

Федеральное государственное автономное образовательное учреждение
высшего образования «Национальный исследовательский университет
ИТМО»

ФАКУЛЬТЕТ СИСТЕМ УПРАВЛЕНИЯ И РОБОТОТЕХНИКИ

Лабораторная работа №2

по дисциплине
“Программирование”

Вариант № 8428661

Выполнил:
Студент группы R3137
Юманов Михаил Алексеевич

Преподаватель:
Иманзаде Фахри Рашидович



Содержание

Задание.....	3
Диаграмма классов реализованной объектной модели.....	4
Исходный код программы.....	5-14
Результаты работы программы.....	15-16
Вывод.....	17

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](#) (обновлён 9.10.2018, исправлен баг с добавлением атак и кодировкой). Документация в формате javadoc - [здесь](#).

Введите вариант: 8428661

Ваши покемоны:


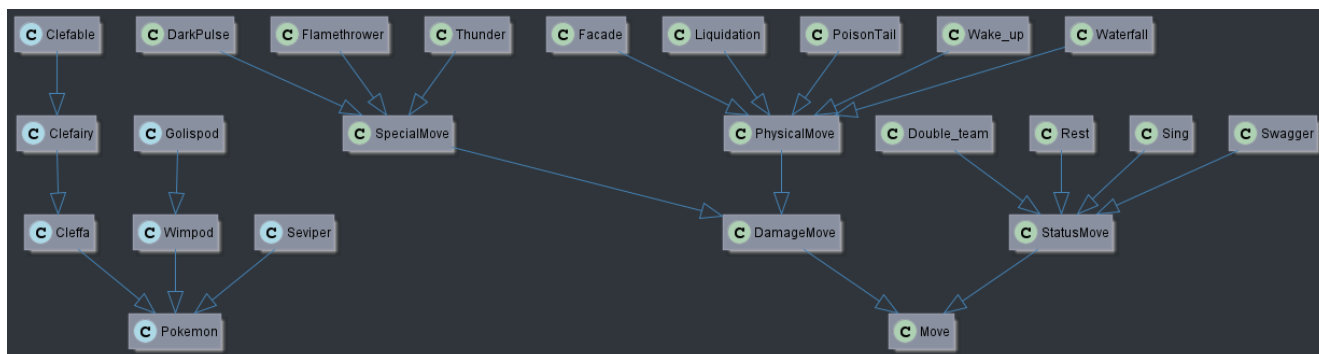
Seviper	Wimpod	Golisopod	Cleffa	Clefairy	Clefable
					
Атаки: <ul style="list-style-type: none">✓ Poison Tail✓ Swagger✓ Rest✓ Dark Pulse	Атаки: <ul style="list-style-type: none">✓ Facade✓ Waterfall✓ Double Team	Атаки: <ul style="list-style-type: none">✓ Facade✓ Waterfall✓ Double Team✓ Liquidation	Атаки: <ul style="list-style-type: none">✓ Flamethrower✓ Sing	Атаки: <ul style="list-style-type: none">✓ Flamethrower✓ Sing✓ Wake-Up Slap	Атаки: <ul style="list-style-type: none">✓ Flamethrower✓ Sing✓ Wake-Up Slap✓ Thunder

Диаграмма классов реализованной объектной модели:



Исходный код программы:

(<https://github.com/Mikhaillum/Pokemon>)

Main:

```
package ru.ifmo.pokemon.task;

import ru.ifmo.se.pokemon.Battle;
import ru.ifmo.se.pokemon.Pokemon;

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

        Cleffa cleffa = new Cleffa( name: "John", level: 50);
        Clefairy clefairy = new Clefairy( name: "Brian", level: 50);
        Clefable clefable = new Clefable( name: "Sara", level: 50);
        Wimpod wimpod = new Wimpod( name: "Alex", level: 50);
        Golispod golispod = new Golispod( name: "Andrew", level: 50);
        Seviper sevipser = new Seviper( name: "Arnold", level: 50);

        b.addAlly(golispod);
        b.addAlly(wimpod);
        b.addAlly(clefairy);
        b.addFoe(clefable);
        b.addFoe(sevipser);
        b.addFoe(cleffa);

        b.go();
    }
}
```

