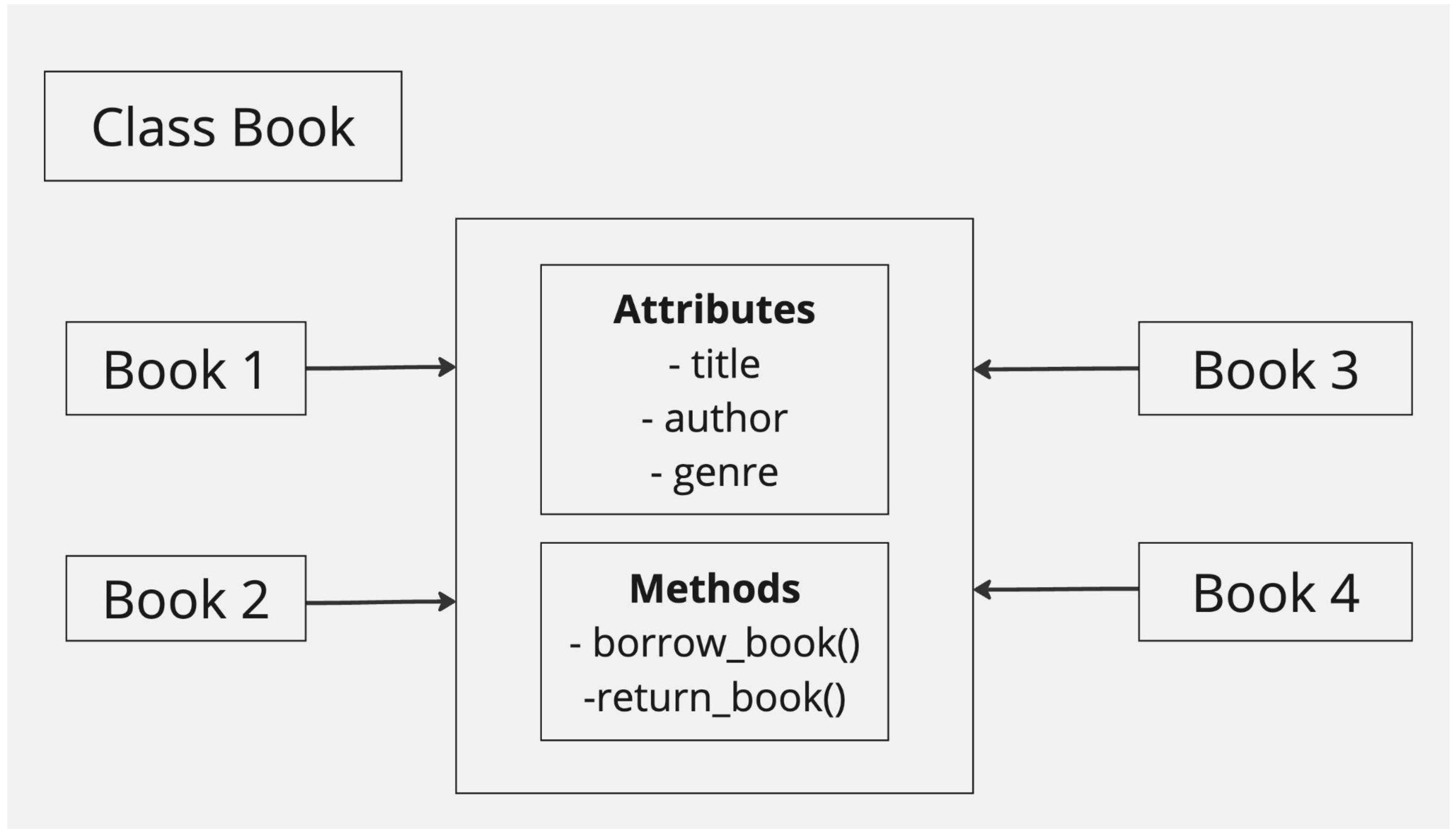
Newton School of Technology

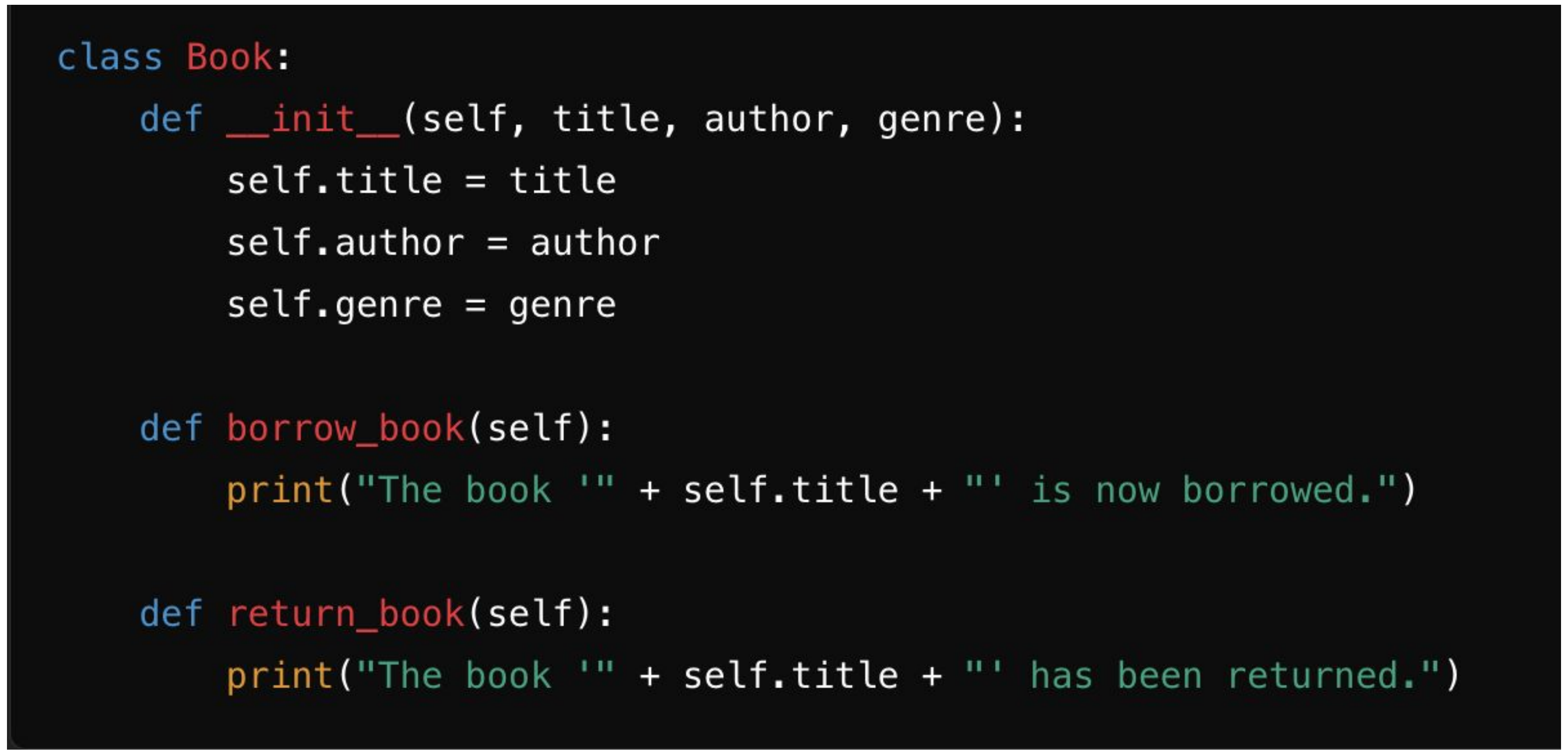
Object-Oriented Programming

by Gladden Rumao

CSA101 : Problem Solving with Programming

Newton School of Technology

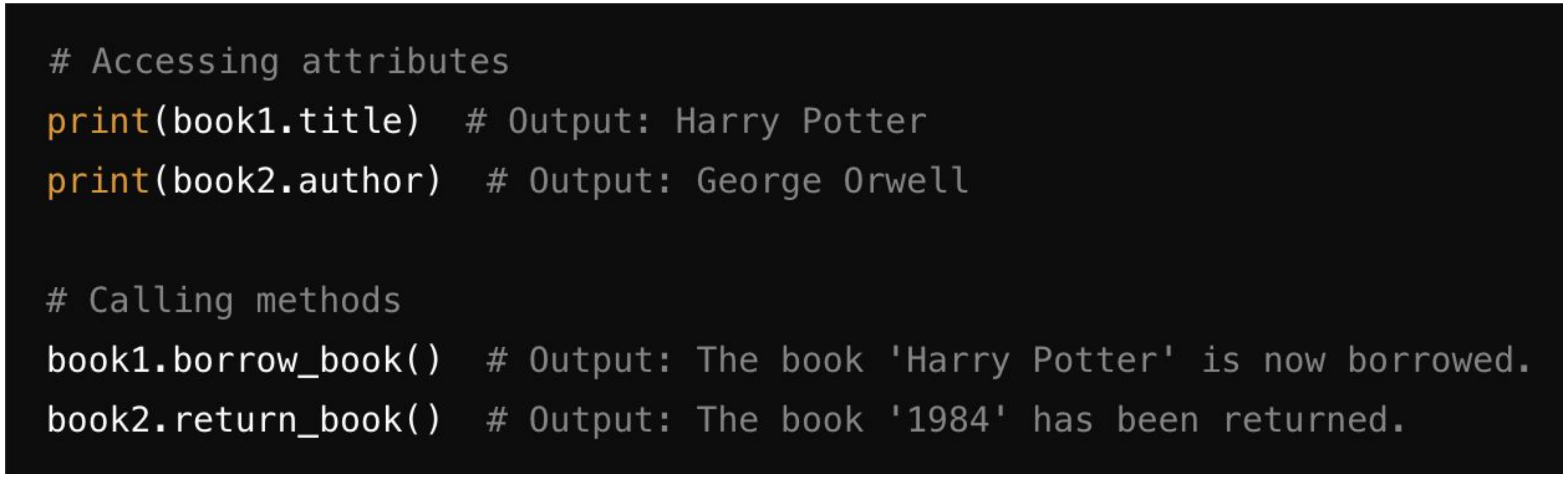




```python
class Book:
    def __init__(self, title, author, genre):
        self.title = title
        self.author = author
        self.genre = genre

    def borrow_book(self):
        print("The book '" + self.title + "' is now borrowed.")

    def return_book(self):
        print("The book '" + self.title + "' has been returned.")
```

Accessing Attributes and Methods :

Attributes: You can access attributes of an object using object.attribute.
Methods: You can call methods using object.method().



```python
# Accessing attributes
print(book1.title)  # Output: Harry Potter
print(book2.author)  # Output: George Orwell

# Calling methods
book1.borrow_book()  # Output: The book 'Harry Potter' is now borrowed.
book2.return_book()  # Output: The book '1984' has been returned.
```

