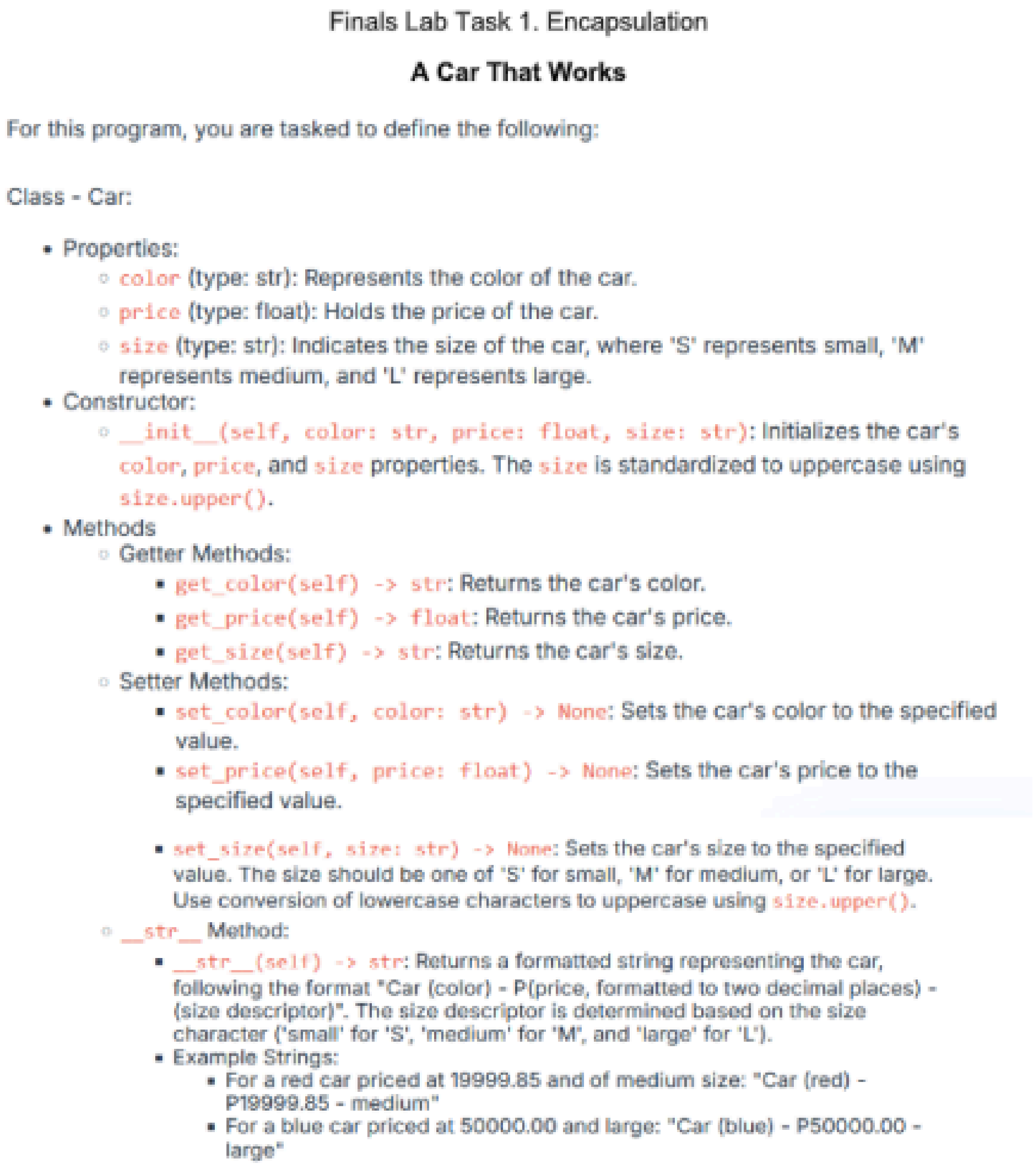
Mechure, Liann S. 7OOP1 BSCS – C204

Finals Lab Task #1 – Encapsulation

A Car That Works

Sample Problem:



