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 | Load History | Show 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)

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 | Load History | Show 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)

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 | Load History | Show CodeLens

```
class Ghost_Pokemon(Pokemon):
    p_type = "Ghost"

    def update(self):
        self.health_boost = 3
        self.attack_boost = 4
        self.defense_boost = 3

```

```
class Fire_Pokemon(Pokemon):
    p_type = "Fire"

class Flying_Pokemon(Pokemon):
    p_type = "Flying"

```

ActiveCode (ee_inheritance_05)

Score Me