Покемоны:

```
package ru.ifmo.pokemon.task;

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

public class Cleffa extends Pokemon {

    public Cleffa(String name, int level){
        super(name, level);
        setType(Type.FAIRY);
        setMove(new Flamethrower(), new Sing());

        int HP = 50;
        int attack = 25;
        int defence = 28;
        int special_attack = 45;
        int special_defence = 55;
        int speed = 15;
        super.setStats(HP, attack, defence, special_attack, special_defence, speed);
    }
}
```

```
1 package ru.ifmo.pokemon.task;
2
3 import ru.ifmo.se.pokemon.Type;
4
5 public class Clefairy extends Cleffa{
6
7     public Clefairy(String name, int level) {
8         super(name, level);
9
10        addMove(new Wake_up());
11
12        int HP = 70;
13        int attack = 45;
14        int defence = 48;
15        int special_attack = 60;
16        int special_defence = 65;
17        int speed = 35;
18        super.setStats(HP, attack, defence, special_attack, special_defence, speed);
19    }
20 }
21
```

```

1  package ru.ifmo.pokemon.task;
2
3  import ru.ifmo.se.pokemon.Type;
4
5  public class Clefable extends Clefairy{
6      public Clefable(String name, int level) {
7          super(name, level);
8
9          addMove(new Thunder());
10
11          int HP = 95;
12          int attack = 70;
13          int defence = 73;
14          int special_attack = 95;
15          int special_defence = 90;
16          int speed = 60;
17          super.setStats(HP, attack, defence, special_attack, special_defence, speed);
18      }
19  }
20

```

```

1  package ru.ifmo.pokemon.task;
2
3  import ru.ifmo.se.pokemon.Pokemon;
4  import ru.ifmo.se.pokemon.Type;
5
6  public class Wimpod extends Pokemon {
7
8      public Wimpod(String name, int level){
9          super(name, level);
10         super.setType(Type.BUG, Type.WATER);
11
12         setMove(new Facade(), new Waterfall(), new Double_team());
13
14         int HP = 25;
15         int attack = 35;
16         int defence = 40;
17         int special_attack = 20;
18         int special_defence = 30;
19         int speed = 80;
20         super.setStats(HP, attack, defence, special_attack, special_defence, speed);
21     }
22 }
23

```

```

1  package ru.ifmo.pokemon.task;
2
3  import ru.ifmo.se.pokemon.Pokemon;
4  import ru.ifmo.se.pokemon.Type;
5
6  public class Golispod extends Wimpod {
7
8      public Golispod(String name, int level){
9          super(name, level);
10
11          addMove(new Liquidation());
12
13          int HP = 75;
14          int attack = 125;
15          int defence = 140;
16          int special_attack = 60;
17          int special_defence = 90;
18          int speed = 40;
19          super.setStats(HP, attack, defence, special_attack, special_defence, speed);
20      }
21  }
22

```

```

1  package ru.ifmo.pokemon.task;
2
3  import ru.ifmo.se.pokemon.Pokemon;
4  import ru.ifmo.se.pokemon.Type;
5
6
7  public class Seviper extends Pokemon {
8      public Seviper(String name, int level){
9          super(name, level);
10         super.addType(Type.POISON);
11
12         setMove(new PoisonTail(), new Swagger(), new Rest(), new DarkPulse());
13
14
15         int HP = 73;
16         int attack = 100;
17         int defence = 60;
18         int special_attack = 100;
19         int special_defence = 60;
20         int speed = 65;
21         super.setStats(HP, attack, defence, special_attack, special_defence, speed);
22     }
23 }
24