Instance Attribute vs Class Attribute

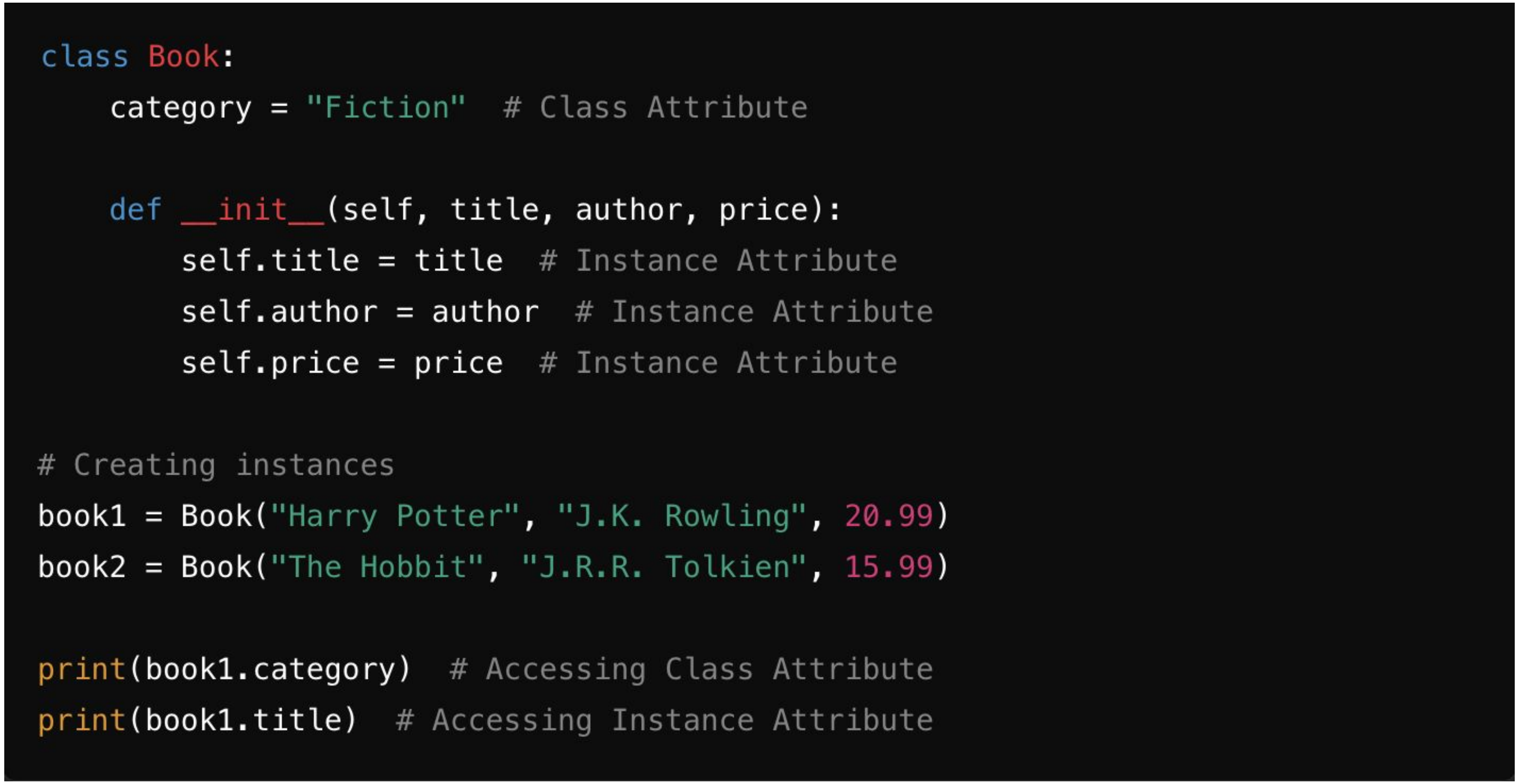
Instance Attribute

- **Defined** inside the `__init__` method.
- **Unique** for each object created from the class.
- **Accessed** using the object (e.g., `book1.title`).

Class Attribute

- **Defined** directly in the class (outside `__init__`).
- **Shared** by all instances of the class.
- **Accessed** using the class name or object (e.g., `Book.category` or `book1.category`).

