Programación orientada a objetos

Introducción a la Programación Orientada a Objetos

La Programación Orientado a Objetos (POO) se basa en la agrupación de objetos de distintas clases que interactúan entre sí y que, en conjunto, consiguen que un programa cumpla su propósito. En Python cualquier elemento del lenguaje pertenece a una clase y todas las clases tienen el mismo rango y se utilizan del mismo modo.

Definición de clase, objeto, atributos y métodos

- Llamamos clase a la representación abstracta de un concepto. Por ejemplo, "perro", "número entero" o "servidor web".
- Las clases se componen de atributos y métodos.
- Un objeto es cada una de las instancias de una clase.
- Los atributos definen las características propias del objeto y modifican su estado. Son datos asociados a las clases y a los objetos creados a partir de ellas.
- Un atributo de clase es compartida por todas las instancias de una clase. Se definen dentro de la clase (después del encabezado de la clase) pero nunca dentro de un método. Este tipo de variables no se utilizan con tanta frecuencia como las variables de instancia.
- Un atributo de objeto se define dentro de un método y pertenece a un objeto determinado de la clase instanciada.
- Los métodos son bloques de código (o funciones) de una clase que se utilizan para definir el comportamiento de los objetos.

Definimos nuestra primera clase:

```
>>> class clase():
...    at_clase=1
...    def metodo(self):
...      self.at_objeto=1
...
>>> type(clase)
<class 'type'>
>>> clase.at_clase
1
>>> objeto=clase()
>>> objeto.at_clase
1
>>> objeto.at_objeto
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'clase' object has no attribute 'at_objeto'
>>> objeto.metodo()
>>> objeto.at_objeto
1
```

Atributos de objetos

Para definir atributos de objetos, basta con definir una variable dentro de los métodos, es una buena idea definir todos los atributos de nuestras instancias en el constructor, de modo que se creen con algún valor válido.

Método constructor init

Como hemos visto anteriormente los atributos de objetos no se crean hasta que no hemos ejecutado el método. Tenemos un método especial, llamado constructor `__init__`, que nos permite inicializar los atributos de objetos. Este método se llama cada vez que se crea una nueva instancia de la clase.

Definiendo métodos. El parámetro self

El método construcutor, al igual que todos los métodos de cualquier clase, recibe como primer parámetro a la instancia sobre la que está trabajando. Por convención a ese primer parámetro se lo suele llamar `self` (que podríamos traducir como yo mismo), pero puede llamarse de cualquier forma.

Para referirse a los atributos de objetos hay que hacerlo a partir del objeto `self`.

Definición de objetos

Vamos a crear una nueva clase:

```
 import math
class punto():
""" Representación de un punto en el plano, los atributos son x e y
que representan los valores de las coordenadas cartesianas."""

def __init__(self,x=0,y=0):
    self.x=x
    self.y=y

def distancia(self, otro):
    """ Devuelve la distancia entre ambos puntos. """
    dx = self.x - otro.x
    dy = self.y - otro.y
    return math.sqrt((dx*dx + dy*dy))
```

Para crear un objeto, utilizamos el nombre de la clase enviando como parámetro los valores que va a recibir el constructor.

```
 >>> punto1=punto()
>>> punto2=punto(4,5)
>>> print(punto1.distancia(punto2))
6.4031242374328485
```

Podemos acceder y modificar los atributos de objeto:

```
 >>> punto2.x
4
>>> punto2.x = 7
>>> punto2.x
7
```