Introduction

Hello Pokemon trainers! In assignment 5, you will implement a text-driven Pokemon Go simulator! You’re responsible for finishing different classes with inheritance and also the Assignment5.java to simulate the game. You are also required to come up with your own tests to make sure your code has the correct behavior as described in this writeup. You should complete this assignment individually. This is a long assignment, so start early!

Part 1: Get started

1. Make sure there is no problem with your Java coding environment. If there is any, review assignment 1 or come to the office hour before you start this assignment.
2. Review everything on and before lecture 10.
3. Download the starter code.
   If you work with your local machine, download it from Piazza -> Resources -> Homework -> Assignment5.java, Berry.java, Item.java, PalPokemon.java, Pokeball.java, Pokemon.java, and WildPokemon.java.
   Start terminal/command line and navigate to the directory that contains Assignment5.java, Berry.java, Item.java, PalPokemon.java, Pokeball.java, Pokemon.java, and WildPokemon.java.
   If you decide to use Linux Cloud, use the following command to copy the starter code to a new directory called HW5:

   ```
   $ cd ~
   $ mkdir HW5
   $ cp -a ../public/assignments/PA5 ./HW5
   ```

   Note: you will not be able to compile the start code because most of the methods are left unimplemented.

Overview:
In this assignment, you will implement a Pokemon Go simulator. In the game, there will be a class named Pokemon. There will be two subclasses of Pokemon class, which are WildPokemon and PalPokemon. Also, there will be a class called Item. Under the Item class, there will be two subclasses named Berry and Pokeball. These two items can help you to catch a wild pokemon to be your pal pokemon. You can use a Pokeball to catch, and use a berry to enhance the catch rate. You can only use up to one berry before you throw a pokeball. There will be different kinds of pokeballs and berries. Details will be explained later.  

**Implementation tip:** you should not change any existing data field or method signature in the starter code. You may add new fields or methods, if you want, but this is not necessary. As you are familiar with this course and our assignment format, we are trying to give you more succinct instructions. Observe the starter code and read the instructions below to make sure you understand what each field means before you start to implement.

**Part 2: Item.java**

First, you need to implement the object class called Item. This is a super class and it is very straightforward. The only field is `String name`, which is the item’s name. Complete all constructors and methods in this class. In the no-arg constructor, set `String name` to be “item”.

**Part 3: Pokeball.java and Berry.java**

Pokeball and Berry are two subclasses under Item. Complete all constructors and methods in those classes. For the no-arg constructors, set every field’s value to be 0. Remember, you cannot change the existing signature or the fields.

- `int performance` of Pokeball is the field that will do with the catch rate. Higher performance will increase catch rate more.
- `int patienceIncrement` of Berry is the field that will also help the catch rate. Higher patienceIncrement will increase catch rate more.
- `int speedDecrement` of Berry is the field that will also help the catch rate from another perspective. Higher speedDecrement will decrease the moving speed of a wild Pokemon to make an easier pokeball hit.

Don’t feel bad if you don’t 100% understand what those fields mean. You can finish the implementation of this part. Details will be explained later.
Part 4: Pokemon.java

Pokemon is another superclass in this assignment. It has three fields. For the field type, it's an attribute of a certain pokemon. For example, Pikachu is an electric type pokemon. Complete all constructors and methods in this class. In the no-arg constructor, set String name to be "Pokemon", String sound to be "sound", String type to be "unknown". Also it has a method speak that is already implemented. Use it properly.

Part 5: WildPokemon.java

WildPokemon is a subclass under Pokemon. This represents a pokemon you encounter at the first time. It has no trainers. By chance, you can catch it and be the trainer.

WildPokemon has the following fields:

1. int patience: an integer representing the patience of a wild pokemon. With no other effects, we assume a wild pokemon’s patience is from 0 to 100 inclusively. Wild pokemons with high patience are easier to catch, and vice versa. If the patience is 100, your catch will definitely succeed. If the patience is 0, the wild pokemon will disappear.

2. int speed: an integer representing the move speed of a wild pokemon. With no other effects, we assume a wild pokemon’s speed is from 0 to 100 inclusively. When we throw a pokeball to catch a wild pokemon, we will also have a speed for the ball (ballSpeed). The ball will only hit the wild pokemon if ballSpeed is faster than or equal to the speed of the pokemon. Remember, there’s no ballSpeed field for Pokeball class. We will explain how to get ballSpeed later.

3. int timesEscapedFromBall: After your ball hits a wild pokemon, there will be two outcomes: 1. The pokemon is caught; 2. The pokemon escapes from the ball. This field records the times that a wild pokemon escapes from a pokeball. It should be initialized to zero when you create any new wild pokemons.

