Reyes, Maurey Shane

BSCS - C204

Finals Task 3. Simple Polymorphism

Problem. Chirp and Tweet

Create a simple program to demonstrate basic polymorphism with bird sounds.

Class - Bird:

- Methods:
  - `def make_sound(self) -> None`: An abstract method that represents making a sound. It doesn't have a specific implementation in the base class `Bird`.

Class - Sparrow (extends Bird):

- Methods:
  - `def make_sound(self) -> None`: Overrides the `make_sound` method from the base class `Bird`. It prints the sound "Chirp Chirp" when called.

Class - Parrot (extends Bird):

- Methods:
  - `def make_sound(self) -> None`: Overrides the `make_sound` method from the base class `Bird`. It prints the sound "Tweet Tweet" when called.

Class - BirdCage:

- Methods:
  - `def make_bird_sounds(self, birds: List) -> None`: Accepts a list of `Bird` objects as input. Iterates through the list of birds and calls the `make_sound` method on each bird to make its sound.

Note:

- *The test cases are not outputs of your main file but of a hidden test file. Create and implement the classes instructed to test your code.*
- *Each class should be defined in its own file, with the file name following camelCase conventions (e.g., `bankAccount.py`).*

CODE

bird.py × sparrow.py parrot.py birdCage.py main.py

```python
from abc import ABC, abstractmethod

class Bird(ABC):  6 usages

    @abstractmethod  1 usage
    def make_sound(self) -> None:
        pass

```

bird.py sparrow.py × parrot.py birdCage.py main.py

```python
from bird import Bird

class Sparrow(Bird):  2 usages

    def make_sound(self) -> None:  1 usage
        return "Chirp Chirp"

```

bird.py sparrow.py parrot.py × birdCage.py main.py

```python
from bird import Bird

class Parrot(Bird):  2 usages

    def make_sound(self) -> None:  1 usage
        return "Tweet Tweet"

```

bird.py sparrow.py parrot.py birdCage.py × main.py

```python
from typing import List
from bird import Bird

class BirdCage:  2 usages

    def make_bird_sounds(self, birds: List[Bird]) -> List[str]:  1 usage
        sounds = []
        for bird in birds:
            sounds.append(bird.make_sound())
        return sounds

```

bird.py sparrow.py parrot.py birdCage.py main.py ×

```python
from sparrow import Sparrow
from parrot import Parrot
from birdCage import BirdCage

s = Sparrow()
p = Parrot()

print(s.make_sound())  # Chirp Chirp
print(p.make_sound())  # Tweet Tweet

cage = BirdCage()
print(cage.make_bird_sounds([s, p]))  # ['Chirp Chirp', 'Tweet Tweet']

```

SAMPLE OUTPUT

```
C:\Users\Admin\PycharmProjects\PythonProject1\.venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\PythonProject1\main.py
Chirp Chirp
Tweet Tweet
['Chirp Chirp', 'Tweet Tweet']

Process finished with exit code 0
```