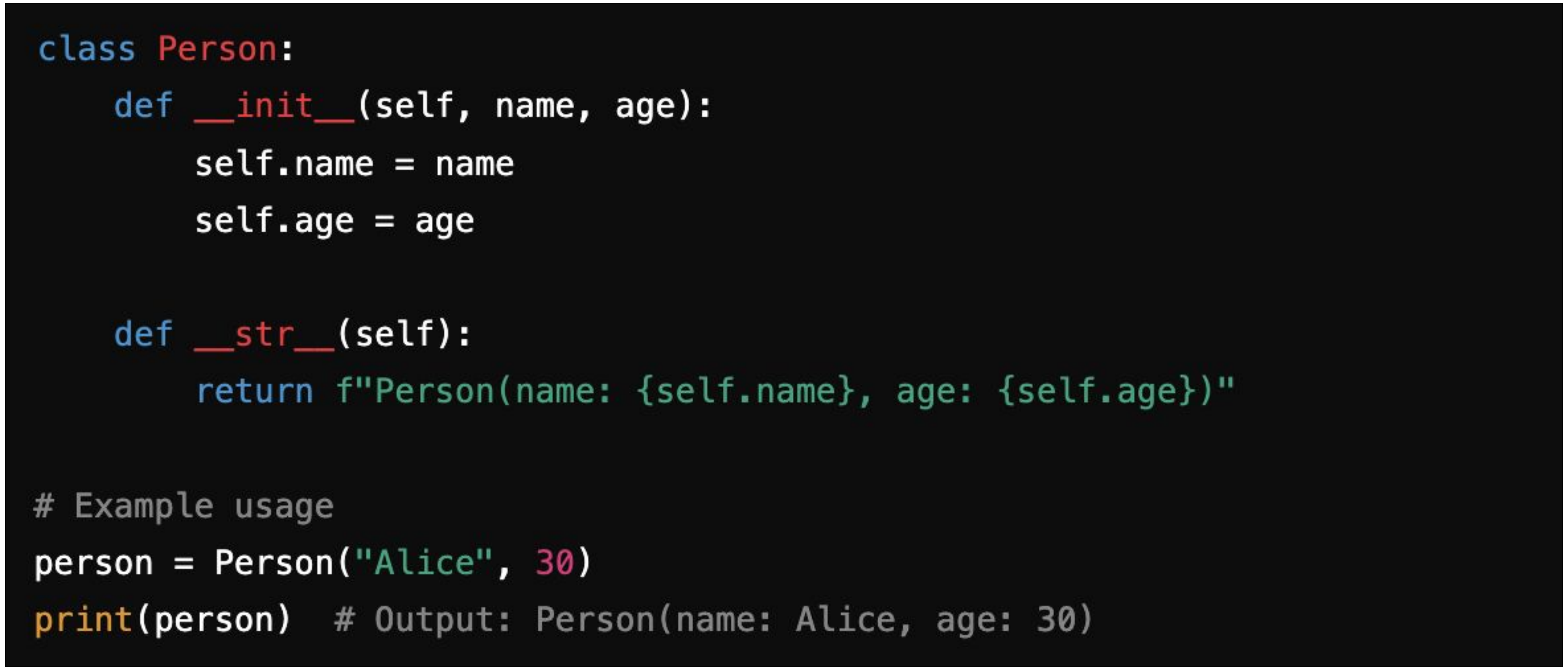
Instance Attribute vs Class Attribute



```python
class Book:
    category = "Fiction"  # Class Attribute

    def __init__(self, title, author, price):
        self.title = title  # Instance Attribute
        self.author = author  # Instance Attribute
        self.price = price  # Instance Attribute

# Creating instances
book1 = Book("Harry Potter", "J.K. Rowling", 20.99)
book2 = Book("The Hobbit", "J.R.R. Tolkien", 15.99)

print(book1.category)  # Accessing Class Attribute
print(book1.title)  # Accessing Instance Attribute
```

__str__ method

- The **__str__ method** in Python is a special method used to **provide a readable, user-friendly string representation of an object.**
- When print() is called on an object, the __str__ method is invoked if it is defined.



```python
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def __str__(self):
        return f"Person(name: {self.name}, age: {self.age})"

# Example usage
person = Person("Alice", 30)
print(person)  # Output: Person(name: Alice, age: 30)
```

Thank You!