For the no-arg constructor, you will set patience to be 100, speed to be 0, and timesEscapedFromBall to be 0.

Except getters and setters, WildPokemon has the following methods:

1. public void appear():

Logically, this method is called when you encounter a new wild pokemon for the first
time. You will not call it if the pokemon escapes from the ball and you’re catching for
another round. This method will print a message “You encountered a wild Pikachu!” if the
wild pokemon is Pikachu. And Pikachu will speak.
Example: If you encounter a wild Pikachu, the following will be printed to the terminal
(The empty line is also required):

```
You encounter a wild Pikachu!
Pikachu: pikapika!
```

2. **public boolean disappear():**
   This method checks if the wild pokemon is going to disappear. If the wild pokemon’s
   patience is less than or equal to 0, or the wild pokemon has escaped from a ball for more
   than three times, this method will return true. Otherwise, return false.

3. **public boolean isCaught(Berry berry, Pokeball pokeball):**
   This method takes in two items and checks if the wild pokemon is caught.
   To determine whether a pokemon is caught or not, first we need to see if the ball is
   faster than the Pokemon or not. Each time you throw a pokeball, it has a different speed.
   Let’s call it **ballSpeed**, a randomly generated integer value ranging from 0 to 100
   inclusive. Here we introduce the java.util.Random class to you. With the following piece
   of code, you can generate a random integer between 0 and 100 both inclusively:

   ```java
   Random rand = new Random(); //instance of random class
   int upperbound = 101;
   //generate random values from 0-100
   int ballSpeed = rand.nextInt(upperbound);
   ```

   Since the berry might decrease pokemon’s speed, you should prepare your pokemon’s
current speed according to your berry choice. If your **ballSpeed** is faster than Pokemon’s
current speed, the next thing you want to check is whether the Pokemon has enough
patience for you to catch. Each WildPokemon has a default patience, but **berry** and
**Pokeball** can further increase upon it. Therefore, the final patience can be calculated as:

   \[
   \text{finalPatience} = \text{SUM}(\text{pokemon’s patience}, \text{berry’s patienceIncrement}, \text{pokeball’s performance})
   \]

   Suppose the **final patience** is 80, that means the probability of catching Pokemon is
roughly 80% given your ball is fast enough.
You will need to generate a random number called `patienceNeed` to compare with `finalPatience`. If `finalPatience` is greater than or equal to `patienceNeed`, the wild Pokemon will be caught. Otherwise, it will escape. Record this escape properly and return accordingly.

Besides the logic of this method, we require you to make many print statements to let you understand what is going on inside your code. These print statements will also help debugging. Below are what you should print under all situations (your random generated number should be different from the following example and the empty line is also required):

- The ball named Pokeball missed

```
BallSpeed: 11
Pikachu's speed: 25
Your pokeball missed Pikachu!
```

- The ball named ultraball hit and caught the pokemon

```
BallSpeed: 62
Pikachu's speed: 25
Your ultraball hit Pikachu!

Patience need: 1
Pikachu's patience: 60
Golden Razz Berry's patience increment: 30
Golden Razz Berry's speed decrement: 0
ultraball's performance: 30
Your ultraball caught Pikachu!
```

- The ball hit named superball but the pokemon escaped

```
BallSpeed: 32
Pikachu's speed: 25
Your superball hit Pikachu!

Patience need: 100
Pikachu's patience: 60
Razz Berry's patience increment: 10
Golden Razz Berry's speed decrement: 0
superball performance: 10
Pikachu escaped from your superball!
```

**Part 5: PalPokemon.java**
PalPokemon is a subclass under Pokemon. This represents a pokemon you caught. PalPokemon has a String pokeballName field. The name of the ball that caught that pokemon will be recorded here. Complete all constructors, setters, and getters. For the no-arg constructor, set the pokeballName to be "undefined".

Then there is a method called comesOutFromBall(). Logically, this is called when you want to check the information of your certain PalPokemon. It will print some information of this PalPokemon and the PalPokemon will speak. Here’s an example of this method’s output:

```
Pikachu in ultraball, electric type pokemon.
Pikachu: pikapika!
```

**Part 6: Unit tests**

As before, you are encouraged to create as many test cases as you think to be necessary to cover all the edge cases. However, since there are so many methods in this assignment, we will not ask you to create test cases for each method. To get full credit, create at least six test cases that test different methods. We suggest making some print messages in each of your test cases so that you will know which test case is failing. The unitTests method should return true only when all the test cases are passed, otherwise return false.

