Tema 0 - Programación Básica con Python

Curso de Física Computacional

M. en C. Gustavo Contreras Mayén

Facultad de Ciencias - UNAM

27 de enero de 2020

Contenido

Para el curso de Física Computacional será necesario que usemos un lenguaje de programación para apoyarnos en la solución de los problemas.

El lenguaje de nuestra elección es un medio para alcanzar nuestro objetivo del curso, más no el fin, por lo que revisaremos lo más básico de `Python`, dando la oportunidad de que por tu cuenta, logres un mayor conocimiento y práctica con `Python`.

Si ya cuentas con el conocimiento y práctica de algún otro lenguaje, podrás trabajar con él, para ello, deberás de enviar detallado tu archivo con el código, así como con el ejecutable.

- Lenguaje de programación de alto nivel, interpretado.

- Lenguaje de programación de alto nivel, interpretado.
- Desarrollado por Guido van Rossum a principios de los años 90.

- Lenguaje de programación de alto nivel, interpretado.
- Desarrollado por Guido van Rossum a principios de los años 90.
- Es multiplataforma (UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS, etc.)

- Lenguaje de programación de alto nivel, interpretado.
- Desarrollado por Guido van Rossum a principios de los años 90.
- Es multiplataforma (UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS, etc.)
- Software libre: Python Software Foundation License (PSFL)

- Es usado en todo tipo de aplicaciones: científicas, administración de sistemas, procesamiento de texto, páginas web, bases de datos, visualización 3D y videojuegos, inteligencia artificial, etc.

- Es usado en todo tipo de aplicaciones: científicas, administración de sistemas, procesamiento de texto, páginas web, bases de datos, visualización 3D y videojuegos, inteligencia artificial, etc.
- Presenta una sintaxis compacta, sencilla e intuitiva, una curva de aprendizaje mínima (comparada por ejemplo contra Fortran o Java), junto a una potente librería de funciones y clases.

- Es usado en todo tipo de aplicaciones: científicas, administración de sistemas, procesamiento de texto, páginas web, bases de datos, visualización 3D y videojuegos, inteligencia artificial, etc.
- Presenta una sintaxis compacta, sencilla e intuitiva, una curva de aprendizaje mínima (comparada por ejemplo contra Fortran o Java), junto a una potente librería de funciones y clases.
- Lo anterior permite programar una aplicación completa en cuestión de horas o incluso minutos.

¿Por qué usar Python 3?

- Tiene un diseño simple e internamente consistente.

¿Por qué usar Python 3?

- Tiene un diseño simple e internamente consistente.
- El código es muy claro y legible.

¿Por qué usar Python 3?

- Tiene un diseño simple e internamente consistente.
- El código es muy claro y legible.
- Es un lenguaje de alto nivel: operaciones complejas en muy pocas líneas.

¿Por qué usar Python 3?

- Tiene un diseño simple e internamente consistente.
- El código es muy claro y legible.
- Es un lenguaje de alto nivel: operaciones complejas en muy pocas líneas.
- Cuenta con módulos que facilitan un amplio espectro de tareas.

¿Por qué usar Python 3?

- Tiene un diseño simple e internamente consistente.
- El código es muy claro y legible.
- Es un lenguaje de alto nivel: operaciones complejas en muy pocas líneas.
- Cuenta con módulos que facilitan un amplio espectro de tareas.
- Versatilidad: distintos paradigmas de programación: estructurada, orientada a objetos, funcional.

Simplicidad como filosofía base

1. Bello es mejor que feo.
2. Explícito es mejor que implícito.
3. Simple es mejor que complejo.
4. Complejo es mejor que complicado.
5. Plano es mejor que anidado.
6. Disperso es mejor que denso.
7. La legibilidad cuenta.
8. ...

El código que sigue los principios de `Python` de legibilidad y transparencia se dice que es "pythonico". Contrariamente, el código opaco u ofuscado es bautizado como "no pythonico"

Pythony otros lenguajes de programación

Para calcular el factorial de un número en C:

```
int factorial(int x)
{
    if (x == 0)
        return 1;
    else
        return x * factorial(x - 1);
}
```

En Python:

```
def factorial(x):
    if x == 0:
        return 1
    else:
        return x * factorial(x - 1)
```

Filosofía de "pilas incluidas"

La librería estándar de `Python` incluye módulos para:

- Funciones matemáticas reales y complejas, números aleatorios, criptografía.
- Manejo de conexiones de red (TCP/IP, Web, FTP, correo, etc.)
- Compresión y descompresión de archivos (zlib, gzip, bzip2, tar)
- Manipulación de texto y expresiones regulares.
- Acceso a bases de datos.
- Acceso al sistema operativo, creación de subprocesos, manejo de archivos y directorio.

