OBJECT ORIENTED PROGRAMMING WITH PYTHON

Second Class

1st Semester

Object and Class

Create Car Class

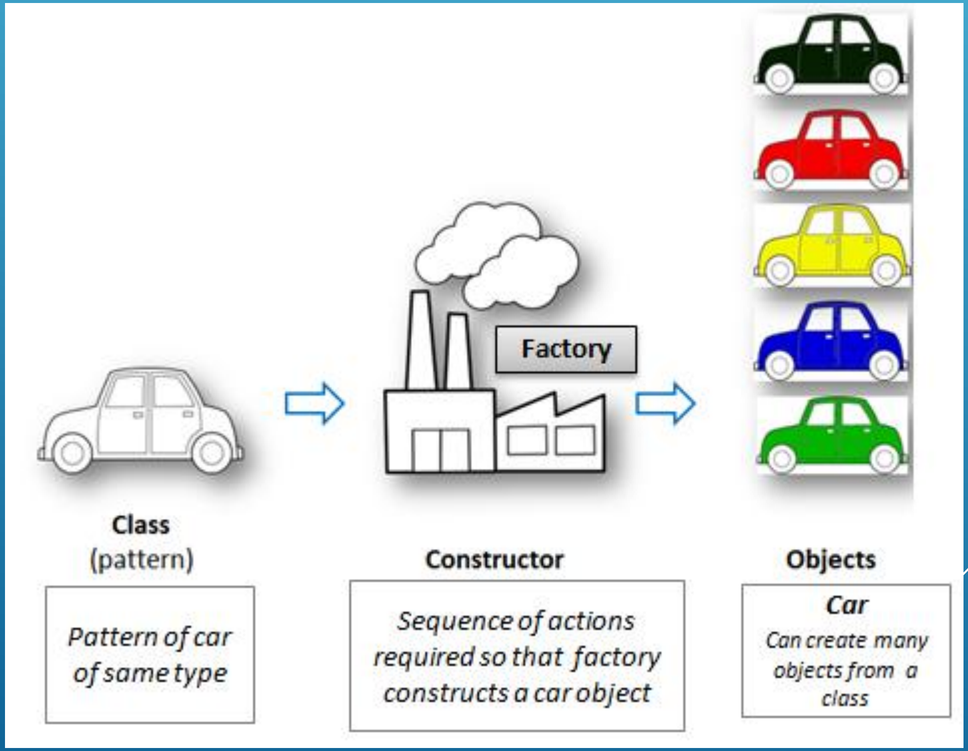
Create Person Class

Inheritance

OBJECT AND CLASS

In Python, everything is an object. Strings, Integers, Float, lists, dictionaries, functions, modules etc are all objects.

