INF100 H17 Assignment 6

Due on Friday, November 17, 2017

Albin Severinson, Dag Haugland

Abstract

Nintendo reportedly made $569 million in the quarter after releasing the Pokemon Sun and Moon games. That is a lot of money. So today we are implementing Pokemon. In particular, we will write a class to represent a Pokemon and we will have them fighting each other. We implement the fighting part using instance methods. As usual the implementation is split over several sub problems, where each step builds upon the previous. We will only grade (pass/fail) the program for the final problem of the assignment. If you have questions specific to this assignment you should send them to me at `albin.severinson@uib.no`.

Submission

1. Add your program file to a `.zip` archive named `ex6_firstname_lastname.zip`. You do this by right clicking on the file ending with `.java` that contains your program code and choosing compress or add to archive (exactly what it is called varies by operating system and version). You should only add your final program, i.e., the one for problem 5. Remember that the file name must match the class name and that you therefore can not rename the `.java` file! Ask your group leader if you are having issues with this. Note that handing in the wrong file may cause you to fail the assignment!
2. If you complete the voluntary getting ahead part of the assignment, it should be handed in separately. If you complete this part you thus need to hand in two files (the `.java` file for problem 5 and the `.java` file for the getting ahead part). Put both files into the same `.zip` archive.
3. Log in to `https://mitt.uib.no` and press the box marked with "INF100" to enter the page for this course.
4. Press "Oppgåver" on the left. Next, select "Assignment 6".
5. Press "Lever oppgåve" and upload the `.zip` file you created in step 1.
6. We grade assignments in one week and semester assignments in two weeks.

General Guidelines

These are general guidelines that will (hopefully) help you maintain your sanity through your programming endeavours.

- Have your computer indent your code for you. You do this in DrJava by selecting all of your code, right clicking, and selecting `indent lines`.
- You should make a copy of all your code after completing each problem so that you can roll back to a previous version when you inadvertently break something. Even the developers at Google sometimes have to roll back.
- Set aside time to clean up your code. The number that gets bandied around on the internet is that the best programmers spend roughly half their time cleaning up and rewriting their code, i.e., you are halfway done when you get it working.
- You save time by making sure you properly understand your current program before moving on.
- You should probably be spending a significant part of your time on stackoverflow.
- Google is your friend when you want to figure out how a standard class or method works (`Scanner`, say). For example, searching for "java Scanner" gives you its Java docs page as the first result.
- When in doubt, look at the Google Java style guide to figure out how to write something.

Pokemon

It seems there is a lot of money in creating Pokemon games. So we are going to build one! We implement it using objects and object methods. We will only implement the most essential features. This means that the game will be text-based, have zero user input, and that its output should look like this:

```
Pikachu HP: (111/111) STR: 19 CHC: 12%
Oddish HP: (94/94) STR: 21 CHC: 11%

Pikachu attacks Oddish.
Oddish takes 27 damage and is left with 67/94 HP
Oddish attacks Pikachu.
Pikachu takes 9 damage and is left with 102/111 HP
Pikachu attacks Oddish.
Oddish takes 11 damage and is left with 56/94 HP
Oddish attacks Pikachu.
Pikachu takes 15 damage and is left with 87/111 HP
Pikachu attacks Oddish.
Oddish takes 13 damage and is left with 43/94 HP
Oddish attacks Pikachu.
Pikachu takes 21 damage and is left with 66/111 HP
Pikachu attacks Oddish.
Oddish takes 8 damage and is left with 35/94 HP
Critical hit!
Oddish takes 8 damage and is left with 27/94 HP
Oddish attacks Pikachu.
Pikachu takes 27 damage and is left with 39/111 HP
Pikachu attacks Oddish.
Oddish takes 14 damage and is left with 13/94 HP
Oddish attacks Pikachu.
Pikachu takes 12 damage and is left with 27/111 HP
Critical hit!
Pikachu takes 12 damage and is left with 15/111 HP
Pikachu attacks Oddish.
Oddish takes 6 damage and is left with 7/94 HP
Oddish attacks Pikachu.
Pikachu takes 26 damage and is left with 0/111 HP
Pikachu is defeated by Oddish.
```

Problem 1

Start by creating a class `Pokemon`. This class should have fields

```java
private String name;
private int healthPoints;
private int maxHealthPoints;
private int strength;
private double criticalChance;
private Random random;
```

