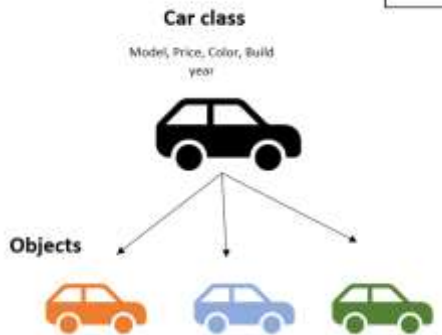
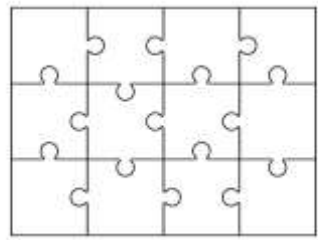
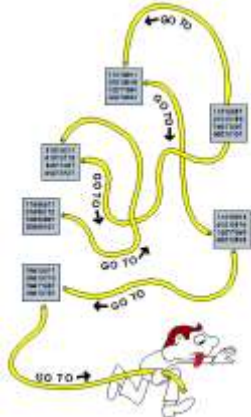


What programming paradigms do these pictures remind you of?

- Procedural Programming
- Structured Programming
- Object Oriented Programming



Object-Oriented Programming. Classes and Objects.

11.4.1.1 create classes and instances of classes

11.4.1.2 develop methods for the class

11.4.1.3 use special method `__init__` to set default properties

What is Object-Oriented Programming?



[What is Object-Oriented Programming?](#)

[| Coding for Kids | Kodable](#)

Object-oriented programming (OOP)

- OOP is a computer programming paradigm that organizes software design around data, or rather than and logic.
- This approach to programming is well-suited for programs that are large, c and actively updated or maintained.
- this method beneficial to collaborative development, where projects are divided into groups.
- Additional benefits of OOP include code reusability, scalability and efficiency.



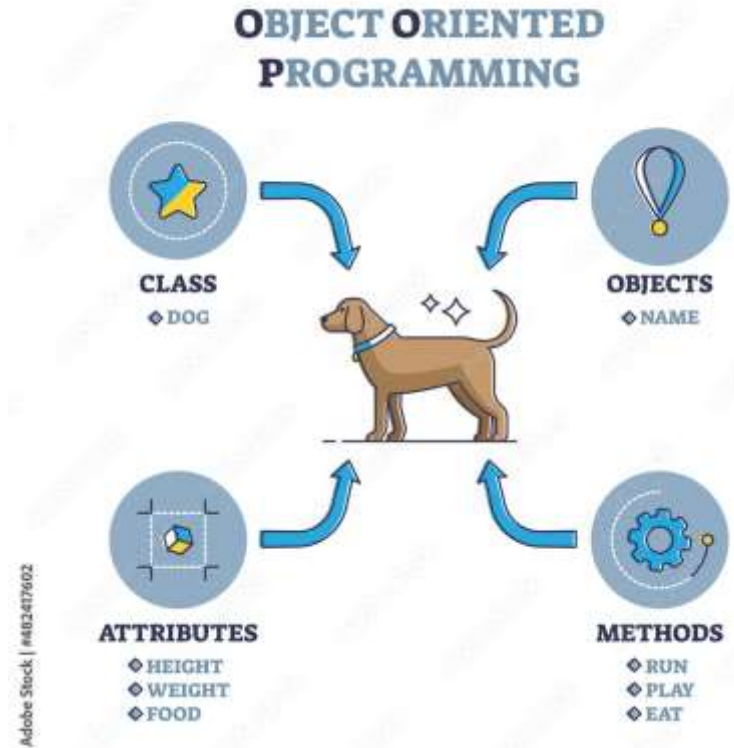
What are Classes and Objects?

Class: The class is a user-defined data that binds the methods into a single unit.

Class is a blueprint or code for object creation. Using a class, you can create as many as you want.

Object: An object is an of a class. It is a collection of attributes (variables) and methods. We use the object of a class to perform actions.

A **method** is an at an object can perform an instance of.



