¿Qué es una Clase?

Una clase (entre otras definiciones) es un conjunto de objetos. Un objeto es un ser perteneciente a una clase.

Un objeto es una encarnación de los requisitos, rasgos y cualidades asignados a una clase específica.

¿Qué contiene un objeto?

La programación orientada a objetos supone que cada objeto existente puede estar equipado con tres grupos de atributos:

- ✓ Un objeto tiene un nombre que lo identifica de forma exclusiva dentro de su namespace (aunque también puede haber algunos objetos anónimos).
- ✓ Un objeto tiene un conjunto de propiedades individuales que lo hacen original, único o sobresaliente (aunque es posible que algunos objetos no tengan propiedades).
- ✓ Un objeto tiene un conjunto de habilidades para realizar actividades específicas, capaz de cambiar el objeto en sí, o algunos de los otros objetos.

Hay una pista (aunque esto no siempre funciona) que te puede ayudar a identificar cualquiera de las tres esferas anteriores. Cada vez que se describe un objeto y se usa:

- ✓ Un sustantivo: probablemente se este definiendo el nombre del objeto.
- ✓ Un adjetivo: probablemente se este definiendo una propiedad del objeto.
- ✓ Un verbo: probablemente se este definiendo una actividad del objeto.

Dos ejemplos deberían servir como un buen ejemplo:

- Max es un gato grande que duerme todo el día.

Nombre del objeto = Max

Clase de inicio = Gato

Propiedad = Tamaño (grande)

Actividad = Dormir (todo el día)

- Un Cadillac rosa pasó rápidamente.

Nombre del objeto = Cadillac

Clase de inicio = Vehículo terrestre

Propiedad = Color (rosa)

Actividad = Pasar (rápidamente)

Definiendo Clases

La definición comienza con la palabra clave reservada **class**. La palabra clave reservada es seguida por un identificador que nombrará la clase

A continuación, se agregan dos puntos:), como clases, como funciones, forman su propio bloque anidado. El contenido dentro del bloque define todas las propiedades y actividades de la clase.

La palabra clave reservada pass llena la clase con nada. No contiene ningún método ni propiedades.

Ejemplo:

```
class ClaseSimple:
  pass
```

Para crear un objeto de esta clase Colocaremos:

```
miPrimerObjeto=ClaseSimple()
```

Creemos la clase Rectángulo

```
Class Prueba:

    def __init__(self):

        print("¡Hola!")


objeto = Prueba()
```

Resultado:

Hola.

Explicación:

- ✓ El nombre del constructor es siempre __init__.
- ✓ Tiene que tener al menos un parámetro; el parámetro se usa para representar el objeto recién creado: puedes usar el parámetro para manipular el objeto y enriquecerlo con las propiedades necesarias.
- ✓ El parámetro obligatorio generalmente se denomina self.

```
class Circulo:
   def __init__(self):
      self.radio=1
circulo=Circulo()
print(circulo.radio)
```

Este es una clase con un atributo radio con valor=1. Este atributo es publico.

Si queremos que un atributo sea privado se lo declara con un nombre que comienza con dos guiones bajos (__). Esto significa que solo puede accederse de la clase.

```python
from math import pi
class Circulo:
    def __init__(self):
        self.__radio=1

    def modificarRadio(self,valor):
        self.__radio=valor

    def devolverRadio(self):
        return self.__radio

    def calcularArea(self):
        return pi*pow(self.__radio,2)

    def calcularLongitud(self):
        return pi*pow(self.__radio,2)

objeto=Circulo()
objeto.modificarRadio(5)
print('Circulo')
print('Radio = ',objeto.devolverRadio())
print('El area es: ',objeto.calcularArea())
print('La longitud es: ',objeto.calcularLongitud())
```

Todas las funciones tienen un parámetro llamado self en la primera posición de la lista de parámetros.

¿Es necesario? Si, lo es.

Todos los métodos deben tener este parámetro. Permite que el método acceda a entidades (propiedades y actividades / métodos) del objeto. No puedes omitirlo. Cada vez que Python invoca un método, envía implícitamente el objeto actual como el primer argumento.

