Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет Программной инженерии и компьютерной техники

Программирование

Лабораторная работа № 2

Вариант № 1323432

Выполнил:

Сандов Кирилл Алекссевич

Группа:

P3113

Проверил:

преподаватель практики Сорокин Роман Борисович

г. Санкт-Петербург

2022

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.

Диаграмма классов модели

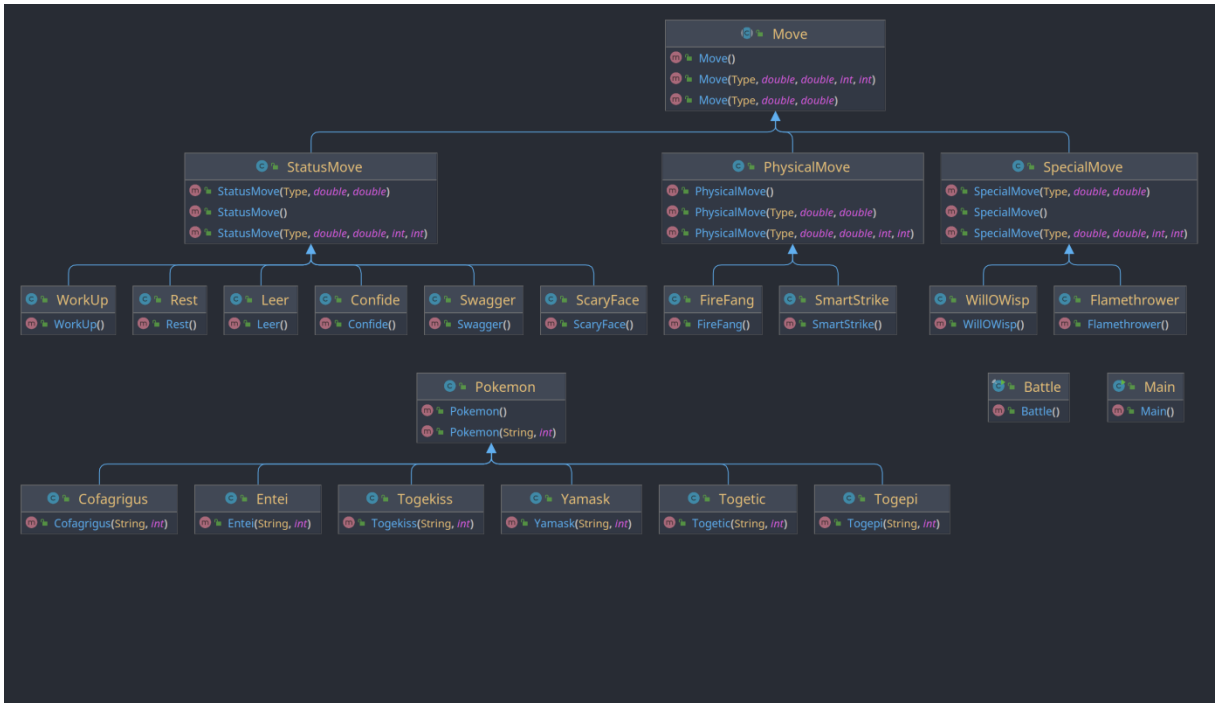

Рисунок 1 – «UML-диаграмма классов построенной объектной модели»

Исходный код программы

src/Main.java:

```
package src;

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

public class Main {
    public static void main(String[] args) {
        Battle b = new Battle();

        Pokemon[] allies = {
                new Togepi("Eggman", 5),
                new Togetic("Chad Eggman", 8),
                new Togekiss("Flying Gigachad Eggman", 1)
        };
        for (Pokemon p : allies)
            b.addAlly(p);

        Pokemon[] foes = {
                new Entei("Doge", 1),
                new Cofagrigus("Prizrachniy Gonchik", 3),
                new Yamask("Casper the Ghost", 5)
        };
        for (Pokemon p : foes)
            b.addFoe(p);

        b.go();
    }
}
```

src/pokemons/Cofagrigus.java:

```
package src.pokemons;

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

public class Cofagrigus extends Pokemon {
    public Cofagrigus(String name, int level) {
        super(name, level);
        setStats(58, 50, 145, 95, 105, 30);
        setType(Type.GHOST);
        setMove(new WillOWisp(), new Confide(), new Swagger(), new
ScaryFace());
    }
}
```

