

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ «Санкт-Петербургский национальный исследовательский
университет ИТМО»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ
ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине

«ПРОГРАММИРОВАНИЕ»

Вариант № 98765

Выполнил:

Студент группы Р3116

Билошицкий Михаил Владимирович

Преподаватель:

Письмак Алексей Евгеньевич

Санкт-Петербург, 2022

Содержание

Задание.....	3
Исходный код программы	4
Результат работы программы	5
UML диаграмма классов	6
Вывод	7

Задание

Лабораторная работа #2

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.

```
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.

Введите вариант: 9876

Ваши покемоны:

 <p>Атаки:</p> <ul style="list-style-type: none">✓ Dark Pulse✓ Baby-Doll Eyes✓ Tackle✓ Bite	 <p>Атаки:</p> <ul style="list-style-type: none">✓ Sand Attack✓ Rock Polish✓ Ancient Power	 <p>Атаки:</p> <ul style="list-style-type: none">✓ Sand Attack✓ Rock Polish✓ Ancient Power✓ Night Slash	 <p>Атаки:</p> <ul style="list-style-type: none">✓ Rest✓ Swagger	 <p>Атаки:</p> <ul style="list-style-type: none">✓ Rest✓ Swagger✓ Wake-Up Slap	 <p>Атаки:</p> <ul style="list-style-type: none">✓ Rest✓ Swagger✓ Wake-Up Slap✓ Confide
--	--	--	---	--	--

Исходный код программы

Ссылка на GitHub с исходным кодом:

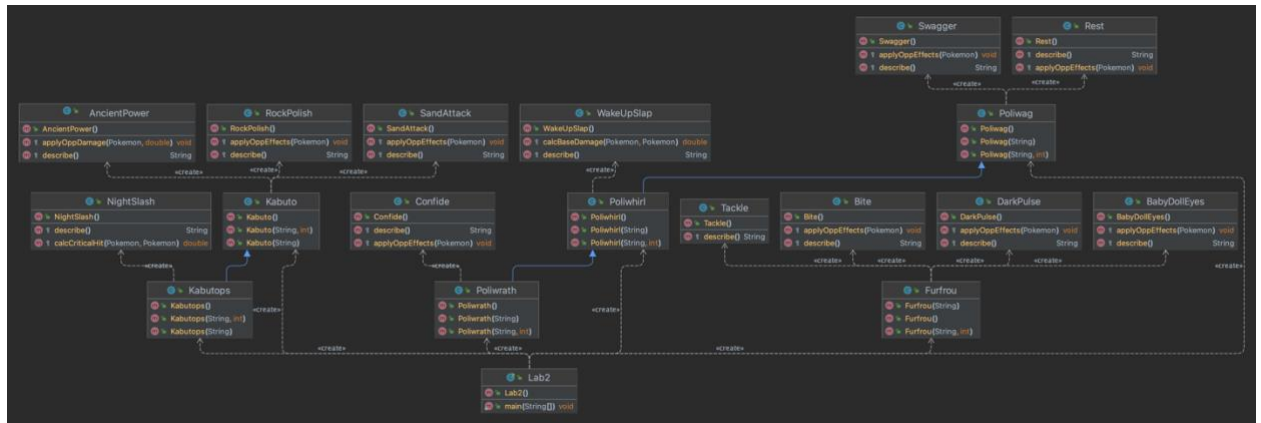
https://github.com/michael-bill/labs_programming_itmo/tree/main/Laba2

Результат работы программы

Результат работы программы находится в GitHub репозитории.

https://github.com/michael-bill/labs_programming_itmo/blob/main/Laba2/program_output.txt

YML диаграмма классов



Чтобы ознакомиться с ней поближе, ссылка на .png файл также прикреплена в GitHub репозитории.

https://github.com/michael-bill/labs_programming_itmo/blob/main/Laba2/YML%20Diagram.png

Вывод

Во время выполнения работы я ознакомился с принципами ООП в Java. Научился работать с внешними Jar библиотеками на практике. Научился создавать классы и наследовать их, описывать логику классов, переопределять методы и работать с документацией. Полученные знания понадобятся в процессе дальнейшего обучения.