```


Атаки:

```
1 package ru.ifmo.pokemon.task;
2
3 import ru.ifmo.se.pokemon.Effect;
4 import ru.ifmo.se.pokemon.Pokemon;
5 import ru.ifmo.se.pokemon.SpecialMove;
6 import ru.ifmo.se.pokemon.Type;
7
8 public class DarkPulse extends SpecialMove {
9
10     public DarkPulse(){
11         super(Type.DARK, 80, 100);
12     }
13
14     @Override
15     protected String describe() {
16         return "запугивает оппонента";
17     }
18
19     @Override
20     protected void applyOppEffects(Pokemon p){
21         double rand = Math.random();
22         if (rand <= 0.2){
23             Effect.flinch(p);
24         }
25     }
26 }
27
```

```
1 package ru.ifmo.pokemon.task;
2
3 import ru.ifmo.se.pokemon.*;
4
5 public class Facade extends PhysicalMove {
6
7     public Facade(){
8         super(Type.NORMAL, 70, 100);
9     }
10
11     @Override
12     protected String describe() { return "в ярости наносит урон"; }
13
14     @Override
15     protected double calcCriticalHit(Pokemon att, Pokemon def){
16         double probability;
17         if (att.getCondition() == Status.POISON || att.getCondition() == Status.BURN || att.getCondition() == Status.PARALYZE)
18             probability = 1;
19         else
20             probability = att.getStat(Stat.SPEED) / 512;
21         double rand = Math.random();
22         if (rand < probability)
23             return 2;
24         return 1;
25     }
26 }
27
28
29
```

```

1 package ru.ifmo.pokemon.task;
2
3 import ru.ifmo.se.pokemon.Pokemon;
4 import ru.ifmo.se.pokemon.Stat;
5 import ru.ifmo.se.pokemon.StatusMove;
6 import ru.ifmo.se.pokemon.Type;
7
8
9 public class Double_team extends StatusMove{
10
11     public Double_team() { super(Type.NORMAL, v: 0, v1: 100); }
12
13
14     @Override
15     protected boolean checkAccuracy(Pokemon att, Pokemon def) { return true; }
16
17
18     @Override
19     protected String describe(){
20         return "потренировался в уклонении";
21     }
22
23     @Override
24     protected void applySelfEffects(Pokemon p) { p.setMod(Stat.EVASION, i: 1); }
25
26 }
27

```

```

1 package ru.ifmo.pokemon.task;
2
3 import ru.ifmo.se.pokemon.Effect;
4 import ru.ifmo.se.pokemon.Pokemon;
5 import ru.ifmo.se.pokemon.SpecialMove;
6 import ru.ifmo.se.pokemon.Type;
7
8 public class Flamethrower extends SpecialMove {
9
10     public Flamethrower() { super(Type.FIRE, v: 90, v1: 100); }
11
12
13     @Override
14     protected String describe() { return "кидает флэербол"; }
15
16
17     @Override
18     protected void applyOppEffects(Pokemon p){
19         double rand = Math.random();
20         if (rand <= 0.1){
21             Effect.burn(p);
22         }
23     }
24
25 }
26
27

```

```

1  package ru.ifmo.pokemon.task;
2
3  import ru.ifmo.se.pokemon.PhysicalMove;
4  import ru.ifmo.se.pokemon.Pokemon;
5  import ru.ifmo.se.pokemon.Stat;
6  import ru.ifmo.se.pokemon.Type;
7
8  public class Liquidation extends PhysicalMove {
9
10     public Liquidation() { super(Type.WATER, v: 85, v1: 100); }
11
12
13
14     @Override
15     protected String describe() { return "ликвидировал соперника"; }
16
17
18
19     @Override
20     protected void applyOppEffects(Pokemon p){
21         double rand = Math.random();
22         if (rand <= 0.2)
23             p.setMod(Stat.DEFENSE, i: -1);
24     }
25 }
26
27

```

```

1  package ru.ifmo.pokemon.task;
2
3  import ru.ifmo.se.pokemon.*;
4
5  public class PoisonTail extends PhysicalMove {
6
7     public PoisonTail() { super(Type.POISON, v: 50, v1: 100); }
8
9
10
11     @Override
12     protected String describe() { return "ударил ядовитым хвостиком"; }
13
14
15
16     @Override
17     protected double calcCriticalHit(Pokemon att, Pokemon def){
18         double probability = 3 * att.getStat(Stat.SPEED) / 512;
19         double rand = Math.random();
20         if (rand < probability)
21             return 2;
22         return 1;
23     }
24
25     @Override
26     protected void applyOppEffects(Pokemon p){
27         double rand = Math.random();
28         if (rand <= 0.1){
29             Effect.poison(p);
30         }
31     }
32 }
33
34