Give your own example of a class and an object

Classes in Python

- Python is an object oriented programming language.
- Almost everything in Python is an object, with its properties and methods.
- A Class is like an object constructor, or a "blueprint" for creating objects.
- All the data types we have studied are classes.
- All variables are objects.

```
a = 5
print(type(a)) # <class 'int'>
x = {1:"one", 2: "two", 3: "three"}
print(type(x)) # <class 'dict'>
```

5 is an instance of class 'int'
{1:"one", 2: "two", 3: "three"} is an instance of class 'dict'

Creating classes

- To create a class, use the keyword **class**:
- The class name starts with Capital Letter
- Object and class attributes are accessed using dot notation in Python.

```
class Person:#creating class  
    x=10 #class variable
```

```
p1=Person()#creating object
```

```
print(Person)  
print(p1.x)  
|
```

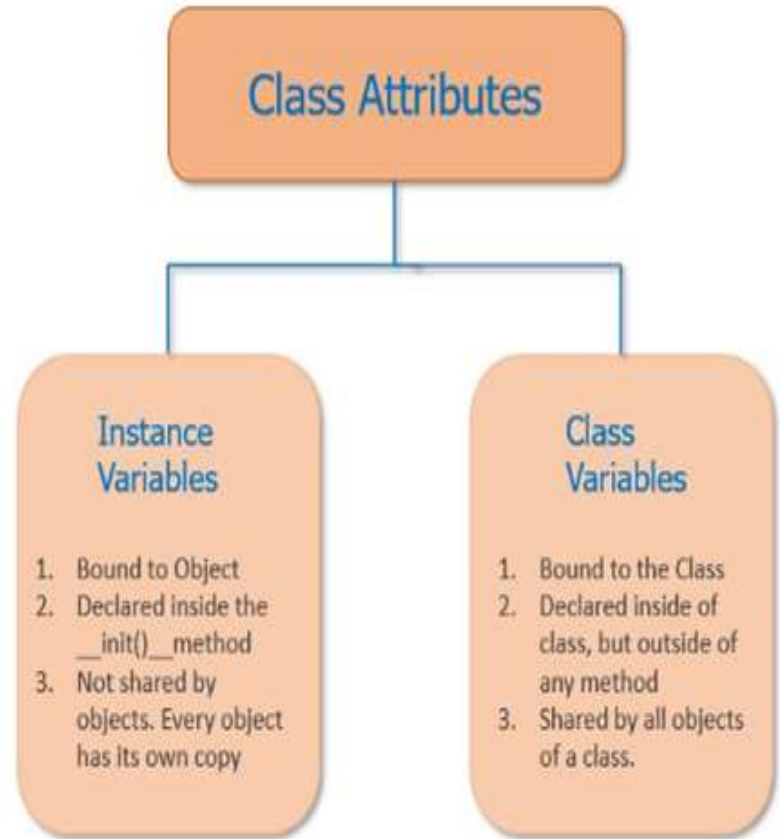
```
<class '__main__.Person'>  
10
```

Class Attributes

When we design a class, we use instance variables and class variables

Instance variables: The instance variables are attributes attached to an instance of a class. We define instance variables in the constructor (the `__init__()` method of a class).

Class Variables: A class variable is a variable that is declared inside of class, but outside of any instance method or `__init__()` method.



Class Methods

- All functions within a class are called **methods**.
- The method must be defined **inside the class** (indentation level added)
- Methods always have **at least one argument**, and the first argument must be called **self**. The argument self is passed to the object that called this method. This is why self is often referred to as a context object.

```
class Person:  
    def hello(self): # hello is a method, self is an object  
        print("Hello, World!")
```

```
p1 = Person()  
p1.hello() # output "Hello, world!"
```

What the program will output?

```
class Greeter:
    def hello(self):
        print("Hello, World!")
    def greeting(self, name):
        print(f"Hello, {name}!")
    def start_talking(self, name, weather_is_good):
        print(f"Hello, {name}!")
        if weather_is_good:
            print("Good weather, isn't it?")
        else:
            print("Disgusting weather, isn't it?")
```

```
greet = Greeter()
greet.hello()
greet.greeting("Halil")
greet.start_talking("Ruslan", False)
greet.start_talking("Ruslan")
```

```
Hello, World!
Hello, Halil!
Hello, Ruslan!
Disgusting weather, isn't it?
```

```
Traceback (most recent call last):
  File "./prog.py", line 17, in <module>
TypeError: start_talking() missing 1 required
```

