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.

CODE:

```
from typing import List
from abc import ABC, abstractmethod


class Bird(ABC):
    @abstractmethod
    def make_sound(self) -> None:
        pass


class Sparrow(Bird):
    def make_sound(self) -> None:
```

```
        print("Chirp Chirp")


class Parrot(Bird):
    def make_sound(self) -> None:
        print("Tweet Tweet")


class BirdCage:
    def make_bird_sounds(self, birds: List[Bird]) -> None:
        for bird in birds:
            bird.make_sound()


if __name__ == "__main__":
    s = Sparrow()
    p = Parrot()
    cage = BirdCage()

    s.make_sound()
    p.make_sound()

    cage.make_bird_sounds([s, p])
```

OUTPUT: