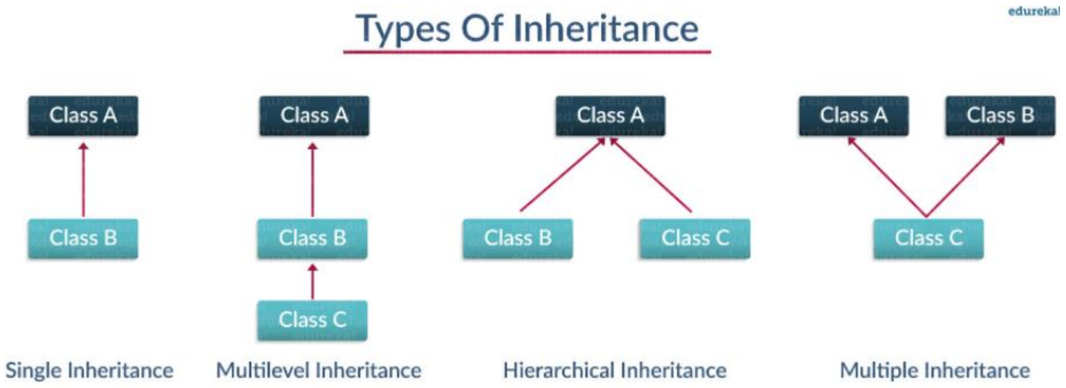
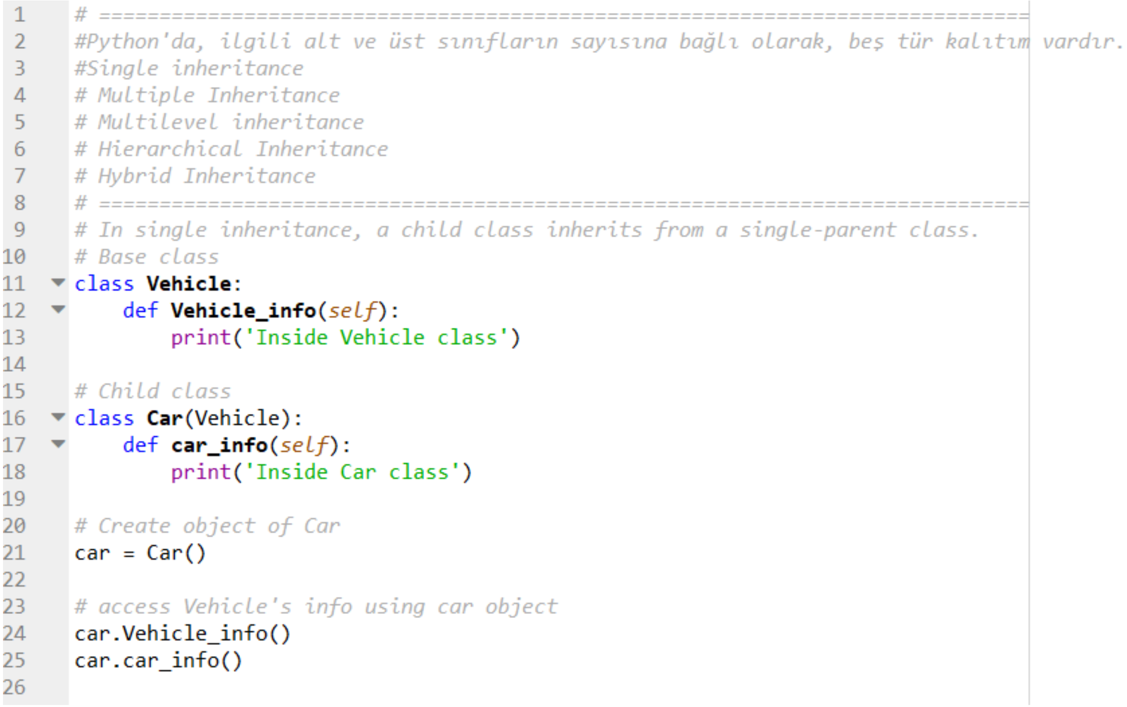
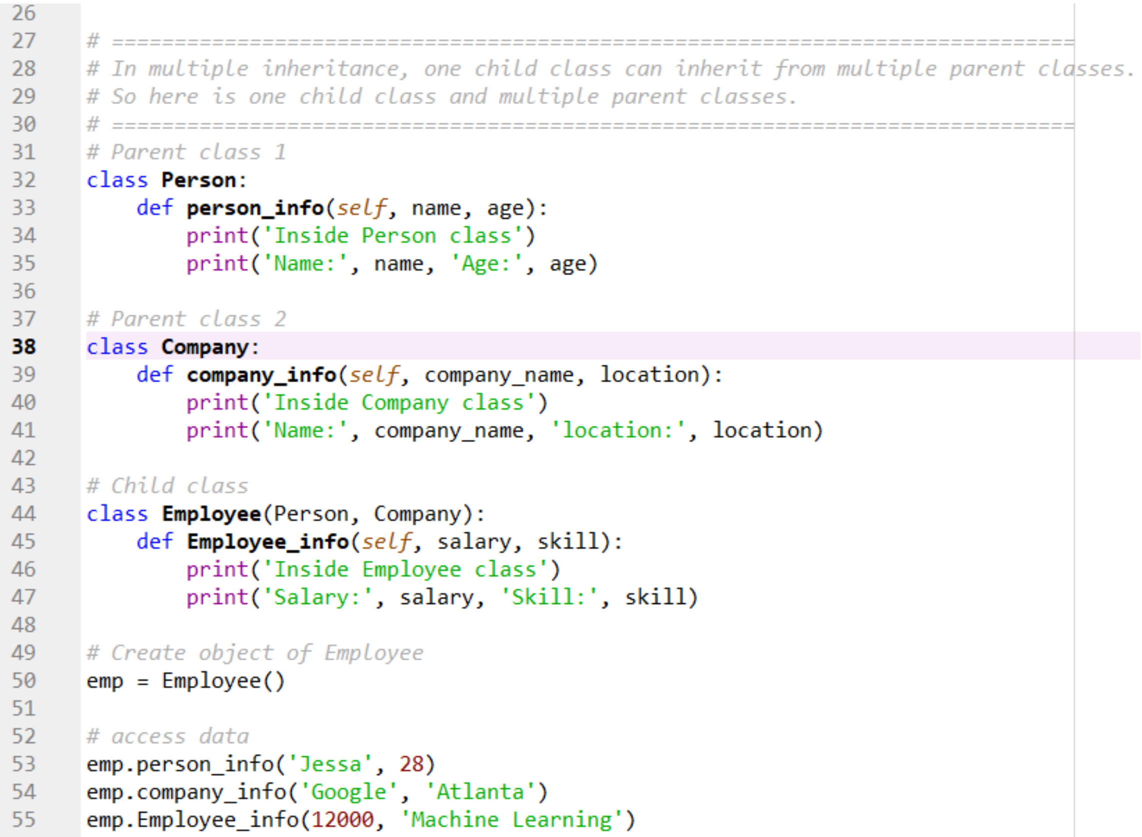
Miras (Inheritance) nesne tabanlı programlamada güçlü bir özelliktir. Varolan bir sınıfı çok az veya hiç