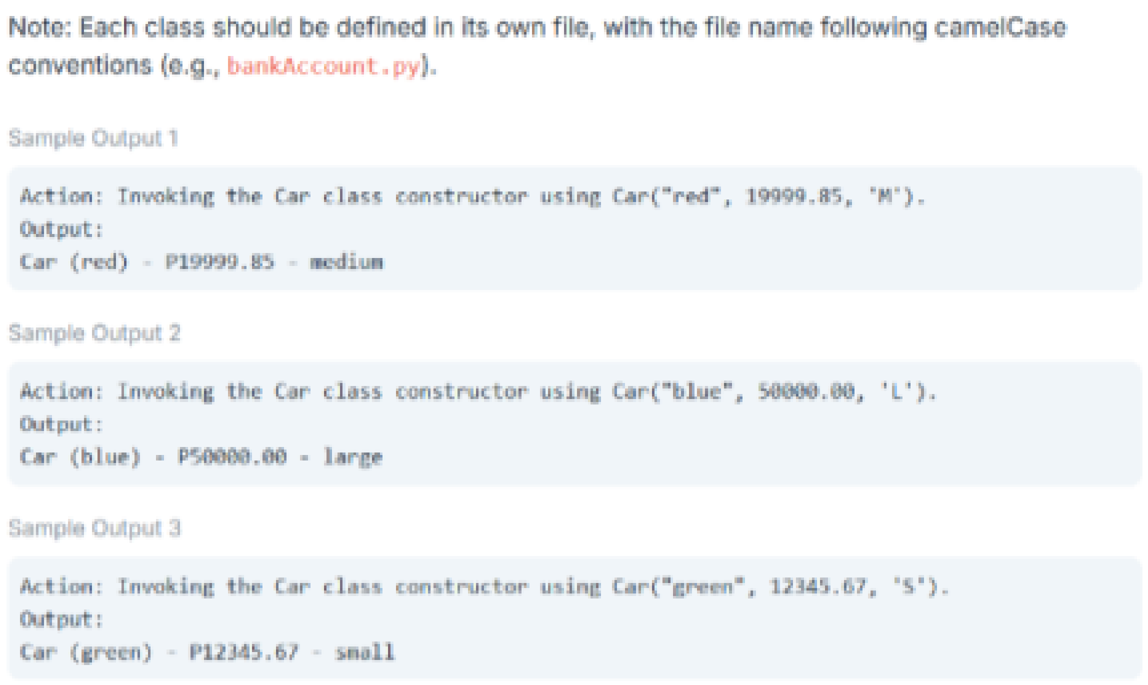
Finals Lab Task 1. Encapsulation

A Car That Works

For this program, you are tasked to define the following:

Class - Car:

- Properties:
  - `color` (type: str): Represents the color of the car.
  - `price` (type: float): Holds the price of the car.
  - `size` (type: str): Indicates the size of the car, where 'S' represents small, 'M' represents medium, and 'L' represents large.
- Constructor:
  - `__init__(self, color: str, price: float, size: str)`: Initializes the car's `color`, `price`, and `size` properties. The `size` is standardized to uppercase using `size.upper()`.
- Methods
  - Getter Methods:
    - `get_color(self) -> str`: Returns the car's color.
    - `get_price(self) -> float`: Returns the car's price.
    - `get_size(self) -> str`: Returns the car's size.
  - Setter Methods:
    - `set_color(self, color: str) -> None`: Sets the car's color to the specified value.
    - `set_price(self, price: float) -> None`: Sets the car's price to the specified value.
    - `set_size(self, size: str) -> None`: Sets the car's size to the specified value. The size should be one of 'S' for small, 'M' for medium, or 'L' for large. Use conversion of lowercase characters to uppercase using `size.upper()`.
  - `__str__` Method:
    - `__str__(self) -> str`: Returns a formatted string representing the car, following the format "Car (color) - P(price, formatted to two decimal places) - (size descriptor)". The size descriptor is determined based on the size character ('small' for 'S', 'medium' for 'M', and 'large' for 'L').
    - Example Strings:
      - For a red car priced at 19999.85 and of medium size: "Car (red) - P19999.85 - medium"
      - For a blue car priced at 50000.00 and large: "Car (blue) - P50000.00 - large"



Note: Each class should be defined in its own file, with the file name following camelCase conventions (e.g., `bankAccount.py`).

