

Содержание

| | |
|--------------------------------------------------------|----|
| Задание | 2 |
| Диаграмма классов реализованной объектной модели | 3 |
| Исходный код программы..... | 4 |
| Класс AuraSphere | 4 |
| Класс Confide | 4 |
| Класс DoubleTeam | 5 |
| Класс Facade..... | 5 |
| Класс FocusBlast | 6 |
| Класс Growl..... | 6 |
| Класс Headbutt | 7 |
| Класс Rest..... | 7 |
| Класс RockSlide | 8 |
| Класс RockTomb | 8 |
| Класс Thunder | 9 |
| Класс Thunderbolt | 9 |
| Класс Dialga | 10 |
| Класс Cranidos | 10 |
| Класс Rampardos..... | 11 |
| Класс NidoranF..... | 11 |
| Класс Nidorina..... | 11 |
| Класс Nidoqueen | 12 |
| Класс Main | 12 |
| Результат работы программы | 13 |
| Вывод | 15 |

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:

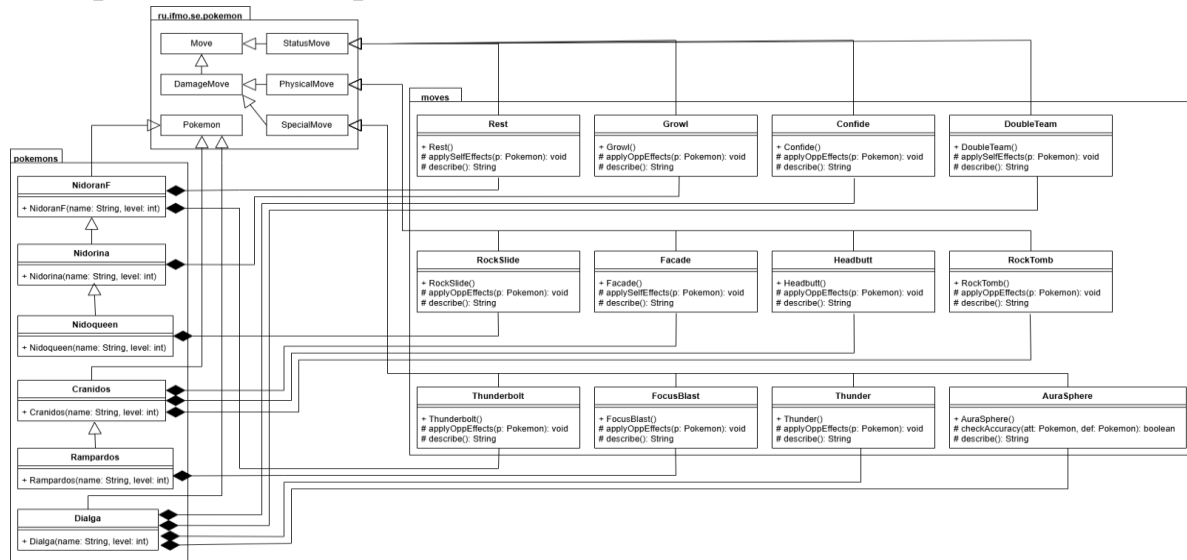
- очки здоровья (HP)
- атака (attack)
- защита (defense)
- специальная атака (special attack)
- специальная защита (special defense)
- скорость (speed)

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.

Список покемонов и их атак:

- Dialga (Confide, Thunder, Double Team, Aura Sphere)
- Cranidos (Facade, Headbutt, Rock Tomb)
- Rampardos (Facade, Headbutt, Rock Tomb, Focus Blast)
- Nidoran-F (Thunderbolt, Rest)
- Nidorina (Thunderbolt, Rest, Growl)
- Nidoqueen (Thunderbolt, Rest, Growl, Rock Slide)

Диаграмма классов реализованной объектной модели



Исходный код программы

Класс AuraSphere

```
package moves;

import ru.ifmo.se.pokemon.*;

public final class AuraSphere extends SpecialMove {
    public AuraSphere() {
        super(Type.FIGHTING, 80, 1);
    }

    @Override protected boolean checkAccuracy(Pokemon att, Pokemon def) {
        return true;
    }

    @Override protected String describe() {
        return "выпускает шар ауры";
    }
}
```

Класс Confide

```
package moves;

import ru.ifmo.se.pokemon.*;

public final class Confide extends StatusMove {
    public Confide() {
        super(Type.NORMAL, 0, 1);
    }

    @Override protected void applyOppEffects(Pokemon p) {
        Effect e = new Effect().stat(Stat.SPECIAL_ATTACK, -1);
        p.addEffect(e);
    }

    @Override protected String describe() {
        return "отвлекает противника";
    }
}
```