değiştirmeden yeni bir sınıf tanımlamaya atıfta bulunur. Yeni sınıfa türetilmiş (veya çocuk) sınıf denir ve onu devraldığı sınıfa taban (veya ana sınıf) denir. Miras kelimesini tanımı olarak, en basit haliyle birinden başka birine kalan varlık anlamına gelir. Programlama tarafındaki anlamı ise; bir class içersindeki elemanların (property, metod) farklı nesneler tarafından kullanılabilmesini sağlayan yapıdır

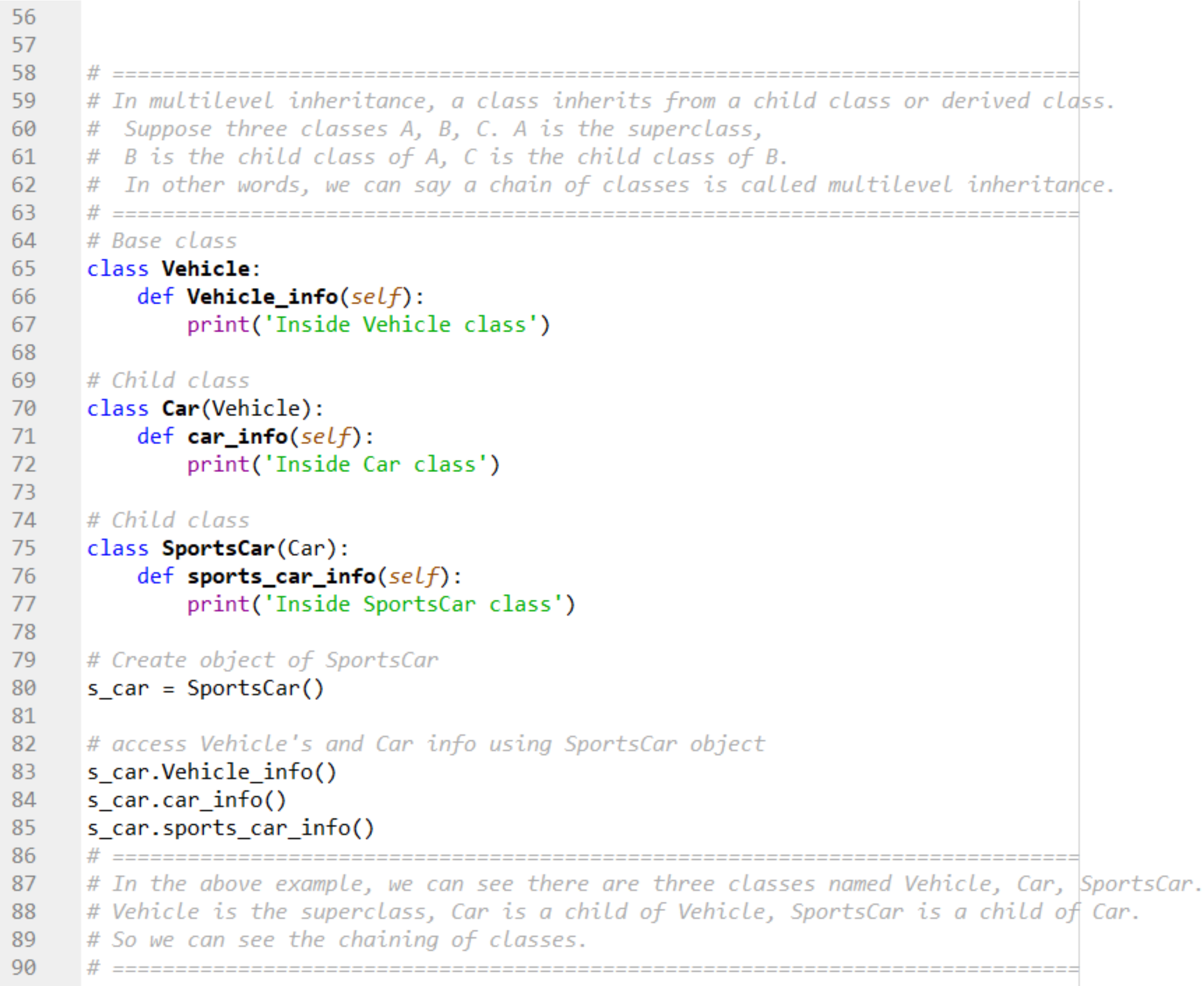