**Part 6: Complete startYourJourney() in Assignment5.java**

Besides main and unitTests, you need to complete the startYourJourney() method in Assignment5.java. main() is already fully implemented for you in the start code (you don’t need to change anything in main).

You will encounter the following 5 pokemons in sequence in startYourJourney(). The following stats are already in Assignment5.java and you can use for loop and index to access these stats.

<table>
<thead>
<tr>
<th>Pokemon Name</th>
<th>Sound</th>
<th>Type</th>
<th>Patience</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pikachu</td>
<td>pikapika</td>
<td>electric</td>
<td>60</td>
<td>25</td>
</tr>
<tr>
<td>Bulbasaur</td>
<td>bulb</td>
<td>grass</td>
<td>50</td>
<td>5</td>
</tr>
<tr>
<td>Charmander</td>
<td>char</td>
<td>fire</td>
<td>50</td>
<td>20</td>
</tr>
</tbody>
</table>
Below are three kinds of balls you can choose:

<table>
<thead>
<tr>
<th>Pokeball Name</th>
<th>performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>pokeball</td>
<td>0</td>
</tr>
<tr>
<td>superball</td>
<td>10</td>
</tr>
<tr>
<td>ultraball</td>
<td>30</td>
</tr>
</tbody>
</table>

There are two parts you need to implement in this method.

1. Encounter and catch each pokemon (inside the given for loop):
   First you need to create the current wild pokemon. Then it will appear. Print the Prompt for berry input after that. Then, you will enter a loop. This loop will only stop/break when the current pokemon is caught or disappear. Inside this loop, first you will keep asking the user to choose a berry (or skip) with the Scanner until there is a match with the preset berry name (or skip). If the input doesn’t match, you will print the provided `PROMPT_MSG_BERRY_WRONG_INPUT` and ask for the correct input again. Prepare your berry according to the correct input. Similarly, you will ask for the correct pokeball input after you receive the correct berry input. An example of this is shown below.

<table>
<thead>
<tr>
<th>You encounter a wild Pikachu!</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pikachu: pikapika!</td>
</tr>
<tr>
<td>Which berry do you want to use? Type skip if you are confident.</td>
</tr>
<tr>
<td>(Razz Berry, Nanap Berry, Golden Razz Berry, skip)</td>
</tr>
</tbody>
</table>
no berry

Your berry input was not recognized. Choose one from the following four options: Razz Berry, Nanap Berry, Golden Razz Berry, skip

Which pokeball do you want to use? (pokeball, superball, ultraball)

noball

Your pokeball input was not recognized. Choose one from the following four options: pokeball, superball, ultraball

pokeball

After the above input asking and item preparation, you will check if the pokemon is caught.

1. If true, you will update the PalPokemon at the corresponding index of the provided myPalPokemons array and start to encounter the next pokemon. For example, if you missed Pikachu but caught Bulbasaur, your myPalPokemons array’s 0 position should be unchanged, and your myPalPokemons array’s 1 position should have the corresponding data of Bulbasaur.

2. If false, you will print a message explicitly in this method saying something like:

Pikachu's times escaped from ball: 1

The “ball” here should be plain text. You don’t need to put the ballName variable here. You need to reset the patience and speed of this wild pokemon to the default value (if you haven’t changed the value of the wild pokemon, ignore this.) Then, you also need to deduct 5*timesEscapedFromBall from the patience. For example, the default patience of Pikachu is 60. If Pikachu escaped twice from balls, the patience after the second escape is 50.

Then, check if the pokemon will disappear. If true, start the next encounter with the next pokemon. If false, then start another round with the current pokemon.

After your pokemon escapes from the ball, all the berry and ball effects will be cleared.

Hint: use the preset stats for every round.

2. Check all pokemon you just caught
After you have finished with all five pokemons, now you can check what you have now. You should be able to see your pals in the myPalPokemon array. Let everyone comesOutFromBall. Make sure you don't call out something that is not your pal.

You can find a detailed expected output at the end of this writeup. It covers almost every case you should take care of (No invalid input cases. See above for reference). If you still have any problem with how everything should work, check the discussion video that will be released on Friday.

You are encouraged to create other settings to validate your implementation or just for fun. Also, make sure you close the scanner after you are done using it.

**Part 6: Coding style**

When coding in Java, there are several style rules that people usually follow to make the code clean and readable. In this course, you are asked to follow rules specified in link below:

https://cseweb.ucsd.edu/classes/fa20/cse8B-a/styleguide.html

Read the coding style guide carefully and refine your code for this and all future assignments.

**Submission**

Very important! Please follow the instructions below carefully and make the exact submission format. This is important since we will use scripts to grade so if you don't follow the same submission format you probably will receive a zero.

