UNIVERSIDAD AUTONOMA DE BAJA CALIFORNIA

FACULTAD DE CONTADURIA Y ADMINSITRACION

PROGRAMACION ORIENTADA A OBJETOS EN PYTHON

SERRANO ROMERO DIANA BERENICE
GRUPO 372
PROGRAMACION EN PYTHON

TAREA 5: PROGRAMACION ORIENTADA A OBJETOS EN PYTHON

La Programación Orientada a Objetos (POO u OOP por sus siglas en inglés), es un paradigma de programación. Los elementos de la POO, pueden entenderse como los materiales que necesitamos para diseñar y programar un sistema, mientras que las características, podrían asumirse como las herramientas de las cuáles disponemos para construir el sistema con esos materiales.

CLASES

Las clases son los modelos sobre los cuáles se construirán nuestros objetos. Podemos tomar como ejemplo de clases, el gráfico que hicimos en la página 8 de este documento.

En Python, una clase se define con la instrucción class seguida de un nombre genérico para el objeto.

EJEMPLO DE CLASES

```
class Objeto:
    Pass
```

METODOS

Los métodos son funciones, solo que técnicamente se denominan métodos, y representan acciones propias que puede realizar el objeto (y no otro).

EJEMPLO DE LOS METODOS

```
class Objeto():
    color = "verde"
    tamanio = "grande"
    aspecto = "feo"
    antenas = Antena()
    ojos = Ojo()
    pelos = Pelo()

    def flotar(self):
        pass
```

OBJETOS

Las clases por sí mismas, no son más que modelos que nos servirán para crear objetos en concreto. Podemos decir que una clase, es el razonamiento abstracto de un objeto, mientras que el objeto, es su materialización. A la acción de crear objetos, se la denomina instanciar una clase y dicha instancia, consiste en asignar la clase, como valor a una variable.

EJEMPLO DE UN OBJETO

```
class Objeto():

    color = "verde"

    tamanio = "grande"

    aspecto = "feo"

    antenas = Antena()

    ojos = Ojo()

    pelos = Pelo()


    def flotar(self):

        print 12


et = Objeto()

print et.color

print et.tamanio

print et.aspecto

et.color = "rosa"

print et.color
```

HERENCIA

Como comentamos en el título anterior, algunos objetos comparten las mismas propiedades y métodos que otro objeto, y además agregan nuevas propiedades y métodos. A esto se lo denomina herencia: una clase que hereda de otra. Vale aclarar, que en Python, cuando una clase no hereda de ninguna otra, debe hacerse heredar de object, que es la clase principal de Python, que define un objeto.

EJEMPLO DE LA HERENCIA

```
class Antena(object):
    color = ""
    longitud = ""

class Pelo(object):
    color = ""
    textura = ""

class Ojo(object):
    forma = ""
    color = ""
    tamanio = ""

class Objeto(object):
    color = ""
    tamanio = ""
    aspecto = ""
    antenas = Antena()
    ojos = Ojo()
    pelos = Pelo()

    def flotar(self):
        pass
```

```
class Dedo(object):
    longitud = ""
    forma = ""
    color = ""

class Pie(object):
    forma = ""
    color = ""
    dedos = Dedo()

# NuevoObjeto sí hereda de otra clase: Objeto
class NuevoObjeto(Objeto):
    pie = Pie()

    def saltar(self):
        pass
```

REFERENCIAS

https://librosweb.es/libro/python/capitulo_5/programacion_orientada_a_objetos.html