Project 3: 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 cataloging system.

Submission Instructions

Name your files pakuri_program.py, pakuri.py, pakudex.py, and upload all three files to the canvas assignment.

Scenario

NOTE: This homework 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 and 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 create three files: a file with a main function (pakuri_program.py) and two files containing classes (pakuri.py and pakudex.py).

Project 3: Pakudex

pakuri_program.py

When run, the program should…

1. Display a welcome message
2. Prompt for / read pakudex capacity
3. Display the menu
4. Prompt for input

```
Welcome to Pakudex: Tracker Extraordinaire!
Enter max capacity of the Pakudex: 30
The Pakudex can hold 30 species of Pakuri.

Pakudex Main Menu
-----------------
1. List Pakuri
2. Show Pakuri
3. Add Pakuri
4. Evolve Pakuri
5. Sort Pakuri
6. 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 “Pikaju” and “Charasaurus” were added to the pakudex (in that order), before sorting, the list should be:

Success:

```
Pakuri In Pakudex:
1. Pikaju
2. Charasaurus
```

Failure:

```
No Pakuri in Pakudex yet!
```

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: PsyGoose

Species: PsyGoose
Attack: 65
Defense: 57
Speed: 61
```

Failure:

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

```
Enter the name of the species to add: PsyGoose
Pakuri species PsyGoose successfully added!
```

Failure - Duplicate:

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

Failure - Full:

```
Error: Pakudex is full!
```

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: PsyGoose
PsyGoose has evolved!
```

Failure:

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

Sort Pakuri

Sort Pakuri in Python standard lexicographical order:
(hint: lists have a .sort() method)

```
Pakuri have been sorted!
```

Exit
Quit the program:

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

pakuri.py

This file will contain the Pakuri class. This class will be the blueprint for the different pakuri objects you will create. You will need to store information about the critter's species, attack, defense, and speed. We recommend the following variable types and names:

species: str
attack, defense, speed: int

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

Attribute	Value
attack	(len(species) * 7) + 9
defense	(len(species) * 5) + 17
speed	(len(species) * 6) + 13

(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 the fighting. Your conscience can be clear!)

The class must also have the following methods and behaviors (**this is mandatory**):

`def __init__(self, species)`
Initialize the pakuri object with species attribute

`def get_species(self)`
Returns the species of this critter

`def get_attack(self)`
Returns the attack value of this critter

`def get_defense(self)`
Returns the defense value for this critter

`def get_speed(self)`
Returns the speed of this critter

def set_attack(self, new_attack)
Changes the attack value for this critter to new_attack

`def evolve(self)`
Will evolve the critter as follows: double the attack, quadruple the defense, and triple the speed

pakudex.py

This file will contain the Pakudex class. This class will contain all the pakuri that you encounter as Pakuri objects. Note: The pakudex will have a set size determined by user input at the beginning of the program's run; the number of species contained in the pakudex will never grow beyond this point.

The class must have the following methods and behaviors (**this is mandatory**):

`def __init__(self, capacity=20)`
Initializes this object to contain exactly capacity objects when completely full. The default capacity pakudex should be 20

`def get_size(self)`
Returns the number of critters currently being stored in the pakudex

`def get_capacity(self)`
Returns the number of critters that the pakudex has the capacity to hold at most

`def get_species_array(self)`
Returns a list of strings containing the species of the critters as ordered in the pakudex; if there are no species added yet, this method should return None

`def get_stats(self, species)`
Returns a list of ints containing the attack, defense, and speed statistics of species at index 0, 1, and 2 respectively; if species is not in the pakudex, returns None

`def sort_pakuri(self)`
Sorts the pakuri objects in this pakudex according to Python standard lexicographical ordering of species name *(hint: lists have a .sort() method)*

`def add_pakuri(self, species)`
Adds species to the pakudex; if successful, return True, and False otherwise

`def evolve_species(self, species)`
Attempts to evolve species within the pakudex; if successful, return True, and False otherwise