Introducción al Lenguaje de programación python

Julián Sepúlveda Berrío

19 de mayo de 2021

1. Generalidades
   - La programación como un procedimiento
   - Un Lenguaje de programación: Python
   - Descarga e instalación de python
   - Python como lenguaje modular
2. Introducción a la sintaxis de Python
   - Importación de módulos y librerías
   - Uso de estructuras condicionales
   - Ciclos
   - Diferencia entre Listas, Arreglos Numpy y Tuplas
3. Entorno de trabajo
   - Spyder, el interprete y la consola
   - Consola vs ipython consola
   - Funciones útiles
4. Lectura de datos en python
   - Método manual
   - Usando las funciones de numpy y scipy
5. Ejemplo de aplicación

Etapas de la programación

1. Análisis del problema
2. Determinar especificaciones
3. Crear el diseño
4. Implementar el diseño
5. Evaluar el programa.
6. Hacer mantenimiento del programa.

Selección del lenguaje de programación

La escogencia de un adecuado lenguaje de programación que satisfaga las necesidades de quien va a resolver el problema, a diferencia de lo que se cree, es una actividad que se lleva a cabo en la etapa tres y no en la uno.

Figura: Logotipo del lenguaje de programación "Python"

Características de python

Entre aquellas más relevantes se pueden destacar:

1. Lenguaje de alto-nivel interpretado (http://www.python.org).
2. Multi-plataforma
3. Es libre y de código abierto.
4. Es modular: Permite importar librerías y funciones propias o desarrolladas por terceros.
5. La sintaxis del lenguaje requiere indentación.
6. Se puede trabajar por medio de linea de comando o mediante la ejecución de un script.

Pasos a seguir

En el proceso de instalación pueden seguirse dos caminos:

- Descargar en instalar los ejecutables para windows que se publican en la página oficial:
  http://www.python.org/download/
  Y luego el paso a seguir será instalar los paquetes o librerías que se requieran.
- Instalar un paquete completo no oficial que viene con las librerías preinstaladas. Entre los cuales se pueden destacar:
  1. Anaconda
  2. Winpython
  3. pythonxy
     (https://code.google.com/p/pythonxy/wiki/Downloads)

Pasos a seguir (Continuación)

- Seleccionar un editor e interprete al gusto. Algunas opciones pueden ser:
  Como editor
  - Spyder
  - Kite
  - Geany (Para Ubuntu)
  - Eclipse
  - notepad ++
  - Gedit
  - vim
  - Emacs
  - Atom, Pycharm, Sublime, **Jupyter**

  Como interprete adicional
  - ipython

Módulos Útiles

- Módulo de cálculo vectorial y cálculo científico: Numpy y Scipy

Figura: Logotipo del paquete scipy (http://www.scipy.org)

- Módulo de graficado

Figura: Logotipo del paquete de Matplotlib (http://www.scipy.org)

Otros módulos a destacar son:

- Modulo OS (Permite usar comandos propios del sistema operativo)
- Modulo glob (Permite listar elementos de una carpeta)
- Modulo Pandas (Facilita el trabajo con series tiempo)
- Modulo time, date y datetime (Permite un manejo óptimo de fechas)
- Cartopy (Permite trabajar con mapas)
- Modulo gdal (Pone a disposición un infinidad de funciones para el análisis geoespacial)

Ejemplos de como importar algunos módulos

```
# La manera de importar los paquetes en python es
import numpy as np
import scipy as sp
import matplotlib.pyplot as plt
import os
import pandas
import my_modulo as mio
```

Estructura condicional: IF .... ELSE, IF...ELIF

```
if x > 0:
    print ('El valor es positivo')
elif x < 0:
    print ('El valor en negativo')
else:
    print ('El numero en la posicion es cero')
```

IF en una linea

```
contador = 0
if (a == b): contador += 1
```

Operadores relacionales

`a == b` *or* `equal(a,b)`	Equal
`a < b` *or* `less(a,b)`	Less than
`a > b` *or* `greater(a,b)`	Greater than
`a <= b` *or* `less_equal(a,b)`	Less than or equal
`a >= b` *or* `greater_equal(a,b)`	Greater than or equal
`a != b` *or* `not_equal(a,b)`	Not Equal

Ejemplo ciclo FOR

```
For i in arreglo:
    print i
```

Ejemplo ciclo WHILE

```
contador = 0
while contador <= 10:
    contador += 1
```

For en una linea

```
test = [(i + 2) for i in list(range(4))]
```

Recordar: Hablar de la sentencia **enumerate**

Definición de una lista

```
Lista = [ Elementos de la lista ]
```

Definición de arreglos de Numpy

```
Arreglo_numpy = numpy.array([ ])
Arreglo_numpy = numpy.empty((), dtype = 'float')
Matriz_numpy = numpy.matrix( arreglo )
```

Definición de tuplas

```
Tupla = (elemento1, elmento2 .....)
```

Gracias