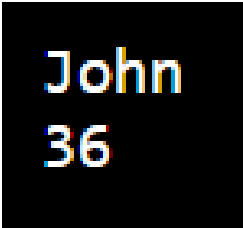
The `__init__()` method

- All classes have a function called `__init__()` which is always executed when the class is being initiated.
- Use the `__init__()` function to assign values to object properties

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

```
p1 = Person("John", 36)
```

```
print(p1.name)  
print(p1.age)
```

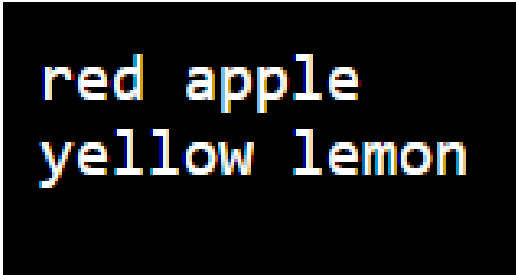


John
36

What the program will output?

```
class Fruit:
    def __init__(self):          # default values
        self.color = 'green'   # each object gets default color 'green'
        self.fruit = 'apple'   # each object gets default fruit 'apple'
    def color(self, color):     # set new color for object of class Fruit
        self.color = color
    def fruit(self, fruit):     # set new fruit for object of class Fruit
        self.fruit = fruit
    def info(self):
        return f'{self.color} {self.fruit}'
```

```
f1 = Fruit()
f1.color = 'red'
print(f1.info())
|
f2= Fruit()
f2.color = 'yellow'
f2.fruit = 'lemon'
print(f2.info())
```



red apple
yellow lemon

Individual work

[Exercise v3.0 \(w3schools.com\)](https://www.w3schools.com/python/python_classes_exercises_v3.php)

Completed 4 of 95 Exercises:

- PYTHON For Loops
- PYTHON Functions
- PYTHON Lambda
- PYTHON Classes ✓
- ✓ Exercise 1
- ✓ Exercise 2
- ✓ Exercise 3
- ✓ Exercise 4

[Go to PYTHON Classes Tutorial](#)

PYTHON Inheritance

Exercise:

Create an object of MyClass called p1:

