Санкт-Петербургский Национальный Исследовательский Университет
Информационных Технологий, Механики и Оптики

Факультет Компьютерных Технологий и Управления

Лабораторная работа №2

по дисциплине
«Программирование»
Вариант №611

Выполнил:
студент группы R3136
Обудов Владислав Антонович
Преподаватель:
Гаврилов Антон Валерьевич

Санкт-Петербург
2019 г.

1. Текст задания

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.

1.1 Комментарии

Цель работы: на простом примере разобраться с основными концепциями ООП и научиться использовать их в программах.

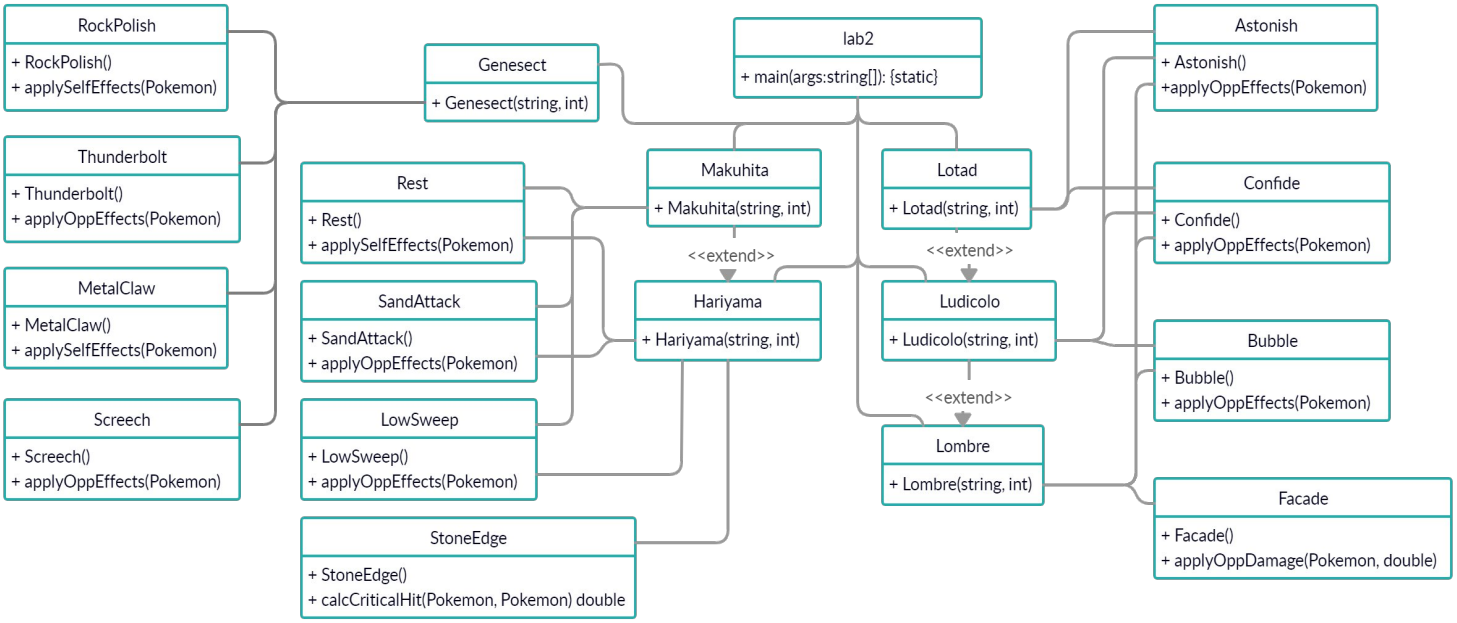
Что надо сделать (краткое описание):

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.

```
Battle b = new Battle();
Pokemon p1 = new Pokemon("Чужой", 1);
Pokemon p2 = new Pokemon("Хищник", 1);
b.addAlly(p1);
b.addFoe(p2);
b.go();
```

4. 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.
5. 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.
6. 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.
7. Доделать все необходимые атаки и всех покемонов, распределить покемонов по командам, запустить сражение.

2. Диаграмма классов



2.1 Покемоны

3. Исходный код программы

lab2.java

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

public class lab2 {
  public static void main(String[] args) {
    Battle b = new Battle();
    Genesect _genesect = new Genesect("Ivan", 1);
    Makuhita _makuhita = new Makuhita("Petr", 1);
    Hariyama _hariyama = new Hariyama("Egor", 1);
    Lotad _lotad = new Lotad("Dima", 1);
    Lombre _lombre = new Lombre("Senya", 1);
    Ludicolo _ludicolo = new Ludicolo("Stas", 1);
    b.addAlly(_genesect);
    b.addAlly(_hariyama);
    b.addAlly(_lombre);
    b.addFoe(_makuhita);
    b.addFoe(_lotad);
    b.addFoe(_ludicolo);
    b.go();
  }
}
```

Genesect.java

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

public class Genesect extends Pokemon {
	public Genesect(String name, int level) {
		super(name, level);
		setStats(71, 120, 95, 120, 95, 99);
		setType(Type.BUG, Type.STEEL);
		RockPolish _rockpolish = new RockPolish();
		Thunderbolt _thunderbolt = new Thunderbolt();
		Screech _screech = new Screech();
		MetalClaw _metalclaw = new MetalClaw();
		setMove(_rockpolish, _thunderbolt, _screech, _metalclaw);
	}
}
```