```

```

1      package ru.ifmo.pokemon.task;
2
3      import ru.ifmo.se.pokemon.*;
4
5      class Rest extends StatusMove {
6
7          public Rest() { super(Type.PSYCHIC, v: 0, v1: 100); }
8
9
10
11         @Override
12         protected String describe() {
13             return "устал и прилѣг отдохнуть";
14         }
15
16         @Override
17         protected void applySelfEffects(Pokemon p) {
18             Effect e = new Effect().condition(Status.SLEEP).turns(2);
19             p.addEffect(e);
20             p.setMod(Stat.HP, i: +6);
21         }
22     }
23

```

```

1      package ru.ifmo.pokemon.task;
2
3      import ru.ifmo.se.pokemon.Effect;
4      import ru.ifmo.se.pokemon.Pokemon;
5      import ru.ifmo.se.pokemon.StatusMove;
6      import ru.ifmo.se.pokemon.Type;
7
8      public class Sing extends StatusMove {
9
10         public Sing(){
11             super(Type.NORMAL, v: 0, v1: 55);
12         }
13
14         @Override
15         protected String describe() { return "усыпляет оппонента"; }
16
17
18         @Override
19         protected void applyOppEffects(Pokemon p){
20             Effect.sleep(p);
21         }
22     }
23
24
25
26

```

```

1      package ru.ifmo.pokemon.task;
2
3      import ru.ifmo.se.pokemon.*;
4
5      public class Swagger extends StatusMove {
6
7          public Swagger() { super(Type.NORMAL, v: 0, v1: 85); }
8
9          @Override
10         protected String describe() { return "начал вилять хвостом"; }
11
12         @Override
13         protected void applyOppEffects(Pokemon p){
14             p.setMod(Stat.ATTACK, i: 2);
15             Effect.confuse(p);
16         }
17     }
18 }
19
20
21
22
23

```

```

1      package ru.ifmo.pokemon.task;
2
3      import ru.ifmo.se.pokemon.Effect;
4      import ru.ifmo.se.pokemon.Pokemon;
5      import ru.ifmo.se.pokemon.SpecialMove;
6      import ru.ifmo.se.pokemon.Type;
7
8
9      public class Thunder extends SpecialMove {
10
11         public Thunder() { super(Type.ELECTRIC, v: 110, v1: 70); }
12
13         @Override
14         protected String describe() { return "бьет молнией"; }
15
16         @Override
17         protected void applyOppEffects(Pokemon p){
18             double rand = Math.random();
19             if (rand <= 0.3)
20                 Effect.paralyze(p);
21         }
22     }
23
24
25
26
27

```

```

1 package ru.ifmo.pokemon.task;
2
3 import ru.ifmo.se.pokemon.*;
4
5 public class Wake_up extends PhysicalMove {
6
7     public Wake_up() { super(Type.FIGHTING, v: 70, v1: 100); }
8
9
10
11     @Override
12     protected String describe() { return "даёт пощечину"; }
13
14
15
16     @Override
17     protected void applyOppEffects(Pokemon p){
18         if (p.getCondition() == Status.SLEEP){
19             p.restore();
20         }
21     }
22
23
24     @Override
25     protected double calcCriticalHit(Pokemon att, Pokemon def){
26         double probability;
27         if (def.getCondition() == Status.SLEEP)
28             probability = 1;
29         else
30             probability = att.getStat(Stat.SPEED) / 512;
31         double rand = Math.random();
32         if (rand < probability)
33             return 2;
34         return 1;
35     }
36 }
37