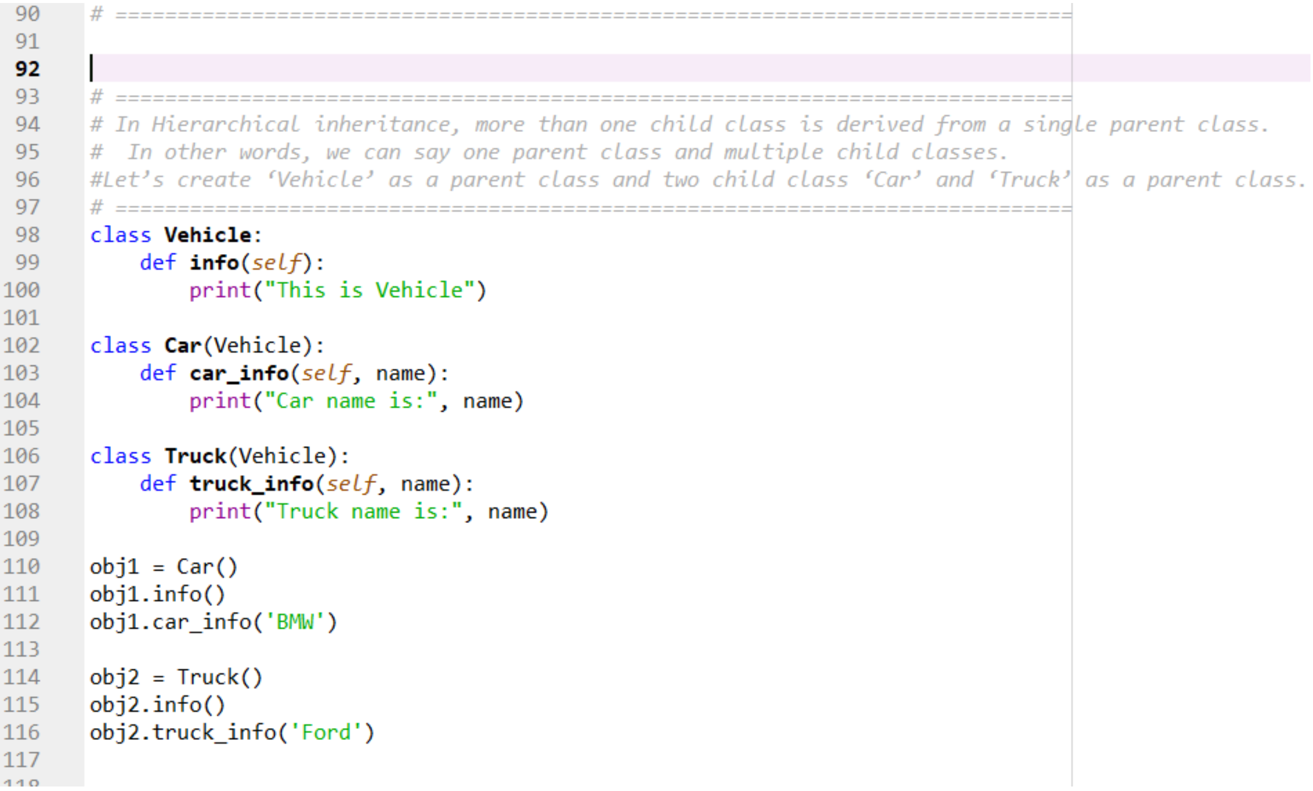
```python
# ==============================================================================
#Python'da, ilgili alt ve üst sınıfların sayısına bağlı olarak, beş tür kalıtım vardır.
