PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2233 - Programación Avanzada
2^o semestre 2016

Actividad 06

Decoradores

Instrucciones

Previo a las fiestas patrias, te encuentras con el Profesor Oak quien te contó sobre la nueva actualización que le hizo a la Pokedex. El profesor te mencionó sobre los nuevos métodos que agregó y te entregó el código con la condición de que **NO LO MODIFIQUES.** Luego de que se fuera ~~a la PrograFonda~~ a seguir con su investigación, notaste que existían errores entre lo que los métodos realmente hacen y lo que el Profesor Oak dijo que hacían. Como tú eres un muy buen alumno de Programación Avanzada y ya aprendiste decoradores, puedes arreglar los errores que cometió el profesor sin editar su código (aparte de agregar los decoradores, claro).

Requerimientos

- El programa debe correr sin errores.
- Programar y aplicar decoradores que tengan los siguientes comportamientos:
  - Verificar si los atributos `nombre`, `level`, `evolutions` y `owner` de Pokemon sean `string`, `int`, `list`, `string`, respectivamente.
  - Recibir una cantidad arbitraria de métodos y convertirlos en privados si es que existen.
  - Retornar el nombre del pokemon cuando se imprime una instancia.
  - Restringir los objetos agregados a la lista del Pokedex a objetos de tipo Pokemon.
  - Imprimir cuál es el Pokemon más fuerte registrado en la lista de tu Pokedex y el nivel en que está.

Notas

- No se puede modificar ninguna linea del archivo `main.py` entregado, excepto para agregar los decoradores. No cumplir con este requisito será **castigado con nota 1.0**.

To - DO

- (1.00 pts) Implementar correctamente el decorador `verify`. Debe recibir los tipos de datos (`str`, `list`, `int`, `str`) para chequearlos con los atributos de la instancia. Si algun atributo no cumple con su tipo de dato, entonces se debe evitar la instanciación y levantar una excepción (Ver Tips)
- (1.50 pts) Implementar correctamente el decorador `protect_method`. Debe recibir una cantidad no determinada de nombres de métodos, revisar si existen dentro de la clase y hacerlos privados. En caso de que el dueño `(owner)` del pokemon sea el Prof. Oak, debe permitir el acceso a esos métodos privados. En caso de que no exista el nombre del método debe imprimir en consola la siguiente linea: `"'<nombre_metodo>' no es un metodo de la clase".`
- (1.00 pts) Implementar correctamente el decorador `my_name`. Este debe modificar la función decorada, de tal forma que su retorno sea el nombre del Pokemon correspondiente.
- (1.50 pts) Implementar correctamente el decorador `is_pokemon`. Debe ser capaz de restringir que el argumento recibido por la función decorada sea de tipo Pokemon.
- (1.00 pts) Implementar correctamente el decorador `strongest`. Este debe cambiar la función decorada por una que, además del nombre, imprima el nivel del Pokemon con más alto nivel registrado en la Pokedex.

Tips

- La función `isinstance(obj, class)` retorna `True` si `obj` es instancia de la clase `class`.
- Para obtener, asignar o borrar un método de un objeto puede usar las siguientes funciones
  - `getattr(obj, nombre_metodo)`
  - `setattr(obj, nombre_metodo, metodo)`
  - `delattr(obj, nombre_metodo)`
- Puede obtener una lista con los nombres de atributos y métodos de un objeto usando el comando `dir(objeto)`.
- Para levantar una excepción puede usar `raise Exception ('mensaje de error').`
- Considere si se levanta una excepción en el `try`, entonces ingresará al `except Exception`.

Output

```
Su verify parece funcionar :)
Error, este no es un Pokemon valido!
Metodo privado
Bienvenido Prof. Oak
Mew
Llevas 2 Pokemon:
[Mew, Magikarp]
El mas fuerte es Mew con level 80
```

Entrega

- **Lugar:** GIT - Carpeta: Actividades/AC06
- **Hora:** 16:55