P2: Pakudex

Overview

This project will provide students with practice building and working with object-oriented programming constructs including classes and objects by building classes to represent creatures and a cataloguing system.

Scenario

NOTE: This project concept is a work of satire. To state the obvious: we do not advise one to go around imprisoning creatures in small receptacles held in one's pockets and/or having them fight for sport.

Pouch Creatures – abbreviated "Pakuri" – are the latest craze sweeping elementary schools around the world. Tiny magical creatures small enough to fit into one's trouser pouches (with enough force applied, 'natch) have begun appearing all around the world in forests. They come in all shapes colors. When stolen from their parents at a young enough age, they can be kept in small spherical cages (for their own good) easily carried by elementary school children (though they are also popular with adults). This has led to an unofficial catch phrase for the phenomenon – "Gotta steal 'em all!" – a play on the abbreviation "Pakuri" (which doubles as Japanese slang meaning "to steal"). Young children can then pit their Pakuri against one another in battle for bragging rights or to steal them from one another. (Don't worry – they heal their wounds quickly!)

Of course, keeping track of all these critters can be a real task, especially when you are trying to steal so many of them at such a young age! You've decided to cash in – hey, if you don't someone else will – on the morally ambiguous phenomenon by developing an indexing system – a ***Pakudex*** – for kids and adult participants.

Requirements

Students will write two files to submit: a main **pakudex** module (**pakudex.py**) containing a driver program with the **main()** function as the entry point and a **Pakudex** class and a **pakuri** module (**pakuri.py**) containing the **Pakuri** class. All output goes to standard output unless otherwise specified. All attributes / methods must be private unless noted in the specification!

Entry Point

When run as a program, the **pakudex** module should…

1) Display a welcome message
2) Prompt for input & conduct input error checking
3) Follow the output and formatting in this document
4) Not have print statements in the class method calls
5) Only run main() if invoked directly (i.e., check __name__)
6) Have ***no global variables***

```
Welcome to Pakudex: Tracker Extraordinaire!

Pakudex Main Menu
-----------------
1. List Pakuri
2. Show Pakuri
3. Add Pakuri
4. Remove Pakuri
5. Evolve Pakuri
6. Sort Pakuri
7. Exit

What would you like to do?
```

Listing Pakuri

This should number and list the critters in the Pakudex in the order contained. For example, if “Pikarat” and “Charasaurus” were added to the Pakudex (in that order), before sorting, the list should be:

Success

```
Pakuri in Pakudex:
1. Pikarat
2. Charasaurus
```

Failure

```
No Pakuri currently in the Pakudex!

Pakudex Main Menu
```

Show Pakuri

The program should prompt for a species and collect species information, then display it:

Success

```
Enter the name of the species to display: Quacker

Species: Quacker
Level: 15
CP: 286
HP: 45
```

Failure

```
Enter the name of the species to display: PsyDuck
Error: No such Pakuri!

Pakudex Main Menu
-----------------
1. List Pakuri
```

Adding Pakuri

When adding a Pakuri, a prompt should be displayed to read in the species name, and a confirmation displayed following successful addition (or failure).

Success

```
Species: Quacker
Level: 15
Pakuri species Quacker (level 15) added!

Pakudex Main Menu
-----------------
1. List Pakuri
2. Show Pakuri
```

Failure – Duplicate

```
Species: Quackers
Error: Pakudex already contains this species!
```

Failure – Level

```
Level: -15
Level cannot be negative.
Level: NINE-THOUSAND
Invalid level!
```

Removing Pakuri

The program should prompt for a species name and then remove the Pakuri if it is in the Pakudex:

Success

```
Enter the name of the Pakuri to remove: Quacker
Pakuri PsyGoose removed.
```

Failure

```
Enter the name of the Pakuri to remove: PsyDuck
Error: No such Pakuri!
```

Evolve Pakuri

The program should prompt for a species and then cause the species to evolve if it exists:

Success

```
Enter the name of the species to evolve: Quacker
Quacker has evolved!
```