The class constructor should look like this:

```java
public Pokemon(String name) {
    this.random = new Random();
    this.name = name;
    this.healthPoints =
        (int) Math.abs(Math.round(100 + 10 * random.nextGaussian()));
    this.maxHealthPoints = this.healthPoints;
    this.strength =
        (int) Math.abs(Math.round(20 + 10 * random.nextGaussian()));
    this.criticalChance = Math.abs(0.1 * random.nextGaussian());
}
```

Furthermore, the `toString()` method should return a string with the following format

```
name HP: (healthPoints/maxHealthPoints) STR: strength CHC: criticalChance
```

`criticalChance` should be written as a percentage chance with no decimal points written. For example, printing a `Pokemon` initialized with `name="Pikachu"` could print

```
Pikachu HP: (105/105) STR: 30 CHC: 11%
```

Problem 2

For this problem we are creating two helper methods that we will need for the rest of the assignment. Both methods should be instance methods of the `Pokemon` class. The first method should return the name of the `Pokemon`

```java
public String getName() {

}
```

The second should return a `boolean` indicating if the Pokemon is conscious or not. It is conscious if and only if its `healthPoints` is larger than 0. It is important to note that no Pokemon are killed in this game.

```java
public boolean isConscious() {

}
```

Problem 3

Write an instance method that inflicts damage on the Pokemon. The method should have signature

```java
public void damage(int damageTaken) {

}
```

This method should decrease the `healthPoints` of the Pokemon by `damageTaken`. The `healthPoints` should never be set to less than 0. If `damageTaken` is larger than `healthPoints`, `healthPoints` should be set equal to 0.

Furthermore, the method should print to the terminal with the following format

```
name takes damageTaken damage and is left with healthPoints/maxHealthPoints HP
```

For example, if our Pikachu takes damage it could look like this

```
Pikachu takes 21 damage and is left with 84/105 HP
```

Problem 4

Write an instance method that makes one Pokemon attack another. The method should have signature

```java
public void attack(Pokemon target) {
    int damageInflicted =
        (int) (this.strength + this.strength / 2 * random.nextGaussian());
    if (damageInflicted < 0) {
        damageInflicted = 0;
    }
    System.out.printf("%s attacks %s.%n", this.getName(), target.getName());

    // the rest of the method is up to you
}
```

The method should call the `damage` method of the `target` Pokemon with argument `damageInflicted`. Furthermore, the attack is a critical hit with probability `this.criticalChance`. Use `Math.random()` to generate a random number between 0 and 1. If this generated number is less than `this.criticalChance` the attack is critical. If the attack is critical, you should print this to the terminal and inflict the same amount of damage to the target again. See the Pokemon section for an example of this.

If the target Pokemon is no longer conscious after the attack, the following should be printed to the terminal

```
target.getName() is defeated by this.getName().
```

You need to use the `isConscious()` method of `target`. For example, if Pikachu is attacking and defeats an Oddish, the following should be printed

```
Oddish is defeated by Pikachu.
```

Problem 5

Finally, create a new file `Main.java` with a `main()` method in it. This method should create two Pokemon and have them battle until either Pokemon is no longer conscious. Specifically, the `main()` method should

1. Create two Pokemon with difference names.
2. Print both Pokemon to the terminal.
3. Use a `while` loop to have the Pokemon take turns attacking each other until either Pokemon falls unconscious. Remember that a Pokemon may only attack if it is still conscious.

When running the program you should see an output like the example in the Pokemon section.

Getting Ahead

We are well on the way to becoming Pokemon millionaires! For the getting ahead part you should implement the rest of the features needed for our game to compete with those released by Nintendo. Some suggestions on where to start are:

- Use a loop to allow battling several Pokemon consecutively without restarting the program.
- Allow the user to select between several Pokemon at the start of each battle.
- Implement experience and leveling. Each Pokemon should then collect experience and its strength and hit points should increase when enough experience is collected.

This part is meant as preparation for the coming course material and will not be graded. However, we will still look at it and comment it! Furthermore, you never have to catch up if you are always getting ahead. This part should be handed in separately. If you complete this part you thus need to hand in two sets of files (the `.java` files for problem 5 and the `.java` files for this part). Put both files into the same `.zip` archive.