Herencia

```
from math import pi
class Circulo:
    def __init__(self):
        self.__radio=1

    def modificarRadio(self,valor):
        self.__radio=valor

    def devolverRadio(self):
        return self.__radio

    def calcularArea(self):
        return pi*pow(self.__radio,2)

    def calcularLongitud(self):
        return 2*pi*self.__radio

class Cilindro(Circulo):
    def __init__(self):
        Circulo.__init__(self)
        self.__altura=1

    def modificarAltura(self,valor):
        self.__altura=valor

    def devolverAltura(self):
        return self.__altura

    def calcularArea(self):
        return 2*Circulo.calcularArea(self)+Circulo.calcularLongitud(self)*s
elf.__altura

    def calcularVolumen(self):
        return Circulo.calcularArea(self)*self.__altura

objetoCirculo=Circulo()
objetoCirculo.modificarRadio(5)
print('Circulo')
print('Radio =',objetoCirculo.devolverRadio())
print('El area es: ',objetoCirculo.calcularArea())
print('Cilindro')
objetoCilindro=Cilindro()
objetoCilindro.modificarRadio(4)
objetoCilindro.modificarAltura(10)
```

```
print('Radio =',objetoCilindro.devolverRadio())
print('Altura =',objetoCilindro.devolverAltura())
print('Area del Cilindro :',objetoCilindro.calcularArea())
print('Volumen del cilindro:',objetoCilindro.calcularVolumen())
```

Variables de Clase

Una variable de clase es una propiedad que existe en una sola copia y se almacena fuera de cualquier objeto.

Nota: no existe una variable de instancia si no hay ningún objeto en la clase; existe una variable de clase en una copia, incluso si no hay objetos en la clase.

Las variables de clase se crean de manera diferente

```
class ClaseEjemplo:
    contador = 0
    def __init__(self, val = 1):
        self.__primera = val
        ClaseEjemplo.contador += 1

objetoEjemplo1 = ClaseEjemplo()
objetoEjemplo2 = ClaseEjemplo(2)
objetoEjemplo3 = ClaseEjemplo(4)

print(objetoEjemplo1.__dict__, objetoEjemplo1.contador)
print(objetoEjemplo2.__dict__, objetoEjemplo2.contador)
```

Observa:

- ✓ Hay una asignación en la primera linea de la definición de clase: establece la variable denominada contador a 0; inicializando la variable dentro de la clase pero fuera de cualquiera de sus métodos hace que la variable sea una variable de clase.
- ✓ El acceder a dicha variable tiene el mismo aspecto que acceder a cualquier atributo de instancia; está en el cuerpo del constructor; como puedes ver, el constructor incrementa la variable en uno; en efecto, la variable cuenta todos los objetos creados

Ejercicios

1. Crear la clase TrianguloRectangulo con atributos cateto1, cateto2 y los métodos para calcular el área, la hipotenusa y el perímetro.
2. Crear la clase ConversionTemperatura que tenga como atributo a temperatura en grados centígrados y tenga un método para convertir grados centígrados en grados farenheit.
3. Crear la clase Producto con atributos nombre, precioDeCosto y precioDeVenta. Debe tener un método para calcular la ganancia.
4. Crear la clase Trabajador con atributos nombre, precioHora y horasTrabajadas. Debe tener los métodos para calcular el salario bruto, calcular el impuesto que son el 10 por ciento del salario bruto y calcular el salario neto.
5. Crear la clase Movil con atributos velocidadInicial, tiempo y acelearacion. Debe tener un método que permita calcular el espacio recorrido por un móvil.
6. Crear una clase llamada Rectangulo que contenga como atributos base y altura, crear un método para calcular el área del Rectángulo . De esta clase derivar una clase denominada Caja que tenga un atributo adicional denominado profundidad y un método que permita calcular el volumen.
7. Crear una clase ObjetoGeometrico con atributos x, y que son el centro del objeto geométrico. Crear dos clases Circulo y Cuadrado que derivan de ObjetoGeometrico y que permitan calcular sus áreas. Una vez creadas las clases. Escribir un programa que cree un objeto de cada clase y visualice los centros de cada figura con sus respectivas áreas.