src/pokemons/Entei.java:

```
package src.pokemons;
```

```java
import ru.ifmo.se.pokemon.*;
import src.moves.*;

public class Entei extends Pokemon {
    public Entei(String name, int level) {
        super(name, level);
        setStats(115, 115, 85, 90, 75, 100);
        setType(Type.FIRE);
        setMove(new Flamethrower(), new Leer(), new Swagger(), new
FireFang());
    }
}
```

src/pokemons/Togekiss.java:

```java
package src.pokemons;

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

public class Togekiss extends Pokemon {
    public Togekiss(String name, int level) {
        super(name, level);
        setStats(85, 50, 95, 120, 115, 80);
        setType(Type.FAIRY, Type.FLYING);
        setMove(new Flamethrower(), new Rest(), new SmartStrike(), new
WorkUp());
    }
}
```

src/pokemons/Togepi.java:

```java
package src.pokemons;

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

public class Togepi extends Pokemon {
    public Togepi(String name, int level) {
        super(name, level);
        setStats(35, 20, 65, 40, 65, 20);
        setType(Type.FAIRY);
        setMove(new Flamethrower(), new Rest());
    }
}
```

src/pokemons/Togetic.java:

```java
package src.pokemons;

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

```
public class Togetic extends Pokemon {
    public Togetic(String name, int level) {
        super(name, level);
        setStats(55, 40, 85, 80, 105, 40);
        setType(Type.FAIRY, Type.FLYING);
        setMove(new Flamethrower(), new Rest(), new SmartStrike());
    }
}
```

src/pokemons/Yamask.java:

```
package src.pokemons;

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

public class Yamask extends Pokemon {
    public Yamask(String name, int level) {
        super(name, level);
        setStats(38, 30, 85, 55, 65, 30);
        setType(Type.GHOST);
        setMove(new WillOWisp(), new Confide(), new Swagger());
    }
}
```

src/moves/Confide.java:

```
package src.moves;

import ru.ifmo.se.pokemon.*;

public class Confide extends StatusMove {
    public Confide() {
        /*
        Confide lowers the target's Special Attack by one stage.
        */
        super(Type.NORMAL, 0, 0);
    }

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

    protected String describe() {
        return "делится своими переживаниями";

    }
}
```

src/moves/FireFang.java:

```
package src.moves;

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

```
public class FireFang extends PhysicalMove {
    public FireFang() {
        /*
        Fire Fang deals damage, has a 10% chance of burning the target and
has a 10% chance of causing the target
        to flinch (if the target has not yet moved).
        */
        super(Type.FIRE, 65, 95);
    }

    protected void applyOppEffects(Pokemon pokemon) {
        pokemon.addEffect(new Effect().chance(0.1).condition(Status.BURN));
        // Dummy 10% chance check
        if (Math.random() <= 0.1)
            Effect.flinch(pokemon);
    }

    protected String describe() {
        return "кусает клыком огня";
    }
}
```

src/moves/Flamethrower.java:

```
package src.moves;

import ru.ifmo.se.pokemon.*;

public class Flamethrower extends SpecialMove {
    public Flamethrower() {
        /*
        Flamethrower deals damage and has a 10% chance of burning the
target.
        */
        super(Type.FIRE, 90, 100);
    }

    protected void applyOppEffects(Pokemon pokemon) {
        pokemon.addEffect(new Effect().chance(0.1).condition(Status.BURN));
    }

    protected String describe() {
        return "поджигает";
    }
}
```

src/moves/Leer.java:

```
package src.moves;

import ru.ifmo.se.pokemon.*;

public class Leer extends StatusMove {
```

```
    public Leer() {
        /*
            Leer lowers the target's Defense by one stage.
            Stats can be lowered to a minimum of -6 stages each.
        */
        super(Type.NORMAL, 0, 100);
    }

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

    protected String describe() {
        return "зловеще смотрит";
    }
}
```

