

Pokémon Battle Version 4: Battle!

Finally, it's time to implement the battle! The battle will be simulated. It means that two Pokémon will battle each other automatically, **without any interaction from the user**.

Battle rules

Round logic

For example, let's assume that Pikachu and Bulbasaur have been selected to battle.

1. Pikachu attempts an attack on Bulbasaur
2. If Bulbasaur is dead after the attack, there is a 50% chance that it is revived. If Bulbasaur has not been revived, the battle is over. Pikachu wins.
3. If Bulbasaur is alive, it attempts an attack on Pikachu.
4. If Pikachu is dead after the attack, there is a 50% chance that it is revived. If Pikachu has not been revived, the battle is over. Bulbasaur wins.
5. If both Pokemon are alive, they continue for the next round.

The battle can have no more than 10 rounds. If both Pokemon are alive after 10 rounds have passed, there is a tie.

Attack logic

Each attack attempt proceeds as follows:

1. First, the coefficient of **luck** is calculated. It is a random value in a range of 0.7 to 1.3 with a step of 0.1.
2. **Damage** is calculated as the attacking Pokemon's attack level multiplied by the coefficient of luck. It's rounded to the nearest integer.
3. If the damage is greater than the defending Pokemon's defense level, then the attack attempt is successful, and the defending Pokemon's health loss is equal to the difference between the damage and their defense level. If the damage is equal to or less than the defending Pokemon's defense level, then the attack is blocked, and the defending Pokemon does not lose any health.

Example of an attack attempt

Attacking Pokemon: Bulbasaur





- attack = 49
- defense = 49
- health = 45

Defending Pokemon: Pikachu

- attack = 55
- defense = 40
- health = 35

The attack will go like this:

1. luck = 1.1 (randomly)
2. damage = 49 (Bulbasaur's attack) * 1.1 = 53.9, rounded to 54
3. $54 > 40$ (Pikachu's defense) → attack is successful → Pikachu loses $(54-40) = 14$ health points. Pikachu had 35 health points, so, after losing 14, it has 21 health points.

<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 10</p> <p>Round 1</p> 	<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 10</p> <p>Round 1</p>  <p>Pikachu attempts attack on Diglett</p>	<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 10</p> <p>Calculate Pikachu's luck based on randomly selecting a value from 0.7 - 1.3 (Must be rounded to the nearest 0.1)</p> <p>This time, our randomly chosen value for Pikachu's luck is 1.1!</p>
<p>Attack: 55 Defense: 40 Health: 35 Luck this round: 1.1</p>  <p>Damage value = Pikachu's attack * luck = 55 * 1.1 = 60.5 (round to 61)</p>	<p>Attack: 55 Defense: 40 Health: 35 Luck this round: 1.1 Damage this round: 61</p> <p>Diglett loses health equal to Pikachu's damage minus Diglett's Defense</p> <p>Health Loss = 61 - 30 = 31</p> <p>Diglett will lose 31 health!</p>	<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 0</p> <p>Diglett lost at least 10 health, so now his health is at zero!</p> <p>He has a 50% chance to be revived. If it works, he can still attack Pikachu back</p> 
<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 10</p> <p>The revive worked this time!</p> <p>Diglett's health is set back to his maximum health of 10</p> 	<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 10</p>  <p>Diglett's turn to attempt an attack on Pikachu</p>	<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 10</p> <p>Calculate Diglett's luck based on randomly selecting a value from 0.7 - 1.3 (Must be rounded to the nearest 0.1)</p> <p>This time, our randomly chosen value for Diglett's luck is 0.7!</p>
<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 10 Luck this round: 0.7</p>  <p>Damage value = Diglett's attack * luck = 55 * 0.7 = 38.5 (round to 39)</p>	<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 10 Luck this round: 0.7 Damage this round: 39</p> <p>Pikachu loses health equal to Diglett's damage minus Pikachu's Defense</p> <p>Health Loss = 39 - 40 = -1</p> <p>Pikachu's defense is higher than the damage, so Pikachu takes no damage and his health does not change!</p>	<p>Attack: 55 Defense: 40 Health: 35</p> <p>Attack: 55 Defense: 30 Health: 10</p>  <p>On to Round 2!</p> <p>Play continues until a Pokemon is brought to zero health and fails to revive, or 10 rounds have completed!</p>

What to do

This version builds on Version 3.

1. First, you will add the `attempt_attack` method to the `Pokemon` class that you implemented in Version 2.

You **must** use the following template to implement the method in the Pokemon class:

```
def attempt_attack(self, other: "Pokemon") -> bool:
    """
    Attempt an attack on another Pokemon.

    """
    # TODO: implement this method based on the attack logic above
```

2. Your main function will implement the game loop, based on the round logic described above. You must use the `is_alive` and `revive` methods of the class to implement the round logic.

Hints

- Import the random module and use `random.choice([0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3])` to get the random coefficient of luck.
- Use `random.choice(True, False)` to implement the 50% chance part.
- Use the following while loop header to implement the round logic:

```
while pokemon1.is_alive() and pokemon2.is_alive():
```

Program name

Save your program as `pokemon4.py`.

Demo

<https://asciinema.org/a/FrJdhkCBez5A0E9pakvh2U0N5>

Testing

To make sure your program works correctly, you should test it.

Run your program with `python pokemon4.py`. Your program selects two random Pokémon, and each attack attempt is simulated with a random component. So each battle will be unique.

Example scenarios

1. [Sliggoo vs Doublade](#)
2. [Stantler vs Toxicroak](#)
3. [Unown vs Swirlx](#)
4. [Trevenant vs Mismagius](#)
5. [Torracat vs Zorua](#)
6. [Rhyhorn vs Frillish](#)
7. [Mismagius vs Amoonguss](#)
8. [Dusclops vs Pachirisu](#)
9. [Bagon vs Carvanha](#)
10. [Bayleef vs Suicune](#)

Submitting

Submit `pokemon4.py` via eClass.

You may submit either all versions you complete, or only the final version.

Copyright

I. Akhmetov, J. Schaeffer, M. Morris and S. Ahmed, Department of Computing Science, Faculty of Science, University of Alberta (2023).