Класс DoubleTeam

```

package moves;

import ru.ifmo.se.pokemon.*;

public final class DoubleTeam extends StatusMove {
    public DoubleTeam() {
        super(Type.NORMAL, 0, 1);
    }

    @Override protected void applySelfEffects(Pokemon p) {
        Effect e = new Effect().stat(Stat.EVASION, 1);
        p.addEffect(e);
    }

    @Override protected String describe() {
        return "быстро перемещается";
    }
}

```

Класс Facade

```

package moves;

import ru.ifmo.se.pokemon.*;

public final class Facade extends PhysicalMove {
    public Facade() {
        super(Type.NORMAL, 70, 1);
    }

    @Override protected void applySelfEffects(Pokemon p) {
        if (p.getCondition() == Status.PARALYZE ||
            p.getCondition() == Status.BURN ||
            p.getCondition() == Status.POISON) {
            Effect e = new Effect().stat(Stat.ATTACK, 1);
            p.addEffect(e);
        }
    }

    @Override protected String describe() {
        return "показывает силу";
    }
}

```

Класс FocusBlast

```

package moves;

import ru.ifmo.se.pokemon.*;

public final class FocusBlast extends SpecialMove {
    public FocusBlast() {
        super(Type.FIGHTING, 120, 0.7);
    }

    @Override protected void applyOppEffects(Pokemon p) {
        if (Math.random() < 0.1) {
            Effect e = new Effect().stat(Stat.SPECIAL_DEFENSE, -1);
            p.addEffect(e);
        }
    }

    @Override protected String describe() {
        return "ментально атакует противника";
    }
}

```

Класс Growl

```

package moves;

import ru.ifmo.se.pokemon.*;

public final class Growl extends StatusMove {
    public Growl() {
        super(Type.NORMAL, 0, 1);
    }

    @Override protected void applyOppEffects(Pokemon p) {
        p.addEffect(new Effect().stat(Stat.ATTACK, -1));
    }

    @Override protected String describe() {
        return "рычит на противника";
    }
}

```

Класс Headbutt

```

package moves;

import ru.ifmo.se.pokemon.*;

public final class Headbutt extends PhysicalMove {

    public Headbutt() {
        super(Type.NORMAL, 70, 1);
    }

    @Override protected void applyOppEffects(Pokemon p) {
        if (Math.random() < 0.3) {
            Effect.flinch(p);
        }
    }

    @Override protected String describe() {
        return "ударяет противника головой";
    }
}

```

Класс Rest

```

package moves;

import ru.ifmo.se.pokemon.*;

public final class Rest extends StatusMove {
    public Rest() {
        super(Type.PSYCHIC, 0, 1);
    }

    @Override protected void applySelfEffects(Pokemon p) {
        p.addEffect(new Effect().stat(Stat.HP, (int) -(12 - p.getHP())).turns(1));
        p.addEffect(new Effect().attack(0).turns(2));
    }

    @Override protected String describe() {
        return "решает отдохнуть";
    }
}

```

Класс RockSlide

```

package moves;

import ru.ifmo.se.pokemon.*;

public final class RockSlide extends PhysicalMove {
    public RockSlide() {
        super(Type.ROCK, 75, 0.9);
    }

    @Override protected void applyOppEffects(Pokemon p) {
        if (Math.random() < 0.3) {
            Effect.flinch(p);
        }
    }

    @Override protected String describe() {
        return "скидывает булыжник";
    }
}

```

Класс RockTomb

```

package moves;

import ru.ifmo.se.pokemon.*;

public final class RockTomb extends PhysicalMove {
    public RockTomb() {
        super(Type.ROCK, 60, 0.95);
    }

    @Override protected void applyOppEffects(Pokemon p) {
        Effect e = new Effect().stat(Stat.SPEED, -1);
        p.addEffect(e);
    }

    @Override protected String describe() {
        return "запускает камни в противника";
    }
}

```

Класс Thunder


```

package moves;

import ru.ifmo.se.pokemon.*;

public final class Thunder extends SpecialMove {
    public Thunder() {
        super(Type.ELECTRIC, 110, 0.7);
    }

    @Override protected void applyOppEffects(Pokemon p) {
        if (Math.random() < 0.3) {
            Effect.paralyze(p);
        }
    }

    @Override protected String describe() {
        return "вызывает Грозу";
    }
}

```

Класс Thunderbolt

```

package moves;

import ru.ifmo.se.pokemon.*;

public final class Thunderbolt extends SpecialMove {
    public Thunderbolt() {
        super(Type.ELECTRIC, 90, 1);
    }

    @Override protected void applyOppEffects(Pokemon p) {
        if (Math.random() < 0.1) {
            Effect.paralyze(p);
        }
    }

    @Override protected String describe() {
        return "вызывает молнию";
    }
}

```

Класс Dialga

```

package pokemons;

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

public final class Dialga extends Pokemon {
    public Dialga(String name, int level) {
        super(name, level);
        setStats(13, 8, 8, 8, 7, 7);
        setType(Type.STEEL, Type.DRAGON);
        setMove(new Confide(), new Thunder(), new DoubleTeam(), new AuraSphere());
    }
}

