Classes Challenge 36: Pythonagachi Simulator App

Description:
You will be responsible for writing a program that simulates the behavior of a retro 90's Tamagachi toy. You program will allow a user to create their own creature, give it a name, and care for it until it unfortunately parishes. Users will monitor the creatures hunger, boredom, tiredness, and dirtiness and take actions to prevent any of the categories from getting to high. If the categories get too high there will be unfortunate consequences.

Step By Step Guide:
Defining your classes

- Define a class Creature.

Defining your class methods

- Define an __init__() method for the Creature class which takes two parameters, the required self parameter and a name.
    - Set the name of the creature to the name provided in title case.
    - Initialize the following creature attributes and set them equal to zero:
        - hunger, boredom, tiredness, dirtiness
    - Initialize an attribute called food which will track how much food is in the creatures inventory and set it equal to two.
    - Initialize an attribute called is_sleeping and set it equal to False.
    - Initialize an attribute called is_alive and set it equal to True.

- Define an eat() method for the Creature class which takes one parameter, the required self parameter.
    - If the creature has food available:
        - Decrease the food supply by one.
        - Decrease the creatures hunger by a random value from 1 to 4.
        - Print a statement stating the creature ate a great meal.
    - Else:
        - Print a statement stating the creature has no food.
    - If the current value of the creatures hunger is less than zero, set it to zero.

- Define a play() method for the Creature class which takes one parameter, the required self parameter.
    - Create a random integer from 0 to 2.
    - Inform the user that the creature wants to play a game.
    - The creature is thinking of a number 0, 1, or 2.
    - Get user input for a number guess.
    - If the guess is correct:
        - Print a statement stating the user is correct.
        - Decrease the creatures boredom by 3.
    - Else, the guess is incorrect:

        - Print a statement stating the user was incorrect.
        - Decrease the creatures boredom by 1.
    - If the current value of the creatures boredom is less than zero, set it equal to zero.

- Define a sleep() method for the Creature class which takes one parameter, the required self parameter.
    - Set the creatures is_sleeping attribute to True.
    - Decrease the creatures tiredness by 3.
    - Decrease the creatures boredom by 2.
    - Print a message indicating the creature is sleeping.
    - If the current value of the creatures tiredness is less than zero, set it equal to zero.
    - If the current value of the creatures boredom is less than zero, set it equal to zero.

- Define an awake() method for the Creature class which takes one parameter, the required self parameter.
    - Create a random integer from 0 to 2.
    - If the integer created is equal to 0:
        - Print a message that the creature just woke up.
        - Set the creatures is_sleeping attribute to False.
        - Set the creatures boredom to 0.
    - Else:
        - Print a message that the creature won't wake up.
        - Call the creatures sleep() method.

- Define a clean() method for the Creature class which takes one parameter, the required self parameter.
    - Set the creatures dirtiness to 0.
    - Print a message stating that the creature took a bath.

- Define a forage() method for the Creature class which takes one parameter, the required self parameter.
    - Create a variable called food_found and set it equal to a random integer from 0 to 4.
    - Increase the creatures food by this amount.
    - Increase the creatures dirtiness by 2.
    - Print a message stating how much food the creature found.

- Define a show_values() method for the Creature class which takes one parameter, the required self parameter.
    - Display all of the current attributes for the creature.
        - Name, hunger, boredom, tiredness, dirtiness, food inventory, and sleeping status.

- Define an incriment_values() method for the Creature class which takes two parameters, the required self parameter and an integer representing the game difficulty .
    - The user will set the difficulty (1 - 5) at the start of the game.
    - Increase the creatures hunger by a random integer from 0 to the value of the difficulty set by the user.

    - If the creatures is_sleeping attribute is False:
        - Increase the creatures boredom by a random integer from 0 to the value of the difficulty set by the user.
        - Increase the creatures tiredness by a random integer from 0 to the value of the difficulty set by the user.
    - Increase the creatures dirtiness by a random integer from 0 to the value of the difficulty set by the user.

