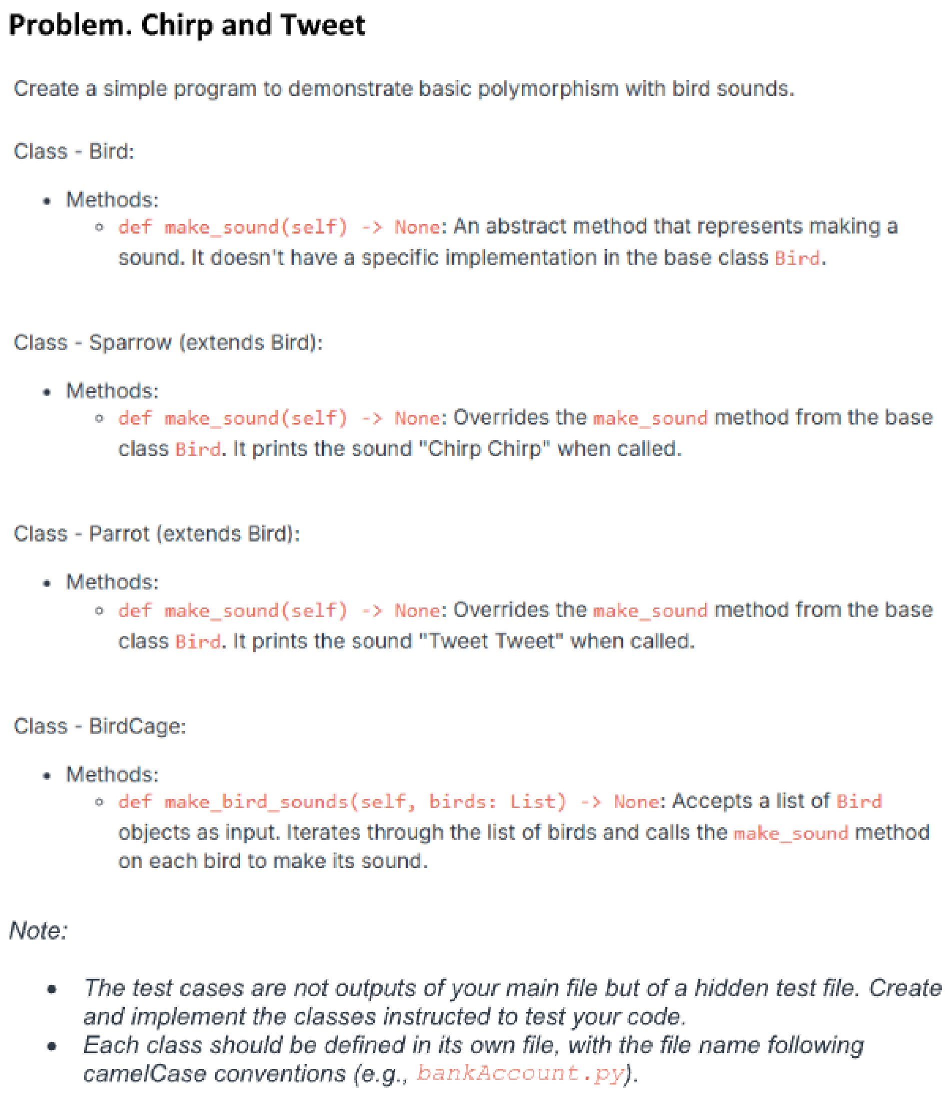
Cabenta, Joseph S.
BSCS – C204

November 23, 2025
7OOP

# FINAL LAB TASK 3

## I. PROBLEM



### 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.

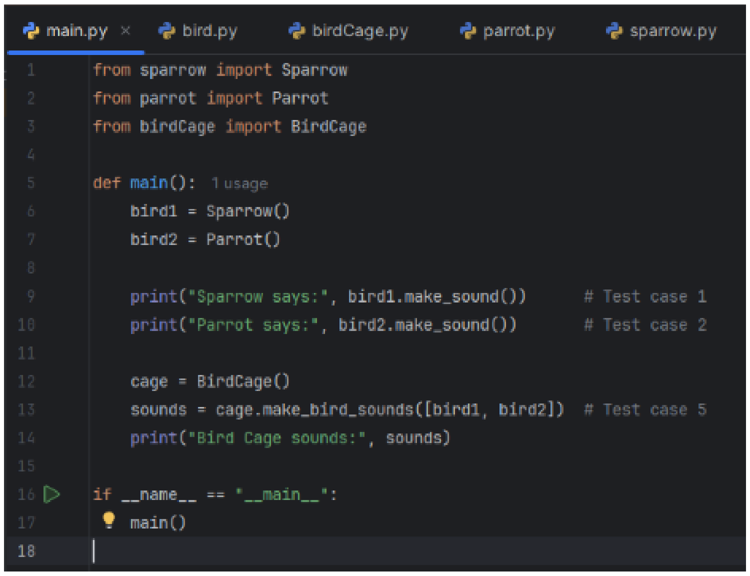
Class - BirdCage:

- 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.