Failure

```
Enter the name of the species to evolve: PsyDuck
Error: No such Pakuri!
```

Sort Pakuri

Sort Pakuri in Java standard lexicographical order:

```
Pakuri have been sorted!
```

Exit

Quit the program:

```
Thanks for using Pakudex! Bye!
```

Pakuri Class

This class will be the blueprint for the different critter objects that you will create. You will need to store information about the critter's species, attack level, defense level, and stamina. All variables storing information about the critters **must be private** (inaccessible from outside of the class per Python convention). We recommend (but do not mandate) the following variable types and names:

```
__species: str
__level, __attack, __defense, __stamina: int
```

These attack, defense, and speed levels should have the following initial values when first created:

Attribute	Value
`attack`	`(len(species) * 7 + 11) % 16`
`defense`	`(len(species) * 5 + 17) % 16`
`stamina`	`(len(species) * 6 + 13) % 16`

While `Pakuri` attributes are never revealed to the user, they are used (along with level) to determine the combat power (CP) and healthy points (HP), as follows:

$$\boldsymbol{HP} = Floor(\boldsymbol{Stamina} \times \boldsymbol{Level} / \mathbf{6})$$

$$\boldsymbol{CP} = Floor(\boldsymbol{Attack} \times \sqrt{\boldsymbol{Defense}} \times \sqrt{\boldsymbol{Stamina}} \times \boldsymbol{Level} \times \mathbf{0.08})$$

(You may have noticed Pakuri don't have individual names, just species; don't worry! They won't live long enough for it to matter with all of the fighting. Your conscience can be clear!)

Required Methods

The class must include the following methods exactly as defined:

`__init__``(self, species: str, level: int)`
This method should be the **only** constructor for this class. <u>There should not be a default constructor!</u>

`get_species``(self) -> `*`str`*
Returns the species of this critter

`get_attack``(self) -> `*`int`*
Returns the attack value for this critter

`get_defense``(self) -> `*`int`*
Returns the defense value for this critter

`get_stamina``(self) -> `*`int`*
Returns the speed of this critter

`set_attack``(self, new_attack: int)`
Changes the attack value for this critter to **`new_attack`**

Required Properties

These properties must be included as part of the class:

`cp` *(read-only)*
Calculates and returns the **`Pakuri`** object's combat power (CP).

`hp` *(read-only)*
Calculates and returns the **`Pakuri`** object's health points (HP).

`level` *(read-write)*
Gets, or sets, the **`Pakuri`** object's level attribute.

Pakudex Class

The `Pakudex` class will contain all the Pakuri that you will encounter as `Pakuri` objects.

Required Methods

The class must include the following methods exactly as defined:

`__init__``(self)`
Default constructor; should prepare a new `Pakudex` object.

`get_species_list``(self) -> list[str]`
Returns a list of names of the species of the critters as ordered in the Pakudex; if there are no species added yet, this method should return `None`.

`get_stats``(self, species: str) -> list[int]`
Returns an `int` list containing the level, CP, and HP of `species` at indices 0, 1, and 2 respectively; if `species` is not in the Pakudex, returns `None`.

`sort_pakuri``(self)`
Sorts `Pakuri` objects in this `Pakudex` according to Python standard lexicographical ordering of species name.

`add_pakuri``(self, species: str, level: int) -> bool`
Adds `species` to the Pakudex; if successful, return `True`, and `False` otherwise

`remove_pakuri``(self, species: str) -> bool`
Removes a `species` from the Pakudex; if successful, return `True`, and `False` otherwise

`evolve_species``(self, species: str) -> bool`
Attempts to evolve `species` within the Pakudex. This should double the Pakuri's level and increment (increase by one) its attack. If successful, return `True`, and `False` otherwise

Submissions

NOTE: Your output must match the example output *exactly*. If it does not, ***you will not receive full credit for your submission***!

Files:	pakudex.py, pakuri.py
Method:	Submit on ZyLabs