src/moves/Rest.java:

```
package src.moves;

import ru.ifmo.se.pokemon.*;

public class Rest extends StatusMove {
    public Rest() {
        /*
        User sleeps for 2 turns, but user is fully healed.
        */
        super(Type.PSYCHIC, 0, 0);
    }

    protected void applySelfEffects(Pokemon pokemon) {
        pokemon.restore();
        pokemon.addEffect(new Effect().condition(Status.SLEEP).turns(2));
    }

    protected boolean checkAccuracy(Pokemon pokemon, Pokemon pokemon1) {
        return true;
    }

    protected String describe() {
        return "решил прилечь поспать";
    }
}
```

src/moves/ScaryFace.java:

```
package src.moves;

import ru.ifmo.se.pokemon.*;

public class ScaryFace extends StatusMove {
    public ScaryFace() {
        /*
        Scary Face lowers the target's Speed by two stages.
```

```
        */
        super(Type.NORMAL, 0, 100);
    }

    protected void applyOppEffects(Pokemon pokemon) {
        pokemon.setMod(Stat.SPEED, -2);
    }

    protected String describe() {
        return "пугает своим страшным лицом";
    }
}
```

src/moves/SmartStrike.java:

```
package src.moves;

import ru.ifmo.se.pokemon.*;

public class SmartStrike extends PhysicalMove {
    public SmartStrike() {
        /*
        The user stabs the target with a sharp horn. This attack never
misses.
        */
        super(Type.STEEL, 70, 0);
    }

    protected boolean checkAccuracy(Pokemon pokemon, Pokemon pokemon1) {
        // Because *attack never misses*
        return true;
    }

    protected String describe() {
        return "ударяет острым рогом";
    }
}
```

src/moves/Swagger.java:

```
package src.moves;

import ru.ifmo.se.pokemon.*;

public class Swagger extends StatusMove {
    public Swagger() {
        /*
            Swagger confuses the target and raises its Attack by two
stages.
            If one of the two effects cannot be invoked
            (for example the target is already confused or its Attack is
already raised to the maximum of +6 stages),
            Swagger still works and will invoke the other effect.
        */
        super(Type.NORMAL, 0, 85);
```

```
    }

    protected void applyOppEffects(Pokemon pokemon) {
        Effect.confuse(pokemon);
        pokemon.setMod(Stat.ATTACK, 2);
    }

    protected String describe() {
        return "становится наглым";
    }
}
```

src/moves/WillOWisp.java:

```
package src.moves;

import ru.ifmo.se.pokemon.*;

public class WillOWisp extends SpecialMove {
    public WillOWisp() {
        /*
        Will-O-Wisp causes the target to become burned, if it hits.
        */
        super(Type.FIRE, 0, 75);
    }

    protected void applyOppEffects(Pokemon pokemon) {
        Effect.burn(pokemon);
    }

    protected String describe() {
        return "становится огненным призраком";
    }
}
```

src/moves/Workup.java:

```
package src.moves;

import ru.ifmo.se.pokemon.*;

public class WorkUp extends StatusMove {
    public WorkUp() {
        /*
        Work Up raises the user's Attack and Special Attack by one stage
each.
        */
        super(Type.NORMAL, 0, 0);
    }

    protected void applySelfEffects(Pokemon pokemon) {
        pokemon.setMod(Stat.ATTACK, 1);
        pokemon.setMod(Stat.SPECIAL_ATTACK, 1);
    }
```

```
    protected boolean checkAccuracy(Pokemon pokemon, Pokemon pokemon1) {
        return true;
    }

    protected String describe() {
        return "получает мотивацию";
    }
}
```