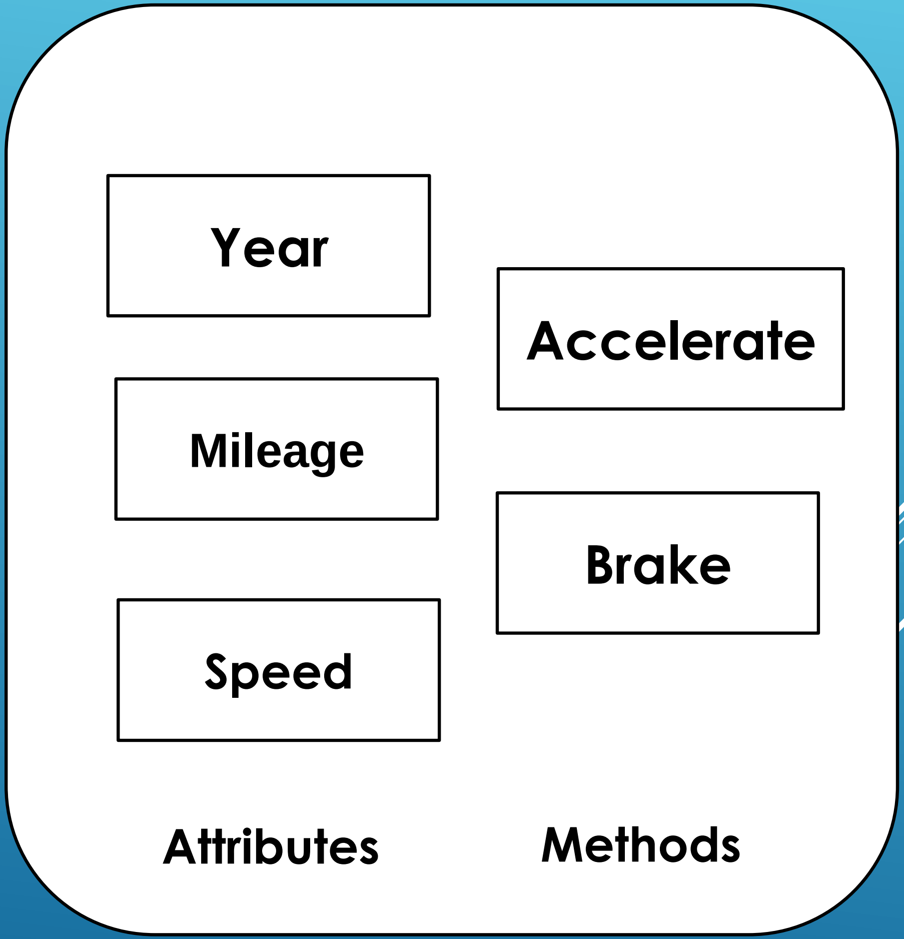
- **Object and Class**
- Class is an architecture of the object. It is a proper description of the attributes and methods of a class. For example, design of a car of same type is a class. You can create many objects from a class. Like you can make many cars of the same type from a design of car.


Factory
Class
(pattern)
Pattern of car of same type
Constructor
Sequence of actions required so that factory constructs a car object
Objects
Car
Can create many objects from a class

CREATE CAR CLASS

Example 1 : Create Car Class
class : car
attributes : year, mileage and speed
methods : accelerate and brake

object : car1



CREATE CAR CLASS

```
class car:
    # attributes
    year = 2016          # car model's year
    mileage = 20         # mileage
    speed = 100          # current speed

    # methods
   def  accelerate(self):
       return car.speed + 20
   def  brake(self):
       return car.speed – 50
```

```
car1=car()

car1.year
2016
car1. mileage
20
car1.speed
100
car1.accelerate()
120
car1.brake()
50
```

Two empty lines then create the object

CREATE PERSON CLASS

Create a class named **Person** with two **methods** **talk** and **leave**

```
class Person:
       def talk(self):
              print("hi everybody!")

      def leave(self):
            print("nice to meet you .Good bye!")
```

Now using the **Person** class we can create an object

```
P1 = Person()
P1.talk()
P1.leave()
```

CREATE PERSON CLASS (CONSTRUCTORS)

The Previous Object needs **attributes**
Attributes are the variables that belong to a particular objects, for example here the person object needs attributes like name and age. To do that we need to use constructor
A **Constructor** is a function that gets called at the time of creating an object.
All classes have a function called

__init__ ()

which is always executed when the class is being initiated.

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

CREATE PERSON CLASS (CONSTRUCTORS)

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

    def talk(self):
           print(f"hi everybody! this is {self.name} I am {self.age} years old ")
    def leave(self):
          print(f"nice to meet you .Good bye!--{self.name}--Left ")



P1 = Person("Ali" , 20)
P2 = Person ("Ahmad", 21)
P1.talk()
P2.talk()
print("***************************************")
P1.leave()
P2.leave()
```

INHERITANCE

Inheritance allows us to define a class that inherits all the methods and properties from another class.

- **Parent class** is the class being inherited from, also called base class.
- **Child class** is the class that inherits from another class, also called derived class.

Let us create a child class named **Students** that Inherits a **Person** class:

INHERITANCE

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

    def talk(self):
        print(f"hi everybody! this is {self.name} I am {self.age} years old ")
    def leave(self):
        print(f"nice to meet you .Good bye!--{self.name}--Left ")


class Student(Person):
    pass


P1 = Person("Ali" , 20)
P2 = Person ("Ahmad", 21)
S1 = Student("Sara",15)
S1.talk()
P1.talk()
P2.talk()
print("***************************************")
P1.leave()
P2.leave()
S1.leave()
```