Makuhita.java

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

public class Makuhita extends Pokemon {
	public Makuhita(String name, int level) {
		super(name, level);
		setStats(72, 60, 30, 20, 30, 25);
		setType(Type.FIGHTING);
		Rest _rest = new Rest();
		SandAttack _sandattack = new SandAttack();
		LowSweep _lowsweep = new LowSweep();
```

```
                setMove(_rest, _sandattack, _lowsweep);
        }
}
```

Hariyama.java

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

public class Hariyama extends Makuhita {
        public Hariyama(String name, int level) {
                super(name, level);
                setStats(144, 120, 60, 40, 60 ,50);
                setType(Type.FIGHTING);
                Rest _rest = new Rest();
                SandAttack _sandattack = new SandAttack();
                LowSweep _lowsweep = new LowSweep();
                StoneEdge _stoneedge = new StoneEdge();
                setMove(_rest, _sandattack, _lowsweep, _stoneedge);
        }
}
```

Lotad.java

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

public class Lotad extends Pokemon {
        public Lotad(String name, int level) {
                super(name, level);
                setStats(40, 30, 30, 40, 50, 30);
                setType(Type.WATER, Type.GRASS);
                Astonish _astonish = new Astonish();
                Confide _confide = new Confide();
                setMove(_astonish, _confide);
        }
}
```

Lombre.java

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

public class Lombre extends Lotad {
        public Lombre(String name, int level) {
                super(name, level);
                setStats(60, 50, 50, 60, 70, 50);
                setType(Type.WATER, Type.GRASS);
                Astonish _astonish = new Astonish();
                Confide _confide = new Confide();
                Bubble _bubble = new Bubble();
                setMove(_astonish, _confide, _bubble);
        }
}
```

Ludicolo.java

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

public class Ludicolo extends Lombre {
```

```
	public Ludicolo(String name, int level) {
		super(name, level);
		setStats(80, 70, 70, 90, 100, 70);
		setType(Type.WATER, Type.GRASS);
		Astonish _astonish = new Astonish();
		Confide _confide = new Confide();
		Bubble _bubble = new Bubble();
		Facade _facade = new Facade();
		setMove(_astonish, _confide, _bubble, _facade);
	}
}
```

RockPolish.java

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

public class RockPolish extends StatusMove {
	protected RockPolish() {
		super(Type.NORMAL, 0, 0);
	}
	@Override
	protected void applySelfEffects(Pokemon p) {
		p.setMod(Stat.SPEED, +2);
	}
}
```

Thunderbolt.java

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

public class Thunderbolt extends SpecialMove {
	protected Thunderbolt() {
		super(Type.ELECTRIC, 95, 100);
	}
	@Override
	protected void applyOppEffects(Pokemon p) {
		if (Math.random() <= 0.1 && !p.hasType(Type.ELECTRIC)) Effect.paralyze(p);
	}
}
```

Screech.java

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

public class Screech extends StatusMove {
	protected Screech() {
		super(Type.NORMAL, 0, 85);
	}
	@Override
	protected void applyOppEffects(Pokemon p) {
		p.setMod(Stat.DEFENSE, -2);
	}
}
```

MetalClaw.java

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

```
public class MetalClaw extends PhysicalMove {
	protected MetalClaw() {
		super(Type.STEEL, 50, 95);
	}
	@Override
	protected void applySelfEffects(Pokemon p) {
		if (Math.random() <= 0.1) {
			p.setMod(Stat.ATTACK, +1);
		}
	}
}
```

Rest.java

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

public class Rest extends StatusMove {
	protected Rest() {
		super(Type.PSYCHIC, 0, 0);
	}
	@Override
	protected void applySelfEffects(Pokemon p) {
		Effect e = new Effect().turns(2);
		e.condition(Status.SLEEP);
		p.setCondition(e);
		p.restore();
	}
}
```

SandAttack.java

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

public class SandAttack extends StatusMove {
	protected SandAttack() {
		super(Type.GROUND, 0, 100);
	}
	@Override
	protected void applyOppEffects(Pokemon p) {
		p.setMod(Stat.ACCURACY, -1);
	}
}
```

LowSweep.java

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

public class LowSweep extends PhysicalMove {
	protected LowSweep() {
		super(Type.FIGHTING, 65, 100);
	}
	@Override
	protected void applyOppEffects(Pokemon p) {
		p.setMod(Stat.SPEED, -1);
```