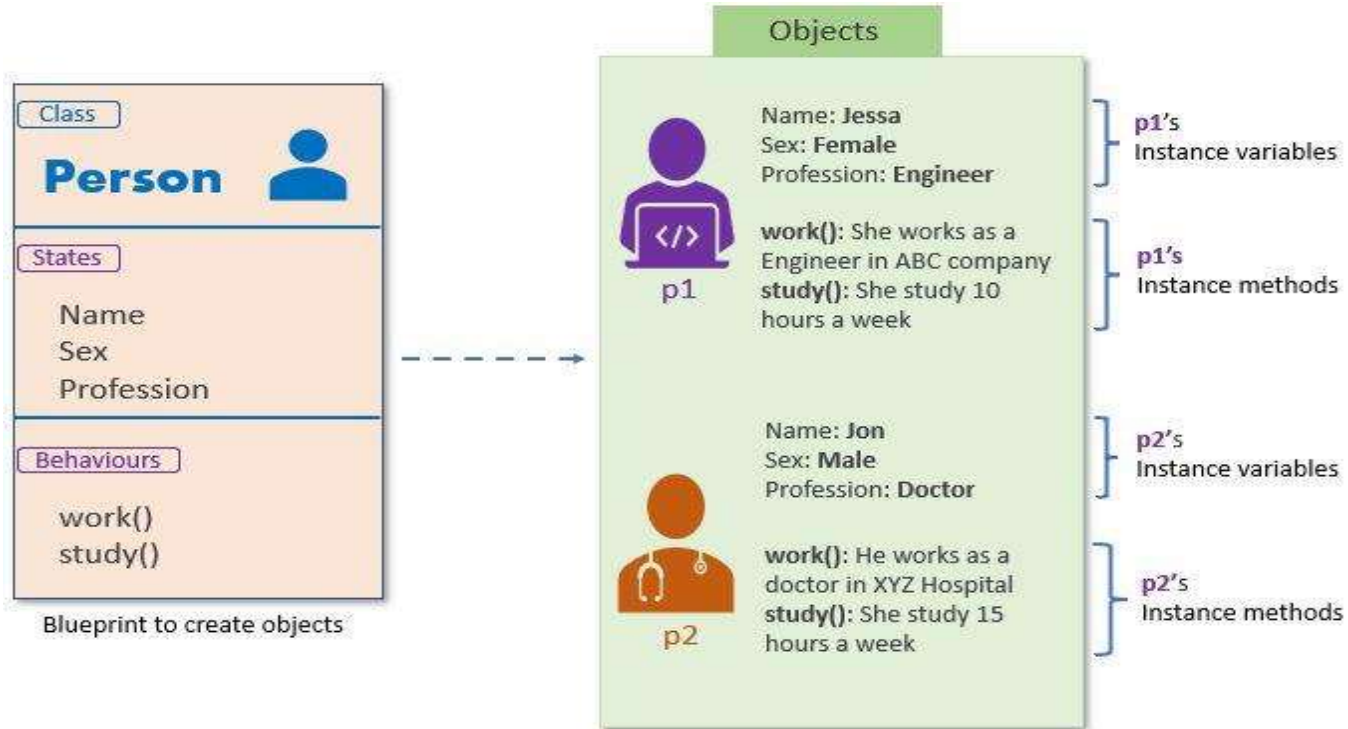
Correct!

[Next >](#)

[Next Exercise >](#)

Pair work.

Create the class Person and objects p1, p2, p3



Individual work

The screenshot displays the Stepik website interface. At the top left is the Stepik logo. The main header area shows the course title "Python Programming for NIS" and the progress "Прогресс по курсу: 5/394". A sidebar menu on the left lists course sections: "5.10 Nested lists", "5.11 Dictionaries", "6 Functions", "6.1 Functions. Arguments", "6.2 Functions. Return", "6.3 Lambda functions", "7 Object-Oriented Program...", "7.1 Python Classes and O...", and "7.2 Polymorphism and Inh...". The "7.1 Python Classes and O..." item is highlighted in green. On the right side, there is a "Help: Pyt" link, "Sample I" with a "2" below it, "Sample C" with a "2" below it, and a "Напиш" section with a green checkmark icon and the text "Пре". At the bottom right, a code editor is visible with a light green background and a list of numbers 1 through 5.

Group 1

- 1. Create a Plate class, then create an instance of it called saucer with the attributes color="white" and size=100. Get output "White saucer, size - 100"**
- 2. Create a new AdvancePlate class that initializes the color and size attributes with the __init__() method. Also, the class must have a output() method that prints information about the plate in the form: "I am <color> a plate with a radius of <size> mm." Get output "I am violet a plate with a radius of 120 mm."**

Group 2

1. **Create a class Car, then create an instance of it named midget_car with brand and length attributes with values "KIA" and 2600 respectively. Get output "KIA is a model of car, length - 2600"**
2. **Create a new AdvanceCar class that initializes the brand and length attributes with the __init__() method. Also, the class should have an info() method that prints information about the car in the form: "I got a taxi <brand> with a length of <length> mm." Get output "I got a taxi BMW with a length of 2800 mm."**

Group 3

- Write a program that asks for the user's number.
- If the number belongs to the range from -100 to 100, then the created instance belongs to the first class, otherwise, the created instance belongs to the second class.
- The class includes the `__init__` method, which in the first-class calculates the square of a number, and in the second class, it multiplies by two.

Additional task

- Create your own class
 - with 3 attributes
 - and 2 methods
- Create 2 objects of your class

Material to read!

1. <https://pynative.com/python-classes-and-objects/>
2. https://www.bzfar.org/load/informatika/computer_science/programmirovanie_na_python_11_klass/11-1-0-48
3. https://www.w3schools.com/python/python_classes.asp