Experiment No 2

Title:

Write and execute python program for implementation of vehicle information display using constructors and destructors

Objectives:

1. Understand the concepts of constructors and destructors in Python.
2. Implement a class for displaying vehicle information.
3. Utilize constructors to initialize vehicle attributes.
4. Utilize destructors to release resources when the object is destroyed.

Theory:

Constructors and destructors are special methods in object-oriented programming languages like Python. They play a crucial role in initializing objects and releasing resources associated with those objects, respectively.

1. Constructors: Constructors are special methods in a class that are automatically called when an object is created. In Python, the constructor method is called **__init__**. The primary purpose of a constructor is to initialize the object's attributes or perform any setup required for the object. Constructors allow us to ensure that newly created objects are in a valid state.
2. Destructors: Destructors are special methods in a class that are automatically called when an object is destroyed or garbage collected. In Python, the destructor method is called **__del__**. The primary purpose of a destructor is to release any resources or perform cleanup operations associated with the object before it is removed from memory. Destructors are useful for managing resources such as file handles, network connections, or database connections.

Constructors and destructors are fundamental concepts in object-oriented programming (OOP) that contribute to the concept of encapsulation, where data and methods are bound together within a class. Constructors ensure that objects are properly initialized with valid data, while destructors help in releasing resources and performing cleanup tasks when objects are no longer needed. By utilizing constructors and destructors effectively, developers can ensure robust memory management and resource handling in their Python programs, enhancing code reliability and maintainability. Additionally, constructors and destructors facilitate the implementation of RAII (Resource Acquisition Is Initialization) principles, promoting safer and more predictable object behavior throughout the program's lifecycle.

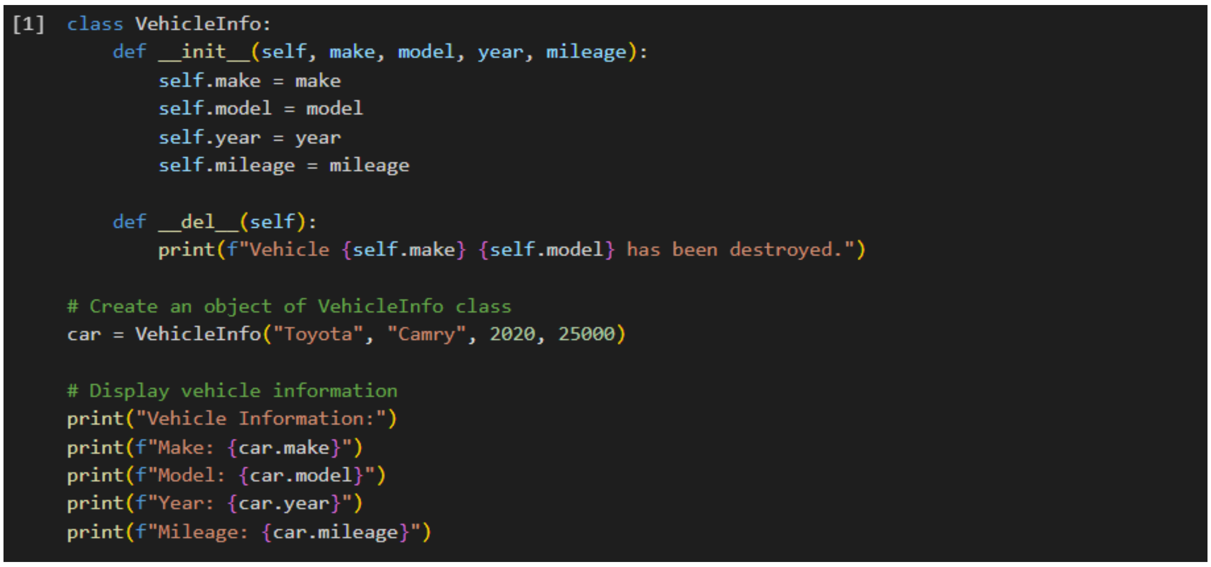
Key Concepts:

1. Constructors: Special methods used for initializing objects.
2. Destructors: Special methods used for releasing resources and performing cleanup operations when an object is destroyed.
3. Class: A blueprint for creating objects that defines attributes and methods.
4. Encapsulation: Binding data and methods that operate on the data within a single unit.
5. Python classes and objects.

Algorithm:

1. Define a class named VehicleInfo.
2. Implement the constructor (**init**) to initialize the vehicle attributes such as make, model, year, and mileage.
3. Implement the destructor (**del**) to print a message indicating the destruction of the object.
4. Create an object of the VehicleInfo class and display its attributes.

Program :



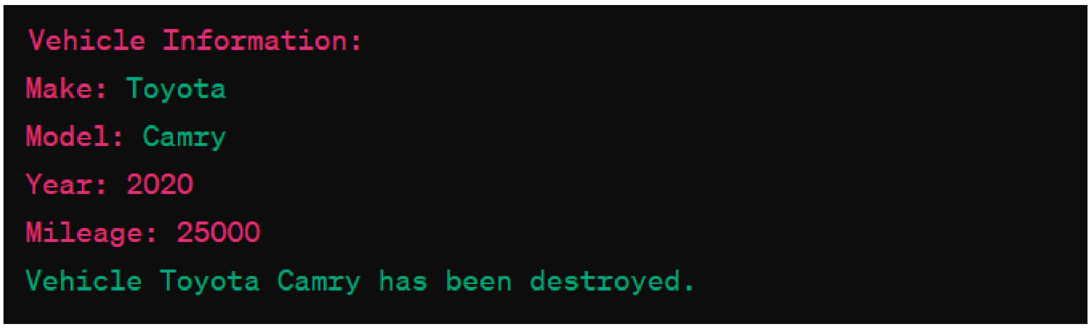
```
[1]  class VehicleInfo:
         def __init__(self, make, model, year, mileage):
             self.make = make
             self.model = model
             self.year = year
             self.mileage = mileage

         def __del__(self):
             print(f"Vehicle {self.make} {self.model} has been destroyed.")

     # Create an object of VehicleInfo class
     car = VehicleInfo("Toyota", "Camry", 2020, 25000)

     # Display vehicle information
     print("Vehicle Information:")
     print(f"Make: {car.make}")
     print(f"Model: {car.model}")
     print(f"Year: {car.year}")
     print(f"Mileage: {car.mileage}")
```

Output :



```
Vehicle Information:
Make: Toyota
Model: Camry
Year: 2020
Mileage: 25000
Vehicle Toyota Camry has been destroyed.
```

Conclusion:

In this lab experiment, we implemented a Python program to display vehicle information using constructors and destructors. Constructors were utilized to initialize the attributes of the VehicleInfo class, and destructors were used to release resources when the object was destroyed. This exercise demonstrated the importance of constructors and destructors in managing object lifecycle and resource cleanup in Python programs.