- Define a kill() method for the Creature class which takes one parameter, the required self parameter.
    - If the creatures hunger is greater than or equal to 10:
        - Print a message stating the creatures starved to death.
        - Set the creatures is_alive attribute to False.
    - Elif the creatures dirtiness is greater than or equal to 10:
        - Print a message that the creature suffered an infection and died.
        - Set the creatures is_alive attribute to False.
    - Elif the creatures boredom is greater than or equal to 10:
        - Set the creatures boredom equal to 10.
        - Print a message stating the creature is bored and falling asleep.
        - Set the creatures is_sleeping attribute to True.
    - Elif the creatures tiredness is greater than or equal to 10:
        - Set the creatures tiredness equal to 10.
        - Print a message that the creature is sleepy and falling asleep.
        - Set the creatures is_sleeping attribute to True.
    -

Defining your functions

- Define a function show_menu() which takes one parameter, a creature object.
    - If the creatures is_sleeping attribute is True:
        - Get user input for a choice that represents a creature method call.
        - Only give them the option to press 6 if the creature is sleeping.
        - Hard code their choice to be 6 to take precaution.
        - A value of 6 will eventually call the awake() method for the creature. If the creature is sleeping, this is the only method we want the user to be able to call.
    - Else, the creature is not sleeping so give the user more options:
        - Enter 1 to eat.
        - Enter 2 to play.

        - Enter 3 to sleep.
        - Enter 4 to take a bath.
        - Enter 5 to forage for food.
        - Get user input for their choice.
    - Return this choice as a string.

- Define a function call_action() which takes two parameters, a creature object and a string representing a choice by the user.
    - If choice is equal to 1 call the creatures eat() method.
    - Elif choice is equal to 2 call the creatures play() method.
    - Elif choice is equal to 3 call the creatures sleep() method.
    - Elif choice is equal to 4 call the creatures clean() method.
    - Elif choice is equal to 5 call the creatures forage() method.
    - Elif choice is equal to 6 call the creatures awake() method.
    - Else print a message that the user entered a non valid choice.

Your main code

- Print a welcome message.
- Get user input for their difficulty level ranging from 1 to 5 and store it as an integer in a variable difficulty.
    - If the difficulty is greater than 5, set it equal to 5.
    - Elif the difficulty is less than 1, set it equal to 1.
- Create an active variable and set it to True. Use this variable to control a while loop.

- Get user input for the name of their creature.
- Create a creature object and store it in a variable player.
- Create a variable rounds and set it equal to 1.

- Create a while loop that will run as long as the creature is alive.
    - Print a message stating the current round.
    - Call creatures show_values() method.
    - Call the show_menu() function.
    - Call the call_action() function.

    - Print a message stating the round summary.
    - Call creatures show_values() method.
    - Prompt the user to press enter to continue.

    - Call the creatures increment_values() method.
    - Call the creatures kill() method.
    - Increment the round number by 1.

- Once the creature has died print an R.I.P message.
- Print a message informing the user how many rounds the creatures survived.
- Get user input for if they would like to play again.

- If not, set the active variable controlling the game loop to False and thank the user for playing.

Example Output:

```
Welcome to the Pythonagachi Simulator App
Please choose a difficulty level (1-5): 4
What name would you like to give your pet Pythonogachi: bobon

----------------------------------------------------------------------
Round #1

Creature Name: Bobon
Hunger (0-10): 0
Boredom (0-10): 0
Tiredness (0-10): 0
Dirtiness (0-10): 0

Food Inventory:  2 pieces
Current Status:  Awake

Enter (1) to eat.
Enter (2) to play.
Enter (3) to sleep.
Enter (4) to take a bath.
Enter (5) to forage for food.
What is your choice: 5
Bobon found 3 pieces of food!

Round #1 Summary:

Creature Name: Bobon
Hunger (0-10): 0
Boredom (0-10): 0
Tiredness (0-10): 0
Dirtiness (0-10): 2

Food Inventory:  5 pieces
Current Status:  Awake

Press (enter) to continue...

----------------------------------------------------------------------
Round #2
```

Creature Name: Bobon
Hunger (0-10): 0
Boredom (0-10): 1
Tiredness (0-10): 4
Dirtiness (0-10): 5

Food Inventory: 5 pieces
Current Status: Awake

Enter (1) to eat.
Enter (2) to play.
Enter (3) to sleep.
Enter (4) to take a bath.
Enter (5) to forage for food.
What is your choice: 5
Bobon found 3 pieces of food!

