course_4_assessment_2

Due: 2019-02-04 15:14:00

Description: Assessment for the Inheritance lesson — Score: 0 of 3 = 0.0%

Questions

Not yet graded

The class, `Pokemon`, is provided below and describes a Pokemon and its leveling and evolving characteristics. An instance of the class is one pokemon that you create.

`Grass_Pokemon` is a subclass that inherits from `Pokemon` but changes some aspects, for instance, the boost values are different.

For the subclass `Grass_Pokemon`, add another method called `action` that returns the string `"[name of pokemon] knows a lot of different moves!"`. Create an instance of this class with the `name` as `"Belle"`. Assign this instance to the variable `p1`.

Save & Run | 5/15/2021, 1:43:21 PM - 2 of 2 | Show in CodeLens

```
class Pokemon(object):
    attack = 12
    defense = 10
    health = 15
    p_type = "Normal"

    def __init__(self, name, level = 5):
        self.name = name
        self.level = level

    def train(self):
        self.update()
        self.attack_up()
        self.defense_up()
```

ActiveCode (ee_inheritance_01)

Result	Actual Value	Expected Value	Notes
Pass	'Belle...oves!'	'Belle...oves!'	Testing that action method is correct and p1 assigned to correct value

Expand Differences

You passed: 100.0% of the tests

Not yet graded

Modify the `Grass_Pokemon` subclass so that the attack strength for `Grass_Pokemon` instances does not change until they reach level 10. At level 10 and up, their attack strength should increase by the `attack_boost` amount when they are trained.

To test, create an instance of the class with the name as `"Bulby"`. Assign the instance to the variable `p2`. Create another instance of the `Grass_Pokemon` class with the name set to `"Pika"` and assign that instance to the variable `p3`. Then, use `Grass_Pokemon` methods to train the `p3` `Grass_Pokemon` instance until it reaches at least level 10.

Save & Run | 5/15/2021, 1:43:49 PM - 2 of 2 | Show in CodeLens

```
class Pokemon(object):
    attack = 12
    defense = 10
    health = 15
    p_type = "Normal"

    def __init__(self, name, level = 5):
        self.name = name
        self.level = level

    def train(self):
        self.update()
        self.attack_up()
        self.defense_up()
```

ActiveCode (ee_inheritance_02)

Result	Actual Value	Expected Value	Notes
Pass	'Pokem...el: 5'	'Pokem...el: 5'	Testing that p2 is assigned to correct value.
Pass	True	True	Testing that attack value is assigned to correct value at level 10.

Expand Differences

You passed: 100.0% of the tests

Not yet graded

Along with the `Pokemon` parent class, we have also provided several subclasses. Write another method in the parent class that will be inherited by the subclasses. Call it `opponent`. It should return which type of pokemon the current type is weak and strong against, as a tuple.

- **Grass** is weak against *Fire* and strong against *Water*
- **Ghost** is weak against *Dark* and strong against *Psychic*
- **Fire** is weak against *Water* and strong against *Grass*
- **Flying** is weak against *Electric* and strong against *Fighting*

For example, if the `p_type` of the subclass is `'Grass'`, `.opponent()` should return the tuple `('Fire', 'Water')`

Save & Run 5/15/2021, 1:44:19 PM - 2 of 2 Show in CodeLens

```
class Pokemon():
    attack = 12
    defense = 10
    health = 15
    p_type = "Normal"

    def __init__(self, name,level = 5):
        self.name = name
        self.level = level
        self.weak = "Normal"
        self.strong = "Normal"

    def train(self):
        self.update()
```

ActiveCode (ee_inheritance_05)

Result	Actual Value	Expected Value	Notes	
Pass	('Fir...ter')	('Fir...ter')	Testing that Grass weak and strong are assigned to correct values.	Expand Differences
Pass	('Wat...ass')	('Wat...ass')	Testing that Fire weak and strong are assigned to correct values.	Expand Differences
Pass	('Dar...hic')	('Dar...hic')	Testing that Ghost weak and strong are assigned to correct values.	Expand Differences
Pass	('Ele...ing')	('Ele...ing')	Testing that Flying weak and strong are assigned to correct values.	Expand Differences

You passed: 100.0% of the tests

Score Me