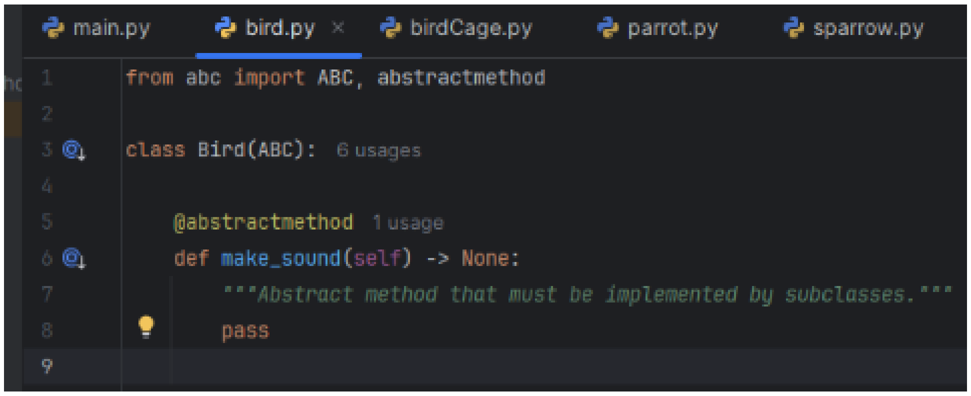
*Note:*

- *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`).*

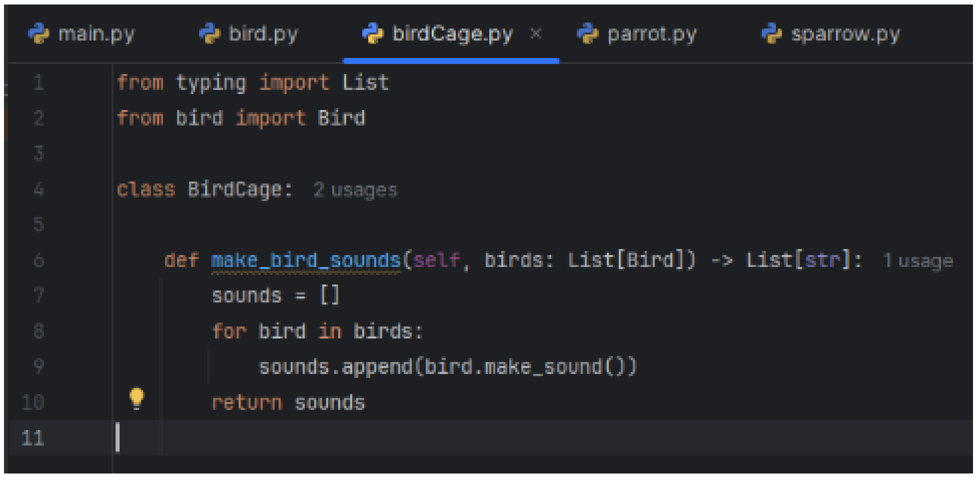
## II. CODE



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

def main():  1 usage
    bird1 = Sparrow()
    bird2 = Parrot()

    print("Sparrow says:", bird1.make_sound())      # Test case 1
    print("Parrot says:", bird2.make_sound())       # Test case 2

    cage = BirdCage()
    sounds = cage.make_bird_sounds([bird1, bird2])  # Test case 5
    print("Bird Cage sounds:", sounds)

if __name__ == "__main__":
    main()
```

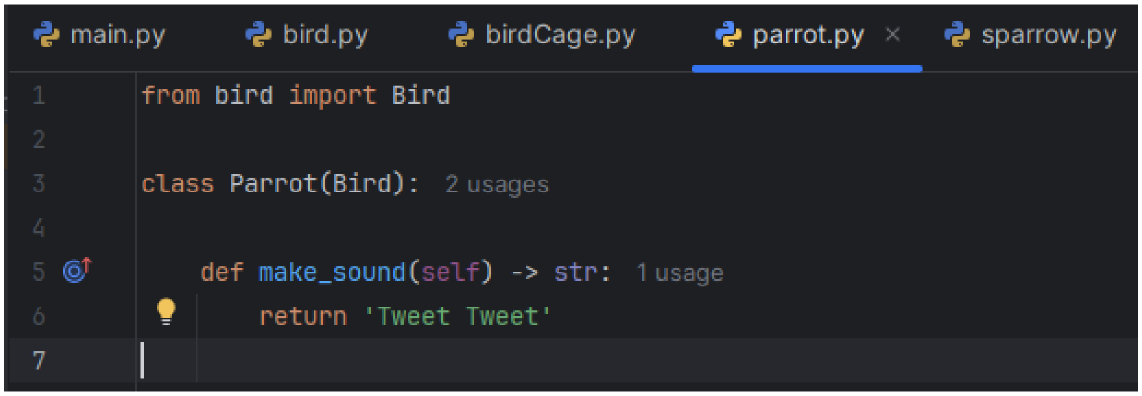


