Mechure, Liann S. 7OOP1 BSCS - C204

Final Task 3. Polymorphism

Sample Problem

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.

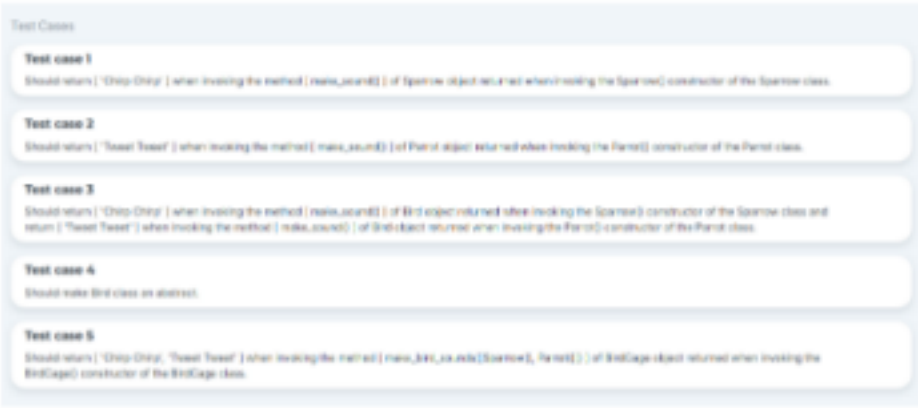
Class - BirdCage:

- 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`).*

TEST CASES:



Source Code

bird.py

```python
from abc import ABC, abstractmethod

class Bird(ABC):
    @abstractmethod
    def make_sound(self) -> str:
        """Abstract method for making a bird sound."""
        pass

```

parrot.py

```python
from bird import Bird

class Parrot(Bird):
    def make_sound(self) -> str:
        return "Tweet Tweet"

```

sparrow.py

```python
from bird import Bird

class Sparrow(Bird):
    def make_sound(self) -> str:
        return "Chirp Chirp"

```

birdCage.py

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

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

```

testCase

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

# Test Case 1
print(Sparrow().make_sound())

# Test Case 2
print(Parrot().make_sound())

# Test Case 3
birds = [Sparrow(), Parrot()]
for bird in birds:
    print(bird.make_sound())

# Test Case 5
cage = BirdCage()
for sound in cage.make_bird_sounds([Sparrow(), Parrot()]):
    print(sound)

```

Sample Output

```
Chirp Chirp
Tweet Tweet
Chirp Chirp
Tweet Tweet
Chirp Chirp
Tweet Tweet
PS C:\Users\Liann>
```