1. Go to Gradescope and click on PA5.
2. Click the DRAG & DROP section and directly select the seven required files (Assignment5.java, Berry.java, Item.java, PalPokemon.java, Pokeball.java, Pokemon.java, WildPokemon.java). Drag & drop is fine. Please make sure you don't submit a zip. Just the file solely. Make sure the name of the file is correct.
3. You can resubmit unlimited times before the due date. Your score will depend on your final submission, even if your former submissions have a higher score.
4. The autograder is for the use of the instructional team. You won't see the result of the autograder. As long as you uploaded your file you're good to go.
Appendix: Expected Output

All unit tests passed.

This is the start of your pokemon trainer journey.  
You will encounter different pokemons. 
You can throw different poke balls to catch them.  
You can use Razz Berry or Golden Razz Berry to increase the catch rate.  
You can use Nanap Berry to make the ball hit more easily.  
Up to one berry per throw. Use the berry before your ball throw.

You encounter a wild Pikachu!  
Pikachu: pikapika!

Which berry do you want to use? Type skip if you are confident. (Razz Berry, Nanap Berry, Golden Razz Berry, skip)  
skip

Which pokeball do you want to use? (pokeball, superball, ultraball)  
pokeball

BallSpeed: 85  
Pikachu's speed: 25  
Your pokeball hit Pikachu!

Patience need: 95  
Pikachu's patience: 60  
item's patience increment: 0  
item's speed decrement: 0  
pokeball's performance: 0  
Pikachu escaped from your pokeball!

Pikachu's times escaped from ball: 1

Which berry do you want to use? Type skip if you are confident. (Razz Berry, Nanap Berry, Golden Razz Berry, skip)  
skip

Which pokeball do you want to use? (pokeball, superball, ultraball)  
ultraball

BallSpeed: 11  
Pikachu's speed: 25  
Your ultraball missed Pikachu!

Pikachu's times escaped from ball: 1
Which berry do you want to use? Type skip if you are confident. (Razz Berry, Nanap Berry, Golden Razz Berry, skip)
Golden Razz Berry

Which pokeball do you want to use? (pokeball, superball, ultraball)
ultraball

BallSpeed: 62
Pikachu's speed: 25
Your ultraball hit Pikachu!

Patience need: 1
Pikachu's patience: 55
Golden Razz Berry's patience increment: 30
Golden Razz Berry's speed decrement: 0
ultraball's performance: 30
Your ultraball caught Pikachu!

You encounter a wild Bulbasaur!
Bulbasaur: bulb!

(some pokemon catches are omitted here.)

You encounter a wild Mew!
Mew: mew!

Which berry do you want to use? Type skip if you are confident. (Razz Berry, Nanap Berry, Golden Razz Berry, skip)
skip

Which pokeball do you want to use? (pokeball, superball, ultraball)
pokeball

BallSpeed: 79
Mew's speed: 50
Your pokeball hit Mew!

Patience need: 60
Mew's patience: 15
item's patience increment: 0
pokeball's performance: 0
Mew escaped from your pokeball!

Mew's times escaped from ball: 1

Which berry do you want to use? Type skip if you are confident. (Razz Berry, Nanap Berry, Golden Razz Berry, skip)
skip

Which pokeball do you want to use? (pokeball, superball, ultraball)
pokeball

BallSpeed: 1
Mew's speed: 50
Your pokeball missed Mew!

Mew's times escaped from ball: 1

Which berry do you want to use? Type skip if you are confident. (Razz Berry, Nanap Berry, Golden Razz Berry, skip)
skip

Which pokeball do you want to use? (pokeball, superball, ultraball)
pokeball

BallSpeed: 51
Mew's speed: 50
Your pokeball hit Mew!

Patience need: 13
Mew's patience: 10
item's patience increment: 0
pokeball's performance: 0
Mew escaped from your pokeball!

Mew's times escaped from ball: 2

Which berry do you want to use? Type skip if you are confident. (Razz Berry, Nanap Berry, Golden Razz Berry, skip)
skip

Which pokeball do you want to use? (pokeball, superball, ultraball)
pokeball

BallSpeed: 72
Mew's speed: 50
Your pokeball hit Mew!

Patience need: 47
Mew's patience: 5
item's patience increment: 0
pokeball's performance: 0
Mew escaped from your pokeball!

Mew's times escaped from ball: 3
Mew disappears...

Now let's check who's your pal now!

Pikachu in ultraball, electric type pokemon.
Pikachu: pikapika!

Bulbasaur in superball, grass type pokemon.
Bulbasaur: bulb!

Squirtle in pokeball, water type pokemon.
Squirtle: squir!