Classes

- Class: blueprint for creating new objects
- Object: instance of a class

For example, we have a class called *human*, and this class will define all the attributes of humans. Then we could create objects, like john, mary, jack and so on.

In [1]:
```
x = 1
print(type(x))
```

```
<class 'int'>
```

How to create custom classes, like customer, shopping cart and point, etc.

Creating Classes

Convention: To name variables and functions, we use all lower case letters, and use an underscore to separate multiple words. However, when naming a class, the first letter of every word should be upper case, and we shouldn't use an underscore to separate multiple words.

Examples:

- Variable / function: my_point
- Class: MyPoint

In [2]:
```
class Point:
    def draw(self):
        print("draw")
```

In [3]:
```
point = Point()
point.draw()
print(type(point))
```

```
draw
<class '__main__.Point'>
```

The above "__main__" is the name of a module, which we will show later in this course.

In [4]:
```
isinstance(point, Point), isinstance(point, int)
```

Out[4]:
```
(True, False)
```

Constructors

A class bundles data and functions related to that data into one unit.

A constructor is a special method that is called when we create a new object.

In [5]:
```
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def draw(self):
        print(f"Point ({self.x}, {self.y})")

point = Point(1, 2)
point.draw()
print(point.x)
```

```
Point (1, 2)
1
```

This magic method "__init__" is called a constructor. "self" is the reference to the current point object.

Class vs Instance Attributes

In [6]:

```
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def draw(self):
        print(f"Point ({self.x}, {self.y})")

point = Point(1, 2)
point.z = 10
point.draw()

another = Point(3, 4)
another.draw()
```

```
Point (1, 2)
Point (3, 4)
```

In the above, we can add an attribute (point.z) whenever we need. We defined two instances with different attributes x and y.

In some cases, all instances of a class share some common attributes, like all humans have one head.

In [7]:

```
class Point:
    default_color = "red"

    def __init__(self, x, y):
        self.x = x
        self.y = y

    def draw(self):
        print(f"Point ({self.x}, {self.y})")

point = Point(1, 2)
print(point.default_color)
print(Point.default_color)
point.draw()

another = Point(3, 4)
another.draw()
```

```
red
red
```

```
Point (1, 2)
Point (3, 4)
```

In [8]:
```
Point.default_color = "yellow"

point = Point(1, 2)
print(point.default_color)
print(Point.default_color)
point.draw()

another = Point(3, 4)
print(another.default_color)
another.draw()
```

```
yellow
yellow
Point (1, 2)
yellow
Point (3, 4)
```

Class-level attributes are shared across all instances of that class.