```

```

1 package ru.ifmo.pokemon.task;
2
3 import ru.ifmo.se.pokemon.Effect;
4 import ru.ifmo.se.pokemon.PhysicalMove;
5 import ru.ifmo.se.pokemon.Pokemon;
6 import ru.ifmo.se.pokemon.Type;
7
8 public class Waterfall extends PhysicalMove {
9
10     public Waterfall() { super(Type.WATER, v: 80, v1: 100); }
11
12
13
14     @Override
15     protected String describe() { return "заливает водой"; }
16
17
18
19     @Override
20     protected void applyOppEffects(Pokemon p){
21         double rand = Math.random();
22         if (rand <= 0.2)
23             Effect.flinch(p);
24     }
25 }
26

```

Результаты работы программы:

Golispod Andrew из команды желтых вступает в бой!
Clefable Sara из команды фиолетовых вступает в бой!
Clefable Sara даёт пощечину.
Golispod Andrew теряет 3 здоровья.

Golispod Andrew в ярости наносит урон.
Clefable Sara теряет 6 здоровья.

Clefable Sara кидает фаербол.
Golispod Andrew теряет 5 здоровья.

Golispod Andrew в ярости наносит урон.
Clefable Sara теряет 6 здоровья.

Clefable Sara даёт пощечину.
Golispod Andrew теряет 3 здоровья.

Golispod Andrew ликвидировал соперника.
Clefable Sara теряет 6 здоровья.
Clefable Sara уменьшает защиту.
Clefable Sara теряет сознание.
Seviper Arnold из команды фиолетовых вступает в бой!
Seviper Arnold ударил ядовитым хвостиком.
Golispod Andrew теряет 4 здоровья.
Golispod Andrew отравлен
Golispod Andrew теряет сознание.
Wimpod Alex из команды желтых вступает в бой!
Wimpod Alex заливает водой.
Seviper Arnold теряет 5 здоровья.

Seviper Arnold начал вилять хвостом.
Wimpod Alex увеличивает атаку.

Wimpod Alex в ярости наносит урон.
Seviper Arnold теряет 5 здоровья.

Seviper Arnold запугивает оппонента.
Wimpod Alex теряет 7 здоровья.

Wimpod Alex потренировался в уклонении.
Wimpod Alex увеличивает уклоняемость.

Seviper Arnold ударил ядовитым хвостиком.
Wimpod Alex теряет 8 здоровья.
Wimpod Alex теряет сознание.
Cleffairy Brian из команды желтых вступает в бой!
Seviper Arnold начал вилять хвостом.
Cleffairy Brian увеличивает атаку.

Cleffairy Brian даёт пощечину.
Seviper Arnold теряет 2 здоровья.

Seviper Arnold начал вилять хвостом.
Cleffairy Brian увеличивает атаку.

Cleffairy Brian даёт пощечину.
Seviper Arnold теряет 2 здоровья.
Seviper Arnold теряет сознание.
Cleffa John из команды фиолетовых вступает в бой!
Cleffairy Brian растерянно попадает по себе.
Cleffairy Brian теряет 4 здоровья.

Cleffa John усыпляет оппонента.
Cleffairy Brian засыпает

Cleffa John усыпляет оппонента.

Cleffairy Brian даёт пощечину.
Cleffa John теряет 3 здоровья.

Cleffa John кидает фаербол.
Cleffairy Brian теряет 5 здоровья.

Cleffairy Brian растерянно попадает по себе.
Cleffairy Brian теряет 5 здоровья.
Cleffairy Brian теряет сознание.
В команде желтых не осталось покемонов.
Команда фиолетовых побеждает в этом бою!

Выводы:

В ходе данной лабораторной работы я:

- 1) Познакомился с принципами ООП на практике
- 2) Научился подключать к своей программе внешние файлы
- 3) Научился компилировать и собирать jar-файлы для большого количества классов, создав манифест с подключенным внешним jar-файлом
- 4) Узнал о том, что такое пакеты в Java и воспользовался ими на практике
- 5) Научился создавать базовые UML-диаграммы классов
- 6) Научился читать и понимать документацию
- 7) Научился переопределять методы и наследоваться от классов