурова использует Wake-Up Slap.

Критический удар!

Doublade Нож теряет 1 здоровья.

Doublade Нож не замечает воздействие типа FIGHTING

Doublade Нож использует Metal Sound.

Miltank Мазурова уменьшает специальную защиту.

Miltank Мазурова использует Rock Slide.

Doublade Нож теряет 11 здоровья.

Doublade Нож промахивается

Miltank Мазурова промахивается

Doublade Нож промахивается

Miltank Мазурова использует Wake-Up Slap.

Doublade Нож теряет 1 здоровья.

Doublade Нож не замечает воздействие типа FIGHTING

Doublade Нож использует Metal Sound.

Miltank Мазурова уменьшает специальную защиту.

Miltank Мазурова использует Rock Slide.

Doublade Нож теряет 9 здоровья.

Doublade Нож теряет сознание.

AegislashBlade Ножище из команды фиолетовых вступает в бой!

Miltank Мазурова использует Wake-Up Slap.

AegislashBlade Ножище теряет 1 здоровья.

AegislashBlade Ножище не замечает воздействие типа FIGHTING

AegislashBlade Ножище использует Brutal Swing.

Miltank Мазурова теряет 7 здоровья.

Miltank Мазурова использует Rock Slide.

AegislashBlade Ножище теряет 28 здоровья.

AegislashBlade Ножище промахивается

Miltank Мазурова использует Blizzard.

AegislashBlade Ножище теряет 23 здоровья.

AegislashBlade Ножище теряет сознание.

В команде фиолетовых не осталось покемонов.

Команда полосатых побеждает в этом бою!

Заключение

При выполнении данной лабораторной работы я изучил основы ООП.