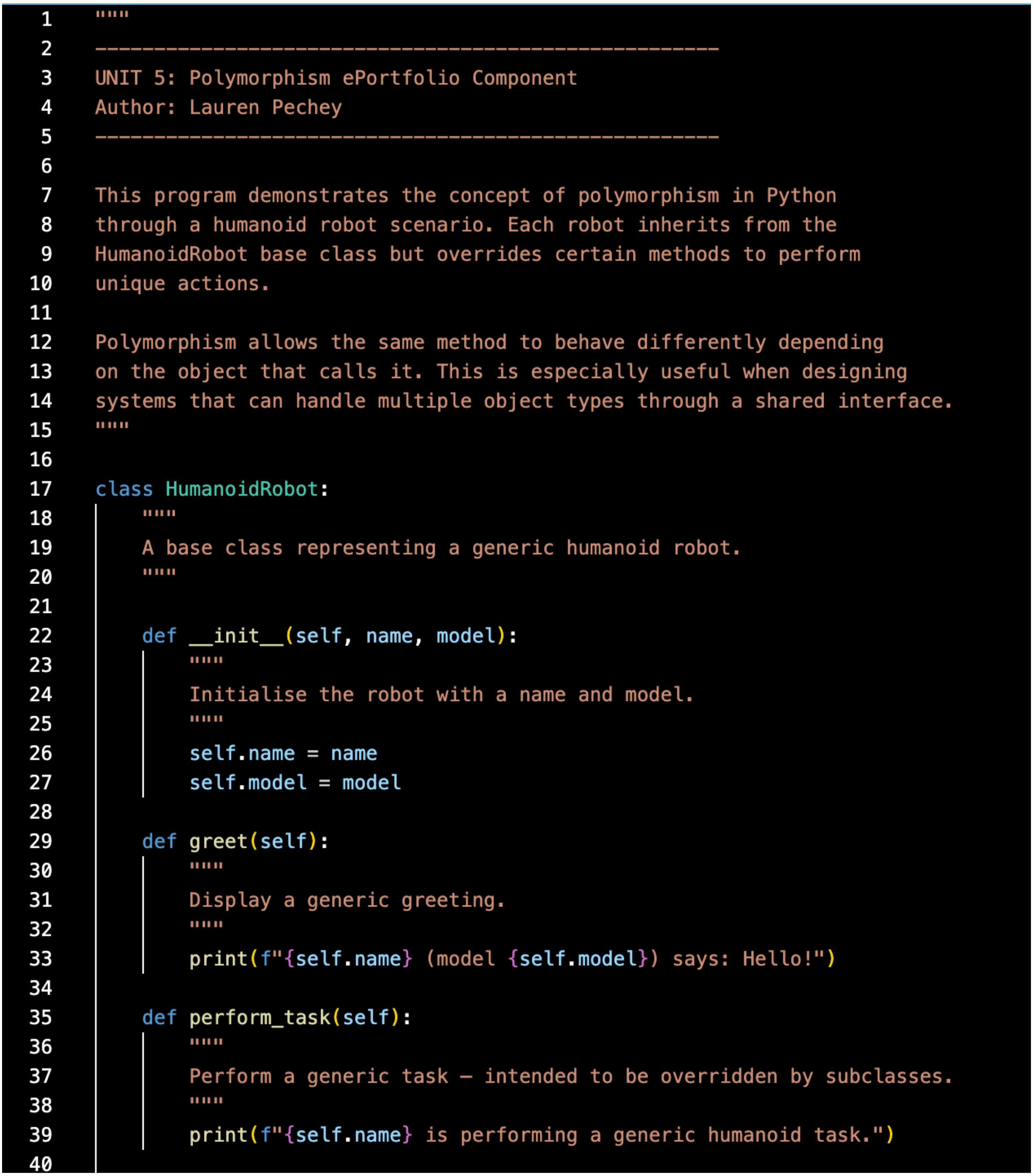
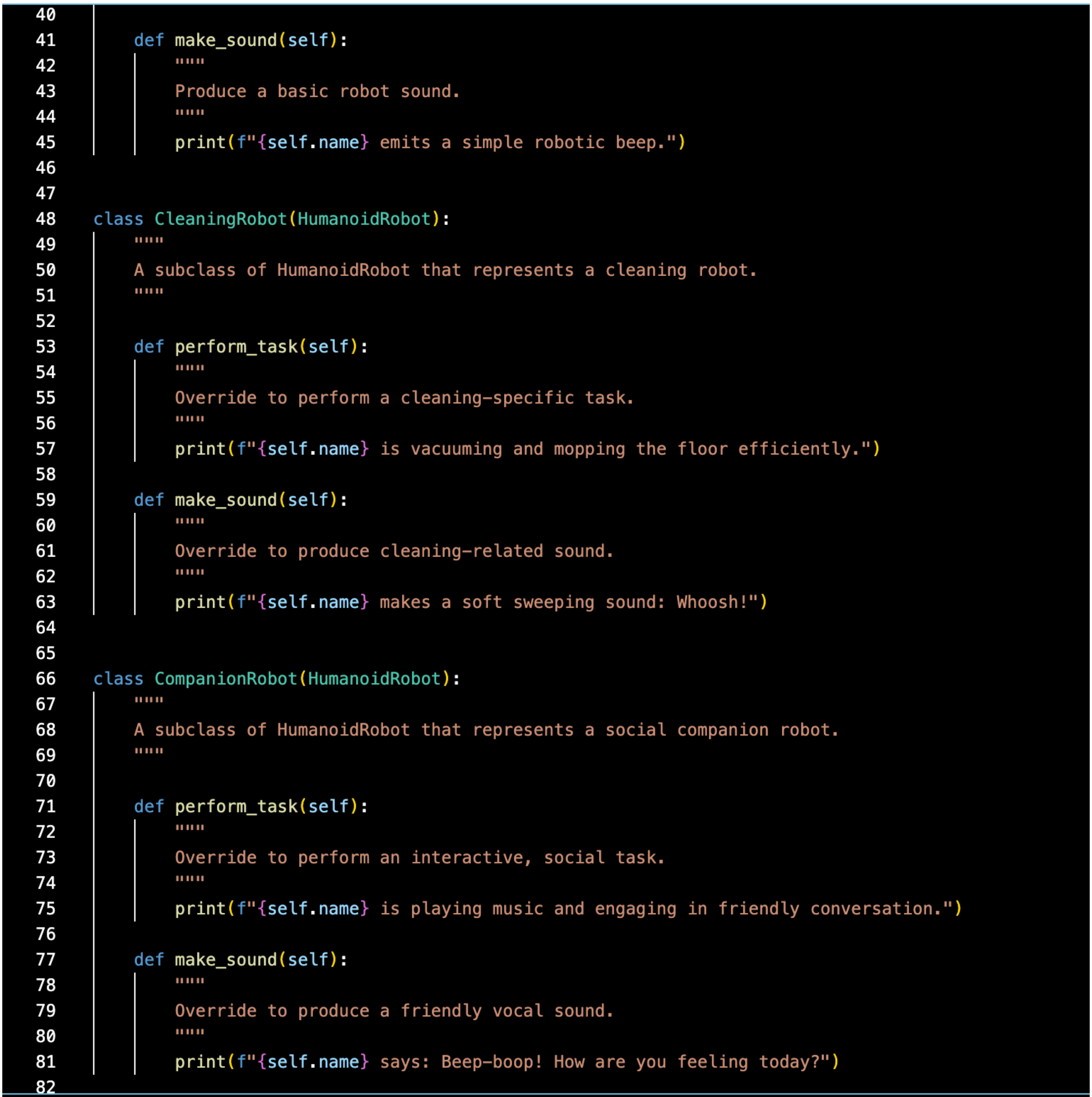
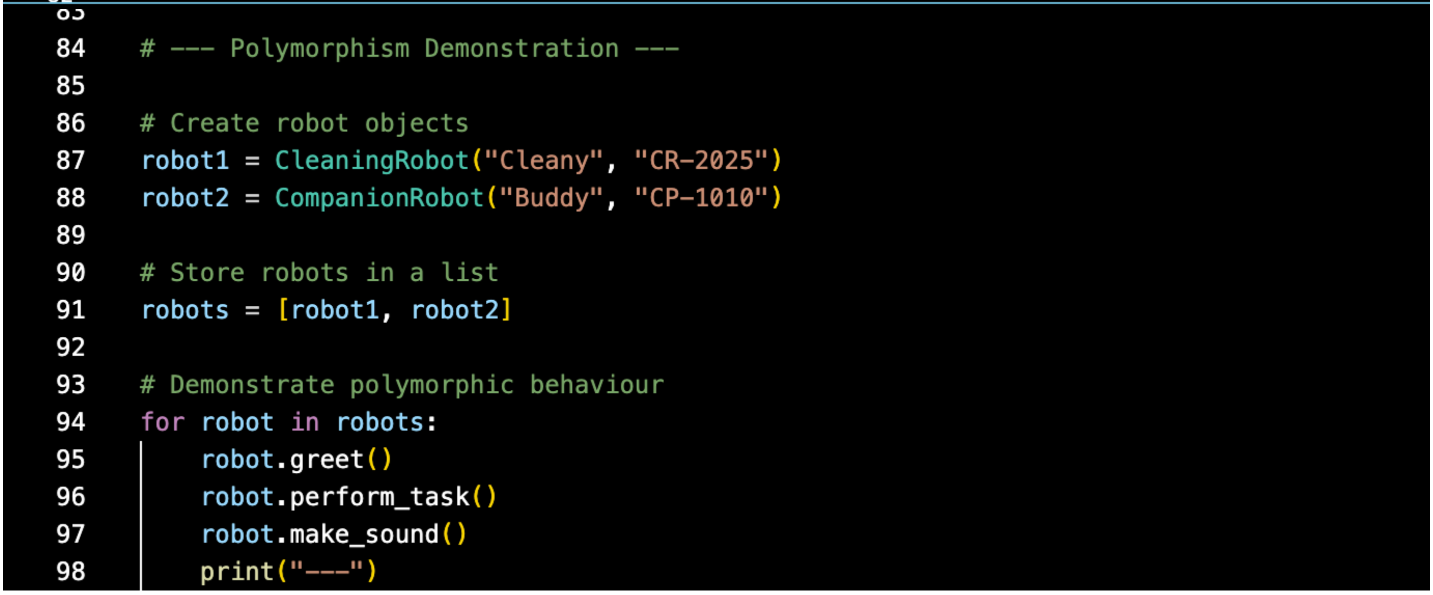


```python
"""
-----------------------------------------------------
UNIT 5: Polymorphism ePortfolio Component
Author: Lauren Pechey
-----------------------------------------------------

This program demonstrates the concept of polymorphism in Python
through a humanoid robot scenario. Each robot inherits from the
HumanoidRobot base class but overrides certain methods to perform
unique actions.

Polymorphism allows the same method to behave differently depending
on the object that calls it. This is especially useful when designing
systems that can handle multiple object types through a shared interface.
"""

class HumanoidRobot:
    """
    A base class representing a generic humanoid robot.
    """

    def __init__(self, name, model):
        """
        Initialise the robot with a name and model.
        """
