Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №2

Вариант 310865

Выполнил:

Попов Кирилл Олегович

Группа P3108

Проверил:

Преподаватель программирования

Наумова Надежда Александрова

Содержание

[Задание 3](#_Toc180985953)

[UML-диаграмма классов 5](#_Toc180985954)

[Исходный код программы 5](#_Toc180985955)

[Результат работы программы 5](#_Toc180985956)

[Вывод 6](#_Toc180985957)

# 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](https://se.ifmo.ru/documents/10180/660917/Pokemon.jar/a7ce60af-6ee6-47d0-a95e-e5ed9a697bd2) (обновлен 9.10.2018, исправлен баг с добавлением атак и кодировкой). Документация в формате javadoc - [здесь](https://se.ifmo.ru/~tony/doc/).

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);

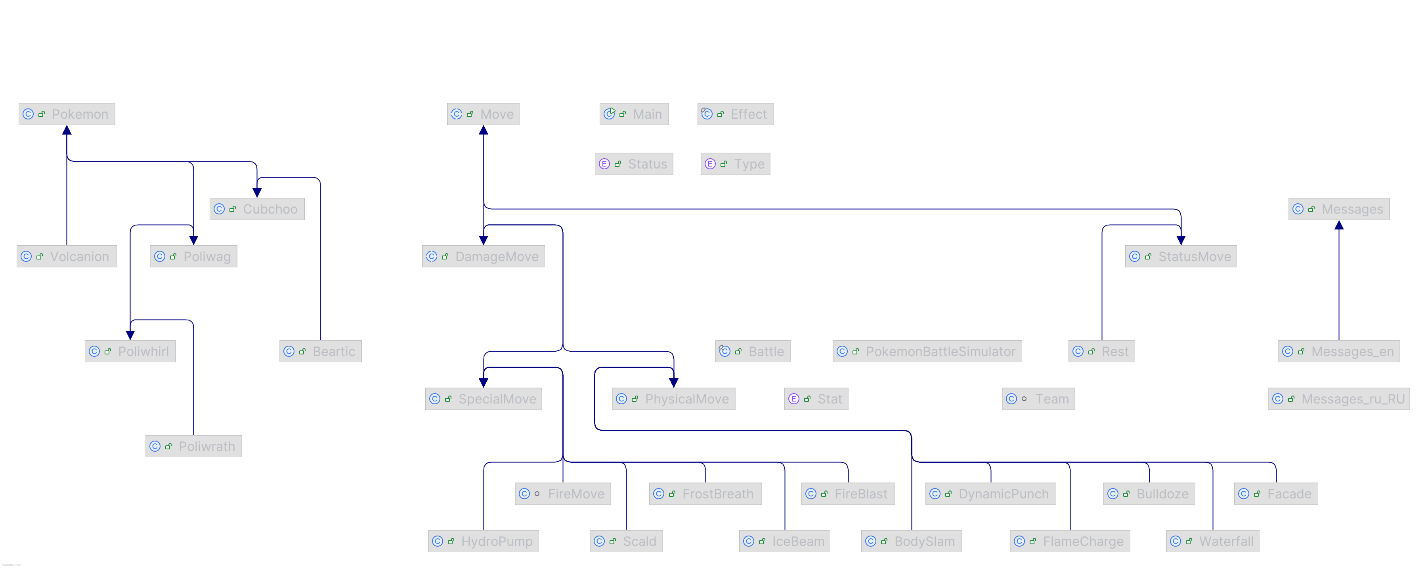
b.go();

1. 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.
2. 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.
3. 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.
4. Доделать все необходимые атаки и всех покемонов, распределить покемонов по командам, запустить сражение.

Изображение выглядит как текст, мультфильм, иллюстрация

Автоматически созданное описание

# UML-диаграмма классов



# Исходный код программы

Код программы представлен в репозитории: <https://github.com/Popandypalo/Proga_Lab_2>

# Результат работы программы

Beartic Косолапыч из команды зеленых вступает в бой!

Cubchoo Терпилыч из команды синих вступает в бой!

Beartic Косолапыч использует Bulldoze.

Cubchoo Терпилыч теряет 5 здоровья.

Cubchoo Терпилыч использует Facade.

Beartic Косолапыч теряет 6 здоровья.

Cubchoo Терпилыч использует Facade.

Beartic Косолапыч теряет 6 здоровья.

Beartic Косолапыч использует Bulldoze.

Cubchoo Терпилыч теряет 3 здоровья.

Cubchoo Терпилыч использует Facade.

Критический удар!

Beartic Косолапыч теряет 8 здоровья.

Beartic Косолапыч теряет сознание.

Volcanion Вулканчик из команды зеленых вступает в бой!

Cubchoo Терпилыч использует Facade.

Критический удар!

Volcanion Вулканчик теряет 8 здоровья.

Volcanion Вулканчик использует Fire Blast.

Cubchoo Терпилыч теряет 21 здоровья.

Cubchoo Терпилыч теряет сознание.

Poliwhirl Боксер из команды синих вступает в бой!

Poliwhirl Боксер использует Body Slam.

Volcanion Вулканчик теряет 6 здоровья.

Volcanion Вулканчик теряет сознание.

Poliwrath Медвежонок из команды зеленых вступает в бой!

Poliwhirl Боксер использует Body Slam, но парализовать противника не удалось.

Poliwrath Медвежонок теряет 6 здоровья.

Poliwrath Медвежонок использует Dynamic Punch.

Poliwhirl Боксер теряет 8 здоровья.

Poliwhirl Боксер использует Body Slam, но парализовать противника не удалось.

Poliwrath Медвежонок теряет 5 здоровья.

Poliwrath Медвежонок использует Dynamic Punch, но дополнительный эффект не сработал.

Poliwhirl Боксер теряет 8 здоровья.

Poliwhirl Боксер теряет сознание.

Poliwag Михалыч из команды синих вступает в бой!

Poliwag Михалыч промахивается

Poliwrath Медвежонок использует Dynamic Punch, но дополнительный эффект не сработал.

Poliwag Михалыч теряет 6 здоровья.

Poliwag Михалыч промахивается

Poliwrath Медвежонок использует Dynamic Punch, нанося дополнительный урон на 40 в течение 4 ходов.

Poliwag Михалыч теряет 10 здоровья.

Poliwag Михалыч теряет сознание.

В команде синих не осталось покемонов.

Команда зеленых побеждает в этом бою!

# Вывод

В ходе данной работы я научился подключать внешние jar файлы и использовать их. Научился работать с документацией, изучил объектно-ориентированный подход на языке Java. Научился работать с классами, конструкторами, полями и модификаторами доступа.