Sample Output 1

```
Action: Invoking the Car class constructor using Car("red", 19999.85, 'M').
Output:
Car (red) - P19999.85 - medium
```

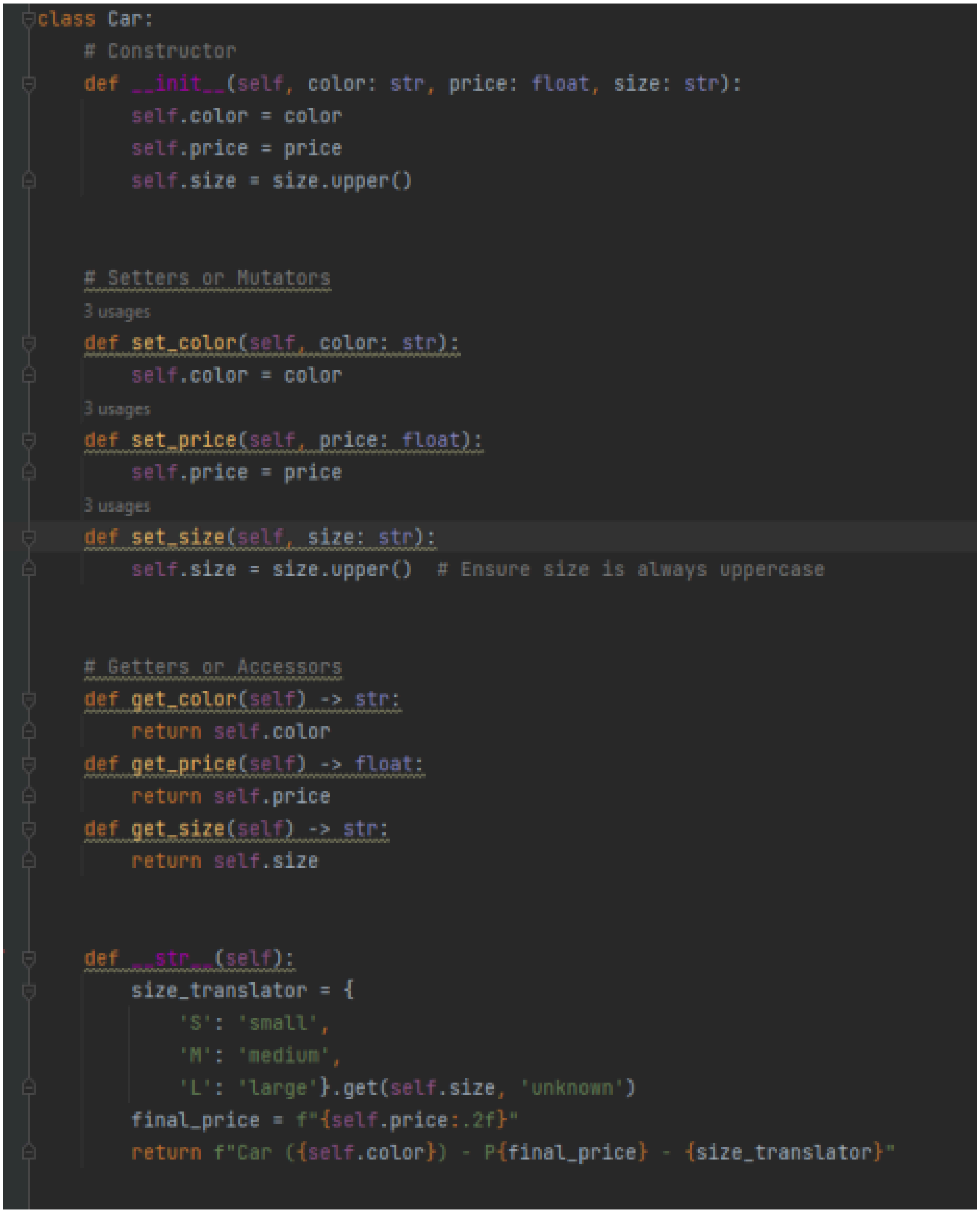
Sample Output 2

```
Action: Invoking the Car class constructor using Car("blue", 50000.00, 'L').
Output:
Car (blue) - P50000.00 - large
```