        self.name = name
        self.model = model

    def greet(self):
        """
        Display a generic greeting.
        """
        print(f"{self.name} (model {self.model}) says: Hello!")

    def perform_task(self):
        """
        Perform a generic task – intended to be overridden by subclasses.
        """
        print(f"{self.name} is performing a generic humanoid task.")

```



```python

    def make_sound(self):
        """
        Produce a basic robot sound.
        """
        print(f"{self.name} emits a simple robotic beep.")


class CleaningRobot(HumanoidRobot):
    """
    A subclass of HumanoidRobot that represents a cleaning robot.
    """

    def perform_task(self):
        """
        Override to perform a cleaning-specific task.
        """
        print(f"{self.name} is vacuuming and mopping the floor efficiently.")

    def make_sound(self):
        """
        Override to produce cleaning-related sound.
        """
        print(f"{self.name} makes a soft sweeping sound: Whoosh!")


class CompanionRobot(HumanoidRobot):
    """
    A subclass of HumanoidRobot that represents a social companion robot.
    """

    def perform_task(self):
        """
        Override to perform an interactive, social task.
        """
        print(f"{self.name} is playing music and engaging in friendly conversation.")

    def make_sound(self):
        """
        Override to produce a friendly vocal sound.
        """
        print(f"{self.name} says: Beep-boop! How are you feeling today?")

```



```python

# --- Polymorphism Demonstration ---

# Create robot objects
robot1 = CleaningRobot("Cleany", "CR-2025")
robot2 = CompanionRobot("Buddy", "CP-1010")

# Store robots in a list
robots = [robot1, robot2]

# Demonstrate polymorphic behaviour
for robot in robots:
    robot.greet()
    robot.perform_task()
    robot.make_sound()
    print("---")
```