 Class and Object Creation Create a class named Vehicle.

The class should have the following properties: make, model, and year. Implement a constructor to initialize these properties.

Add a method named display_info that prints the make, model, and year of the vehicle.

Create an object of the Vehicle class with specific values and call display_info to show the output.

```
class Vehicle:
  def __init__(self,make,model,year):
    self.make = make
    self.model = model
    self.year = year

  def display_info(self):
    print("Make:",self.make)
    print("Model:",self.model)
    print("Year:",self.year)


my_vehicle =Vehicle("Toyota","Camry",2020)

my_vehicle.display_info()
```

```
Make: Toyota
Model: Camry
Year: 2020
```

 Inheritance Create a subclass of Vehicle named Car.

Add additional properties specific to a car, such as num_doors and fuel_type.

Override the display_info method to include the additional details (num_doors and fuel_type).

Create an instance of Car, initialize it with all relevant values, and call display_info to display the full details.

```
# Parent class
class Vehicle:
  def __init__(self,make,model,year):
    self.make = make
    self.model = model
    self.year = year

  def display_info(self):
    print("Make:",self.make)
    print("Model:",self.model)
    print("Year:",self.year)


# Subclass
class Car(Vehicle):
  def __init__(self,make,model,year,num_doors,fuel_type):
    super().__init__(make,model,year)

    self.num_doors = num_doors
    self.fuel_type = fuel_type

  def display_info(self):
    super().display_info()
    print("No_of_doors:",self.num_doors)
    print("Fuel_type:",self.fuel_type)

my_car = Car("Honda","Civic",2022,4,"petrol")
my_car.display_info()
```

```
Make: Honda
Model: Civic
Year: 2022
No_of_doors: 4
Fuel_type: petrol
```

 Polymorphism with Method Overriding Create another subclass of Vehicle called Bike.

Add a property named engine_type specific to the Bike class.

Override the display_info method in Bike to include engine_type along with the vehicle’s make, model, and year.

Create an instance of Bike, initialize it with all relevant values, and call display_info to test polymorphism.

```
# Parent class
class Vehicle:
  def __init__(self,make,model,year):
    self.make = make
    self.model = model
    self.year = year

  def display_info(self):
    print("Make:",self.make)
    print("Model:",self.model)
    print("Year:",self.year)


# Bike Subclass

class Bike(Vehicle):
  def __init__(self,make,model,year,engine_type):
    super().__init__(make,model,year)
    self.engine_type = engine_type

  def display_info(self):
    super().display_info()
    print("Engine_type:",self.engine_type)

my_bike = Bike("Yamaha","Mt-07",2021,"Parallel_twin")

my_bike.display_info()
```

```
Make: Yamaha
Model: Mt-07
Year: 2021
Engine_type: Parallel_twin
```

 Polymorphism with a Function Write a function named show_vehicle_info that takes a Vehicle object as an argument.

Use this function to display the details of any Vehicle object (e.g., an object of Vehicle, Car, or Bike) by calling the overridden display_info method.

Demonstrate polymorphism by passing instances of Vehicle, Car, and Bike to show_vehicle_info and observe the different outputs.

```
# Parent class
class Vehicle:
  def __init__(self,make,model,year):
    self.make = make
    self.model = model
    self.year = year

  def display_self(self):
    print("Make:",self.make)
    print("Model:",self.model)
    print("year:",self.year)

class Car(Vehicle):
  def __init__(self,make,model,year,num_doors,fuel_type):
    super().__init__(make,model,year)
    self.num_doors = num_doors
    self.fuel_type = fuel_type

  def display_self(self):
    super().display_self()
    print("No_Of_Doors:",self.num_doors)
    print("Fuel_type:",self.fuel_type)

class Bike(Vehicle):
  def __init__(self,make,model,year,engine_type):
    super().__init__(make,model,year)
    self.engine_type = engine_type

  def display_self(self):
    super().display_self()
    print("No_of_doors",self.engine_type)

vehicle = Vehicle("Toyota","Camry",2020)
car = Car("Honda","Civic",2022,4,"petrol")
bike = Bike("Yamaha","Mt-07",2021,"Parallel_twin")

vehicle.display_self()
car.display_self()
bike.display_self()
```

```
Make: Toyota
Model: Camry
year: 2020
Make: Honda
Model: Civic
year: 2022
No_Of_Doors: 4
Fuel_type: petrol
Make: Yamaha
Model: Mt-07
year: 2021
No_of_doors Parallel_twin
```