```

Класс Cranidos

```

package pokemons;

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

public class Cranidos extends Pokemon {
    public Cranidos(String name, int level) {
        super(name, level);
        setStats(12, 8, 6, 6, 6, 6);
        setType(Type.ROCK);
        setMove(new Facade(), new Headbutt(), new RockTomb());
    }
}

```

Класс Rampardos

```

package pokemons;

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

public class Cranidos extends Pokemon {
    public Cranidos(String name, int level) {
        super(name, level);
        setStats(12, 8, 6, 6, 6, 6);
        setType(Type.ROCK);
        setMove(new Facade(), new Headbutt(), new RockTomb());
    }
}

```

Класс NidoranF

```

package pokemons;

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

public class NidoranF extends Pokemon {
    public NidoranF(String name, int level) {
        super(name, level);
        setStats(12, 6, 6, 6, 6, 6);
        setType(Type.POISON);
        setMove(new Thunderbolt(), new Rest());
    }
}

```

Класс Nidorina

```

package pokemons;

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

public class Nidorina extends NidoranF {
    public Nidorina(String name, int level) {
        super(name, level);
        setStats(13, 6, 6, 6, 6, 6);
        addMove(new Growl());
    }
}

```

Класс Nidoqueen

```

package pokemons;

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

public final class Nidoqueen extends Nidorina {
    public Nidoqueen(String name, int level) {
        super(name, level);
        setStats(13, 7, 7, 7, 7, 7);
        addMove(new RockSlide());
    }
}

```

Класс Main

```

package main;

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

public class Main {
    public static void main(String[] args){
        Battle b = new Battle();
        Pokemon p1 = new Dialga("Dialga", 1);
        Pokemon p2 = new Cranidos("Cranidos", 1);
        Pokemon p3 = new Rampardos("Rampardos", 1);
        Pokemon p4 = new NidoranF("NidoranF", 1);
        Pokemon p5 = new Nidorina("Nidorina", 1);
        Pokemon p6 = new Nidoqueen("Nidoqueen", 1);
        b.addAlly(p1);
        b.addAlly(p2);
        b.addAlly(p3);
        b.addFoe(p4);
        b.addFoe(p5);
        b.addFoe(p6);
        b.go();
    }
}

```

Результат работы программы

При каждом запуске программы вывод программы будет ращным. Один из примеров вывода программы:

Dialga Dialga из команды полосатых вступает в бой!

NidoranF NidoranF из команды фиолетовых вступает в бой!

Dialga Dialga выпускает шар ауры.

NidoranF NidoranF теряет 3 здоровья.

NidoranF NidoranF вызывает молнию.

Dialga Dialga теряет 2 здоровья.

Dialga Dialga выпускает шар ауры.

NidoranF NidoranF теряет 3 здоровья.

NidoranF NidoranF вызывает молнию.

Dialga Dialga теряет 3 здоровья.

Dialga Dialga вызывает Грозу.

NidoranF NidoranF теряет 7 здоровья.

NidoranF NidoranF парализован

NidoranF NidoranF теряет сознание.

Nidorina Nidorina из команды фиолетовых вступает в бой!

Dialga Dialga вызывает Грозу.

Nidorina Nidorina теряет 6 здоровья.

Nidorina Nidorina решает отдохнуть.

Nidorina Nidorina восстанавливает 6 здоровья.

Dialga Dialga отвлекает противника.

Dialga Dialga вызывает Грозу.

Nidorina Nidorina теряет 7 здоровья.

Nidorina Nidorina рычит на противника.

Dialga Dialga вызывает Грозу.

Nidorina Nidorina теряет 5 здоровья.

Nidorina Nidorina парализован

Nidorina Nidorina теряет сознание.

Nidoqueen Nidoqueen из команды фиолетовых вступает в бой!

Dialga Dialga выпускает шар ауры.

Dialga Dialga теряет 9 здоровья.

Оба покемона теряют сознание.

Cranidos Cranidos из команды полосатых вступает в бой!

Nidoqueen Nidoqueen вызывает молнию.

Cranidos Cranidos теряет 5 здоровья.

Cranidos Cranidos ударяет противника головой.

Nidoqueen Nidoqueen теряет 5 здоровья.

Nidoqueen Nidoqueen скидывает булыжник.

Cranidos Cranidos теряет 4 здоровья.

Cranidos Cranidos запускает камни в противника.

Nidoqueen Nidoqueen теряет 7 здоровья.

Nidoqueen Nidoqueen теряет сознание.

В команде фиолетовых не осталось покемонов.

Команда полосатых побеждает в этом бою!

Вывод

Во время выполнения данной лабораторной работы были изучены основы Объектно-ориентированного программирования, стандартные способы работы с классами и объектами, а также способы работы с пакетами.