Round #2 Summary:

Creature Name: Bobon
Hunger (0-10): 0
Boredom (0-10): 1
Tiredness (0-10): 4
Dirtiness (0-10): 7

Food Inventory: 8 pieces
Current Status: Awake

Press (enter) to continue...

Round #3

Creature Name: Bobon
Hunger (0-10): 4
Boredom (0-10): 3
Tiredness (0-10): 4
Dirtiness (0-10): 8

Food Inventory: 8 pieces
Current Status: Awake

Enter (1) to eat.
Enter (2) to play.
Enter (3) to sleep.

Enter (4) to take a bath.
Enter (5) to forage for food.
What is your choice: 4
Bobon has taken a bath. All clean!

Round #3 Summary:

Creature Name: Bobon
Hunger (0-10): 4
Boredom (0-10): 3
Tiredness (0-10): 4
Dirtiness (0-10): 0

Food Inventory: 8 pieces
Current Status: Awake

Press (enter) to continue...

Round #4

Creature Name: Bobon
Hunger (0-10): 8
Boredom (0-10): 5
Tiredness (0-10): 5
Dirtiness (0-10): 4

Food Inventory: 8 pieces
Current Status: Awake

Enter (1) to eat.
Enter (2) to play.
Enter (3) to sleep.
Enter (4) to take a bath.
Enter (5) to forage for food.
What is your choice: 1
Yumm! Bobon ate a great meal!

Round #4 Summary:

Creature Name: Bobon
Hunger (0-10): 4
Boredom (0-10): 5
Tiredness (0-10): 5
Dirtiness (0-10): 4

Food Inventory: 7 pieces
Current Status: Awake

Press (enter) to continue...

Round #5

Creature Name: Bobon
Hunger (0-10): 6
Boredom (0-10): 8
Tiredness (0-10): 5
Dirtiness (0-10): 6

Food Inventory: 7 pieces
Current Status: Awake

Enter (1) to eat.
Enter (2) to play.
Enter (3) to sleep.
Enter (4) to take a bath.
Enter (5) to forage for food.
What is your choice: 2

Bobon wants to play a game.
Bobon is thinking of a number 0, 1, or 2.
What is your guess: 0
That is correct!!!

Round #5 Summary:

Creature Name: Bobon
Hunger (0-10): 6
Boredom (0-10): 5
Tiredness (0-10): 5
Dirtiness (0-10): 6

Food Inventory: 7 pieces
Current Status: Awake

Press (enter) to continue...

Round #6

Creature Name: Bobon
Hunger (0-10): 9
Boredom (0-10): 7
Tiredness (0-10): 6
Dirtiness (0-10): 7

Food Inventory: 7 pieces
Current Status: Awake

Enter (1) to eat.
Enter (2) to play.
Enter (3) to sleep.
Enter (4) to take a bath.
Enter (5) to forage for food.
What is your choice: 3
Zzzzzz....Zzzzzz....Zzzzzz....

Round #6 Summary:

Creature Name: Bobon
Hunger (0-10): 9
Boredom (0-10): 5
Tiredness (0-10): 3
Dirtiness (0-10): 7

Food Inventory: 7 pieces
Current Status: Sleeping

Press (enter) to continue...
Bobon has starved to death...

R.I.P.
Bobon survived a total of 6 rounds.
would you like to play again (y/n): n
Thank you for playing Pythonagachi!