Результат работы программы

```
Togepi Eggman из команды фиолетовых вступает в бой!
Entei Doge из команды черных вступает в бой!
Togepi Eggman решил прилечь поспать.
Togepi Eggman засыпает

Entei Doge поджигает.
Togepi Eggman теряет 7 здоровья.

Togepi Eggman поджигает.
Entei Doge теряет 5 здоровья.

Entei Doge поджигает.
Togepi Eggman теряет 6 здоровья.

Togepi Eggman поджигает.
Entei Doge теряет 3 здоровья.

Entei Doge зловеще смотрит.
Togepi Eggman уменьшает защиту.

Togepi Eggman поджигает.
Entei Doge теряет 4 здоровья.

Entei Doge поджигает.
Togepi Eggman теряет 6 здоровья.

Togepi Eggman решил прилечь поспать.
Togepi Eggman засыпает

Entei Doge зловеще смотрит.
Togepi Eggman уменьшает защиту.

Togepi Eggman поджигает.
Entei Doge теряет 5 здоровья.
Entei Doge теряет сознание.
Cofagrigus Prizrachniy Gonchik из команды черных вступает в бой!
Togepi Eggman поджигает.
Cofagrigus Prizrachniy Gonchik теряет 7 здоровья.

Cofagrigus Prizrachniy Gonchik становится огненным призраком.
Togepi Eggman теряет 2 здоровья.
Togepi Eggman воспламеняется

Togepi Eggman теряет 1 здоровья.
Togepi Eggman решил прилечь поспать.
Togepi Eggman засыпает

Cofagrigus Prizrachniy Gonchik становится наглым.
Togepi Eggman увеличивает атаку.

Togepi Eggman растерянно попадает по себе.
```

Togepi Eggman теряет 3 здоровья.

Cofagrigus Prizrachniy Gonchik пугает своим страшным лицом.
Togepi Eggman уменьшает скорость.

Togepi Eggman решил прилечь поспать.
Togepi Eggman засыпает

Cofagrigus Prizrachniy Gonchik пугает своим страшным лицом.
Togepi Eggman уменьшает скорость.

Togepi Eggman решил прилечь поспать.
Togepi Eggman засыпает

Cofagrigus Prizrachniy Gonchik пугает своим страшным лицом.
Togepi Eggman уменьшает скорость.

Togepi Eggman поджигает.
Критический удар!
Cofagrigus Prizrachniy Gonchik теряет 15 здоровья.
Cofagrigus Prizrachniy Gonchik теряет сознание.
Yamask Casper the Ghost из команды черных вступает в бой!
Togepi Eggman растерянно попадает по себе.
Togepi Eggman теряет 3 здоровья.

Yamask Casper the Ghost промахивается

Togepi Eggman растерянно попадает по себе.
Togepi Eggman теряет 3 здоровья.

Yamask Casper the Ghost становится огненным призраком.
Togepi Eggman теряет 2 здоровья.
Togepi Eggman воспламеняется

Togepi Eggman теряет 1 здоровья.
Togepi Eggman поджигает.
Критический удар!
Yamask Casper the Ghost теряет 8 здоровья.

Yamask Casper the Ghost становится огненным призраком.
Togepi Eggman теряет 4 здоровья.

Togepi Eggman теряет 1 здоровья.
Togepi Eggman решил прилечь поспать.
Togepi Eggman засыпает

Yamask Casper the Ghost становится огненным призраком.
Togepi Eggman теряет 3 здоровья.
Togepi Eggman воспламеняется

Togepi Eggman теряет 1 здоровья.
Yamask Casper the Ghost становится наглым.
Togepi Eggman увеличивает атаку.

Togepi Eggman поджигает.
Yamask Casper the Ghost теряет 7 здоровья.

Togepi Eggman теряет 1 здоровья.
Yamask Casper the Ghost становится огненным призраком.
Togepi Eggman теряет 2 здоровья.

Togepi Eggman решил прилечь поспать.
Togepi Eggman засыпает

Yamask Casper the Ghost становится наглым.
Togepi Eggman увеличивает атаку.

Togepi Eggman поджигает.
Yamask Casper the Ghost теряет 7 здоровья.
Yamask Casper the Ghost теряет сознание.
В команде черных не осталось покемонов.
Команда фиолетовых побеждает в этом бою!

Заключение

В результате выполнения данной лабораторной работы были изучены основы ООП в языке Java. Изучены классы, их методы и конструкторы. Применена концепция наследования: конкретные виды покемонов наследовались от общего абстрактного класса, похожим образом наследовались и атаки. Также использовалось переопределение методов классами наследниками. После создания всей модели из классов было применено создание экземпляров объектов. При этом использовался принцип полиморфизма, например, когда покемонов разных видов можно было добавить в коллекцию из объектов класса `Pokemon`, наследниками которого были эти виды.