```python
from abc import ABC, abstractmethod

class Bird(ABC):  6 usages

    @abstractmethod  1 usage
    def make_sound(self) -> None:
        """Abstract method that must be implemented by subclasses."""
        pass
```

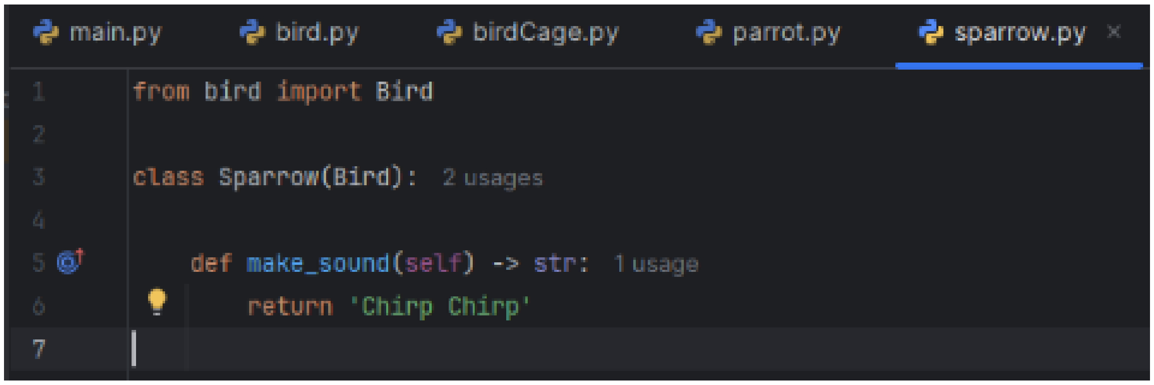


```python
from typing import List
from bird import Bird

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
```

Cabenta, Joseph S. November 23, 2025
BSCS – C204 7OOP

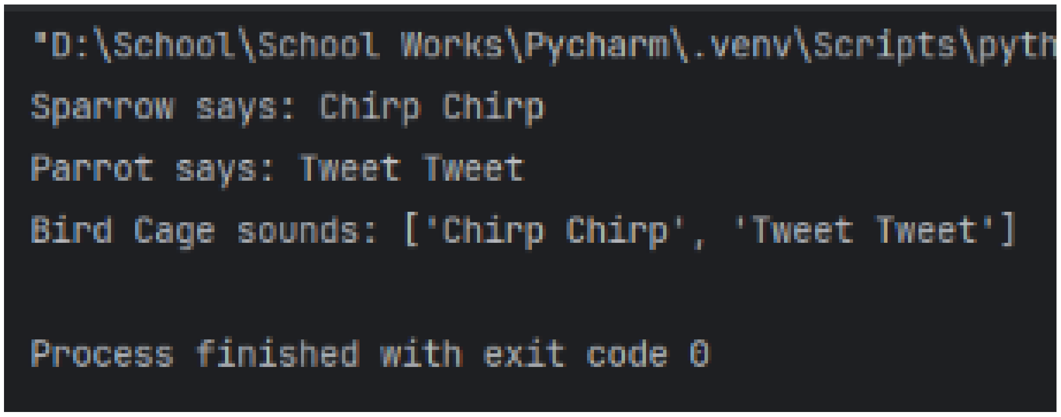

main.py bird.py birdCage.py parrot.py sparrow.py

```python
from bird import Bird

class Parrot(Bird):  2 usages

    def make_sound(self) -> str:  1 usage
        return 'Tweet Tweet'

```


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

```python
from bird import Bird

class Sparrow(Bird):  2 usages

    def make_sound(self) -> str:  1 usage
        return 'Chirp Chirp'

```

## III. SAMPLE OUTPUT



```
"D:\School\School Works\Pycharm\.venv\Scripts\pyth
Sparrow says: Chirp Chirp
Parrot says: Tweet Tweet
Bird Cage sounds: ['Chirp Chirp', 'Tweet Tweet']

Process finished with exit code 0
```