

Desarrollo de Software en Astrofísica

Sala 763 Pabellón Forma, Profesor Sebastián Pérez
4 de abril 2023

Introducción a PYTHON (para astrónomes)

Programación Orientada al Objeto (POO o OOP)



enfoque de programación que organiza el código en objetos que interactúan entre sí para realizar tareas. Estos objetos se definen por sus propiedades (conocidas como atributos o campos) y los comportamientos que exhiben (conocidos como métodos o funciones).

En la POO, los objetos se crean a partir de clases, que son como planos para crear múltiples instancias del mismo tipo de objeto. Una clase define los atributos y métodos que tendrán todos los objetos de ese tipo, pero cada objeto individual puede tener sus propios valores para los atributos.

En resumen, la POO se basa en la idea de que el código debe organizarse en términos de objetos, y estos objetos deben interactuar entre sí para lograr los objetivos de la aplicación.

Programación Orientada a Objeto - ejemplo.

```
class Persona:
    def __init__(self, nombre, edad):
        self.nombre = nombre
        self.edad = edad

    def saludar(self):
        print("Hola, mi nombre es " + self.nombre + " y tengo " + str(self.edad) + " años.")

# Crear una instancia de la clase Persona
persona1 = Persona("Norman", 25)

# Llamar al método saludar
persona1.saludar()
```

Programación Orientada a Objeto - ejemplo.

La primera línea define la clase Persona y los dos atributos que tiene: nombre y edad.

```
class Persona:  
    def __init__(self, nombre, edad):  
        self.nombre = nombre  
        self.edad = edad
```

`__init__` es el constructor de la clase, y se llama cuando se crea una nueva instancia de la clase.
Recibe los parámetros nombre y edad, que se asignan a los atributos correspondientes de la instancia.

`self` es una referencia a la instancia actual

```
    def saludar(self):  
        print("Hola, mi nombre es " + self.nombre + " y tengo " + str(self.edad) + " años.")
```

El método `saludar` simplemente imprime un saludo con el nombre y edad de la persona.

```
# Crear una instancia de la clase Persona  
persona1 = Persona("Norman", 25)
```

Después de definir la clase, creamos una instancia de la misma con el nombre `persona1`, con nombre "Norman" y edad 25.

```
# Llamar al método saludar
```

```
persona1.saludar()
```

llamamos al método `saludar` de la instancia `persona1`, que imprime "Hola, mi nombre es Norman y tengo 25 años."

La única manera de aprender P00 es practicando DATA CHALLENGE

Escribe un código en Python de **programación orientada a objetos** que lea datos de una **tabla ASCII (bajar del link más abajo)** con información sobre exoplanetas y crea un gráfico de masa vs radio orbital. También se pueden proponer otros gráficos para visualizar otras propiedades de los exoplanetas.

Los datos en la tabla son 5 columnas para: Nombre del planeta, Masa relativa a la Tierra, Radio relativo a la Tierra, Semieje mayor de la orbita (unidad astronómica), Periodo Orbital (días)

Sugerencia: define dos clases, una para representar un objeto “exoplaneta” que tenga los atributos “nombre, masa, radio, orbita, periodo” y una clase para la tabla de datos de exoplanetas.

Usa matplotlib y trata de usar escala log-log (el rango de tamaños de órbita y masas es muy grande!).

https://www.dropbox.com/s/36h1j5nwzp947a9/exoplanets_all.txt?dl=0

```
2M 0746+20 b,9534.85,10.87,2.897,4640.00
2M 2140+16 b,6356.57,10.31,3.530,7340.00
2M 2206-20 b,9534.85,14.57,4.480,8686.00
51 Eri b,635.66,11.21,14.000,15000.00
55 Cancri e,7.98,1.88,0.015,0.74
AU Mic b,57.21,4.20,0.066,8.46
BD+20 594 b,16.30,2.23,0.241,41.69
BD+45 564 b,432.25,13.56,0.830,307.88
BD+55 362 b,228.84,14.01,0.780,265.59
BD+63 1405 b,1258.60,13.00,2.060,1198.48
CFHTWIR-Oph 98 A b,2479.06,20.85,200.000,8040000.00
CoRoT-10 b,874.03,10.87,0.105,13.24
CoRoT-11 b,791.39,15.58,0.044,2.99
CoRoT-12 b,291.45,16.14,0.040,2.83
CoRoT-13 b,415.72,9.92,0.051,4.04
CoRoT-14 b,2415.50,12.22,0.027,1.51
CoRoT-16 b,168.45,13.11,0.062,5.35
CoRoT-17 b,778.68,11.43,0.046,3.77
```