#Single inheritance
# Multiple Inheritance
# Multilevel inheritance
# Hierarchical Inheritance
# Hybrid Inheritance
# ==============================================================================
# In single inheritance, a child class inherits from a single-parent class.
# Base class
class Vehicle:
    def Vehicle_info(self):
        print('Inside Vehicle class')

# Child class
class Car(Vehicle):
    def car_info(self):
        print('Inside Car class')

# Create object of Car
car = Car()

# access Vehicle's info using car object
car.Vehicle_info()
car.car_info()

```



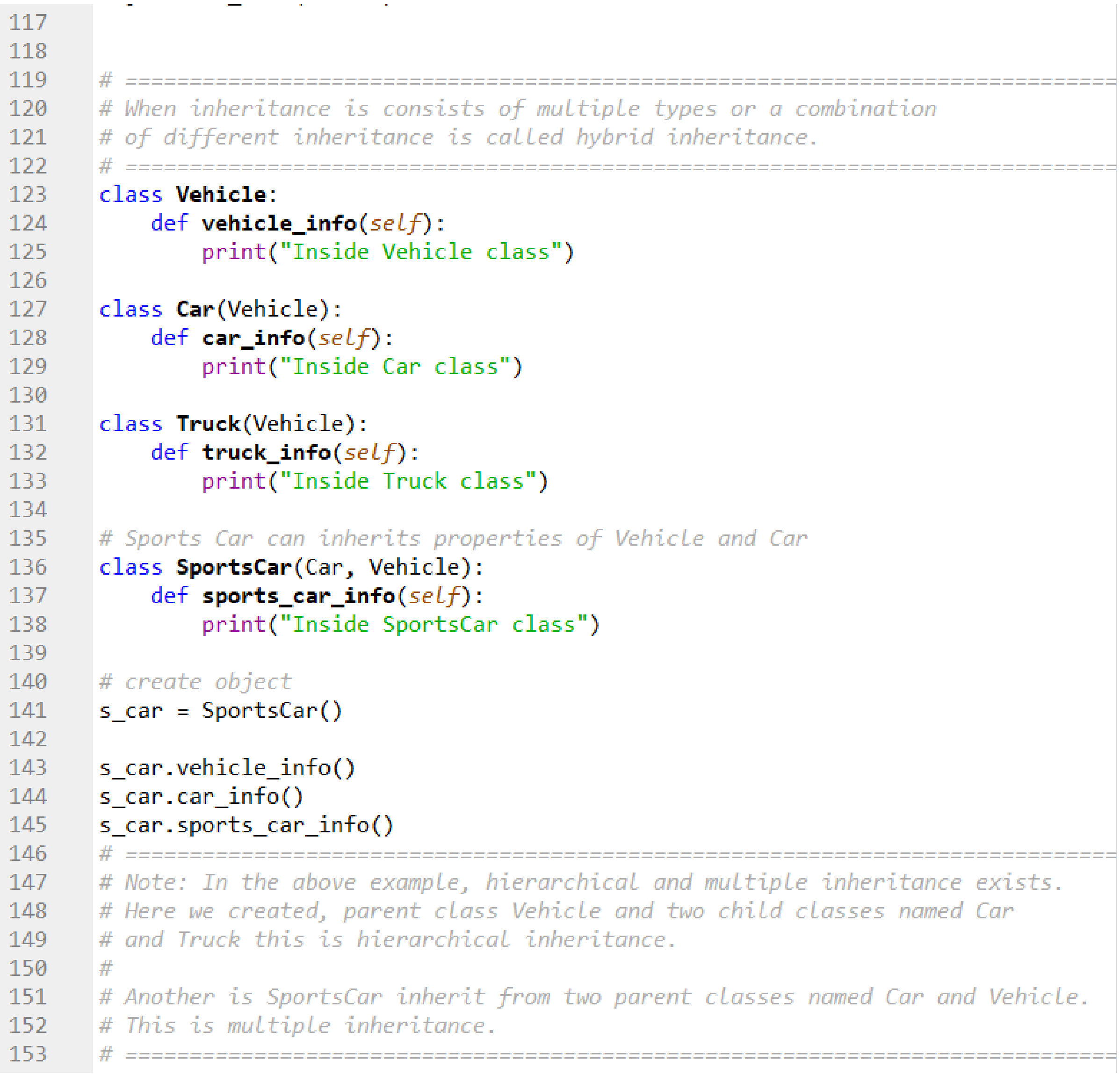
```python