```
	}
}
```

StoneEdge.java

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

public class StoneEdge extends PhysicalMove {
	protected StoneEdge() {
		super(Type.ROCK, 100, 80);
	}
	@Override
	protected double calcCriticalHit(Pokemon att, Pokemon def) {
		if (Math.random() < att.getStat(Stat.SPEED) / 512 * 8) {
			System.out.println("Critical hit!");
			return 2.0;
		} else {
			return 1.0;
		}
	}
}
```

Astonish.java

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

public class Astonish extends PhysicalMove {
	protected Astonish() {
		super(Type.GHOST, 30, 100);
	}
	@Override
	protected void applyOppEffects(Pokemon p) {
		if (Math.random() <= 0.3) Effect.flinch(p);
	}
}
```

Confide.java

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

public class Confide extends StatusMove {
	protected Confide() {
		super(Type.NORMAL, 0, 0);
	}
	@Override
	protected void applyOppEffects(Pokemon p) {
		p.setMod(Stat.SPECIAL_ATTACK, -1);
	}
}
```

Bubble.java

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

public class Bubble extends SpecialMove {
	protected Bubble() {
		super(Type.WATER, 20, 100);
```

```
	}
	@Override
	protected void applyOppEffects(Pokemon p) {
		if (Math.random() <= 0.1) p.setMod(Stat.SPEED, -1);
	}
}
```

Facade.java

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

public class Facade extends PhysicalMove {
	protected Facade() {
		super(Type.NORMAL, 70, 100);
	}
	@Override
	protected void applyOppDamage (Pokemon p, double dmg) {
		Status pCondition = p.getCondition();
		if (pCondition.equals(Status.BURN) || pCondition.equals(Status.POISON) ||
pCondition.equals(Status.PARALYZE)) {
			p.setMod(Stat.HP, (int) Math.round(dmg) * 2);
			System.out.println("Critical hit!");
		}
	}
}
```

4. Результат работы программы

Genesect Ivan из команды фиолетовых вступает в бой!

Makuhita Petr из команды красных вступает в бой!

Genesect Ivan атакует.

Makuhita Petr теряет 4 здоровья.

Makuhita Petr атакует.

Genesect Ivan теряет 4 здоровья.

Genesect Ivan уменьшает скорость.

Genesect Ivan атакует.

Makuhita Petr теряет 5 здоровья.

Makuhita Petr промахивается

Genesect Ivan атакует.

Makuhita Petr теряет 7 здоровья.

Makuhita Petr теряет сознание.

Lotad Dima из команды красных вступает в бой!

Genesect Ivan атакует.

Lotad Dima теряет 3 здоровья.

Lotad Dima промахивается

Genesect Ivan атакует.

Lotad Dima уменьшает защиту.

Lotad Dima промахивается

Genesect Ivan атакует.

Lotad Dima уменьшает защиту.

Lotad Dima атакует.

Genesect Ivan теряет 4 здоровья.

Genesect Ivan атакует.

Lotad Dima теряет 3 здоровья.

Lotad Dima промахивается

Genesect Ivan атакует.

Lotad Dima теряет 4 здоровья.

Lotad Dima атакует.

Genesect Ivan теряет 3 здоровья.

Genesect Ivan промахивается

Lotad Dima атакует.

Genesect Ivan теряет 3 здоровья.

Genesect Ivan теряет сознание.

Hariyama Egor из команды фиолетовых вступает в бой!

Hariyama Egor промахивается

Lotad Dima атакует.

Hariyama Egor теряет 3 здоровья.

Hariyama Egor атакует.

Lotad Dima уменьшает точность.

Lotad Dima атакует.

Hariyama Egor теряет 4 здоровья.

Hariyama Egor промахивается

Lotad Dima промахивается

Hariyama Egor промахивается

Lotad Dima атакует.

Hariyama Egor теряет 4 здоровья.

Hariyama Egor атакует.

Lotad Dima теряет 8 здоровья.

Lotad Dima теряет сознание.

Ludicolo Stas из команды красных вступает в бой!

Ludicolo Stas атакует.

Hariyama Egor теряет 4 здоровья.

Hariyama Egor теряет сознание.

Lombre Senya из команды фиолетовых вступает в бой!

Ludicolo Stas промахивается

Lombre Senya атакует.
Ludicolo Stas теряет 4 здоровья.

Lombre Senya атакует.
Ludicolo Stas теряет 4 здоровья.

Lombre Senya промахивается

Lombre Senya атакует.
Ludicolo Stas теряет 1 здоровья.

Ludicolo Stas промахивается

Lombre Senya атакует.
Ludicolo Stas теряет 5 здоровья.
Ludicolo Stas теряет сознание.
В команде красных не осталось покемонов.
Команда фиолетовых побеждает в этом бою!

5. Вывод

Во время выполнения данной лабораторной работы я ознакомился с основными концепциями объектно-ориентированного программирования и научился использовать их в программе.