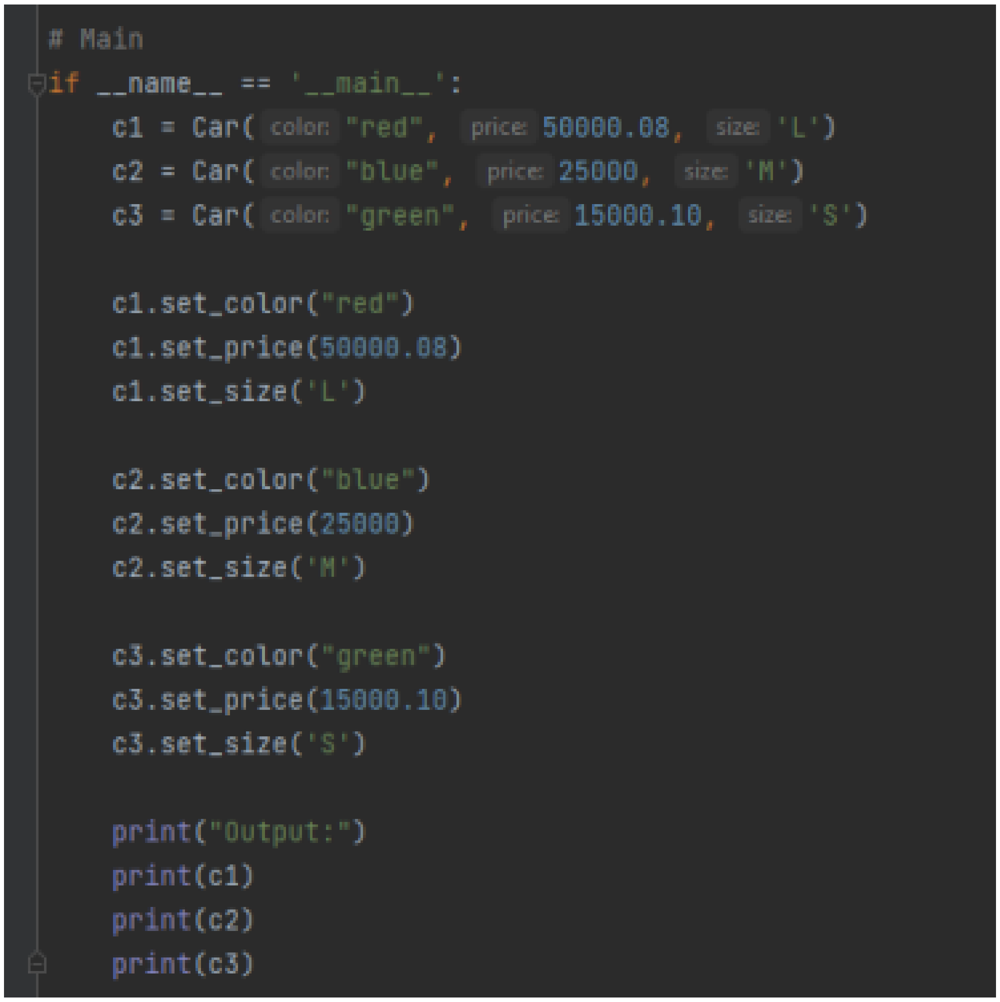
Sample Output 3

```
Action: Invoking the Car class constructor using Car("green", 12345.67, 'S').
Output:
Car (green) - P12345.67 - small
```

Source Code:

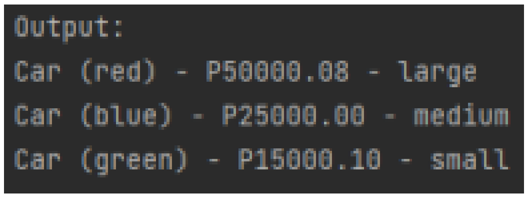


```
class Car:
    # Constructor
    def __init__(self, color: str, price: float, size: str):
        self.color = color
        self.price = price
        self.size = size.upper()


    # Setters or Mutators
    3 usages
    def set_color(self, color: str):
        self.color = color
    3 usages
    def set_price(self, price: float):
        self.price = price
    3 usages
    def set_size(self, size: str):
        self.size = size.upper()  # Ensure size is always uppercase


    # Getters or Accessors
    def get_color(self) -> str:
        return self.color
    def get_price(self) -> float:
        return self.price
    def get_size(self) -> str:
        return self.size


    def __str__(self):
        size_translator = {
            'S': 'small',
            'M': 'medium',
            'L': 'large'}.get(self.size, 'unknown')
        final_price = f"{self.price:.2f}"
        return f"Car ({self.color}) - P{final_price} - {size_translator}"
```



```
# Main
if __name__ == '__main__':
    c1 = Car( color: "red",  price: 50000.08,  size: 'L')
    c2 = Car( color: "blue",  price: 25000,  size: 'M')
    c3 = Car( color: "green",  price: 15000.10,  size: 'S')

    c1.set_color("red")
    c1.set_price(50000.08)
    c1.set_size('L')

    c2.set_color("blue")
    c2.set_price(25000)
    c2.set_size('M')

    c3.set_color("green")
    c3.set_price(15000.10)
    c3.set_size('S')

    print("Output:")
    print(c1)
    print(c2)
    print(c3)
```

Sample Output:



```
Output:
Car (red) - P50000.08 - large
Car (blue) - P25000.00 - medium
Car (green) - P15000.10 - small
```