# =============================================================================
# In multiple inheritance, one child class can inherit from multiple parent classes.
# So here is one child class and multiple parent classes.
# =============================================================================
# Parent class 1
class Person:
    def person_info(self, name, age):
        print('Inside Person class')
        print('Name:', name, 'Age:', age)

# Parent class 2
class Company:
    def company_info(self, company_name, location):
        print('Inside Company class')
        print('Name:', company_name, 'location:', location)

# Child class
class Employee(Person, Company):
    def Employee_info(self, salary, skill):
        print('Inside Employee class')
        print('Salary:', salary, 'Skill:', skill)

# Create object of Employee
emp = Employee()

# access data
emp.person_info('Jessa', 28)
emp.company_info('Google', 'Atlanta')
emp.Employee_info(12000, 'Machine Learning')
```



```python


# =============================================================================
# In multilevel inheritance, a class inherits from a child class or derived class.
#  Suppose three classes A, B, C. A is the superclass,
#  B is the child class of A, C is the child class of B.
#  In other words, we can say a chain of classes is called multilevel inheritance.
# =============================================================================
# Base class
class Vehicle:
    def Vehicle_info(self):
        print('Inside Vehicle class')

# Child class
class Car(Vehicle):
    def car_info(self):
        print('Inside Car class')

# Child class
class SportsCar(Car):
    def sports_car_info(self):
        print('Inside SportsCar class')

# Create object of SportsCar
s_car = SportsCar()

# access Vehicle's and Car info using SportsCar object
s_car.Vehicle_info()
s_car.car_info()
s_car.sports_car_info()
# =============================================================================
# In the above example, we can see there are three classes named Vehicle, Car, SportsCar.
# Vehicle is the superclass, Car is a child of Vehicle, SportsCar is a child of Car.
# So we can see the chaining of classes.
# =============================================================================
```



```python
# ==============================================================================


# ==============================================================================
# In Hierarchical inheritance, more than one child class is derived from a single parent class.
#  In other words, we can say one parent class and multiple child classes.
#Let’s create ‘Vehicle’ as a parent class and two child class ‘Car’ and ‘Truck’ as a parent class.
# ==============================================================================
class Vehicle:
    def info(self):
        print("This is Vehicle")

class Car(Vehicle):
    def car_info(self, name):
        print("Car name is:", name)

class Truck(Vehicle):
    def truck_info(self, name):
        print("Truck name is:", name)

obj1 = Car()
obj1.info()
obj1.car_info('BMW')

obj2 = Truck()
obj2.info()
obj2.truck_info('Ford')

```



```python


# ==============================================================================
# When inheritance is consists of multiple types or a combination
# of different inheritance is called hybrid inheritance.
# ==============================================================================
class Vehicle:
    def vehicle_info(self):
        print("Inside Vehicle class")

class Car(Vehicle):
    def car_info(self):
        print("Inside Car class")

class Truck(Vehicle):
    def truck_info(self):
        print("Inside Truck class")

# Sports Car can inherits properties of Vehicle and Car
class SportsCar(Car, Vehicle):
    def sports_car_info(self):
        print("Inside SportsCar class")

# create object
s_car = SportsCar()

s_car.vehicle_info()
s_car.car_info()
s_car.sports_car_info()
# ==============================================================================
# Note: In the above example, hierarchical and multiple inheritance exists.
# Here we created, parent class Vehicle and two child classes named Car
# and Truck this is hierarchical inheritance.
#
# Another is SportsCar inherit from two parent classes named Car and Vehicle.
# This is multiple inheritance.
# ==============================================================================
```