Contenido

Características de Python

Algunas de las principales características de Python que lo convierten en un lenguaje versátil, son:

- Tipado dinámico.
- Fuertemente tipado.
- Orientado a objetos.

Tipado dinámico

Cada dato es de un tipo determinado y sólo se puede operar con él de formas bien definidas.

La ventaja es que NO hay que declarar variables antes de su uso.

Fuertemente tipado

Se dice que es un lenguaje cuyos tipos son estrictos. Java y `Python` son fuertemente tipados.

Si se tiene un tipo de dato entero, no se puede tratar como una cadena de texto sin convertirlo explícitamente.

Programación orientada a objetos

La programación orientada a objetos es un paradigma de programación que busca representar entidades u objetos agrupando datos y métodos que puedan describir sus características y comportamientos.

Complementos para Python

Complementos para Python

- NumPy: paquete fundamental para computación científica.

Complementos para Python

- NumPy: paquete fundamental para computación científica.
- SciPy: librería para computación científica (extiende a NumPy)

Complementos para Python

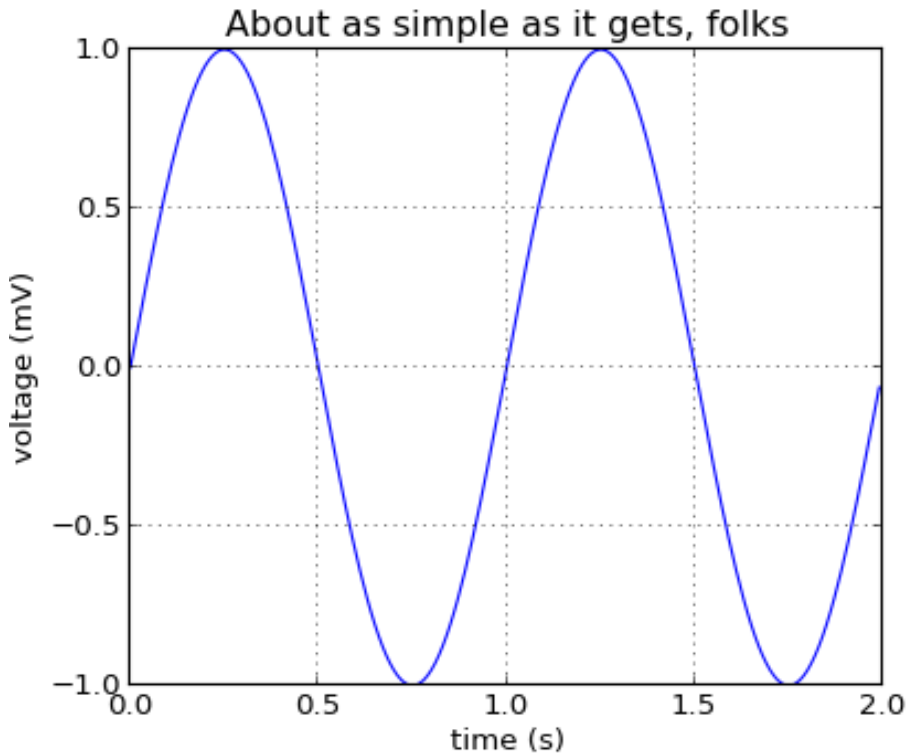
- NumPy: paquete fundamental para computación científica.
- SciPy: librería para computación científica (extiende a NumPy)
- matplotlib: librería para gráficos 2D (soporta gráficos 3D también)

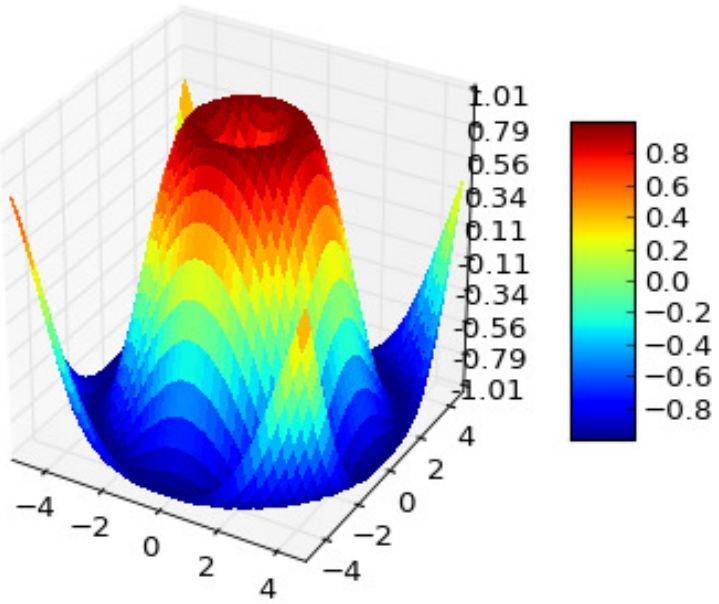
Complementos para Python

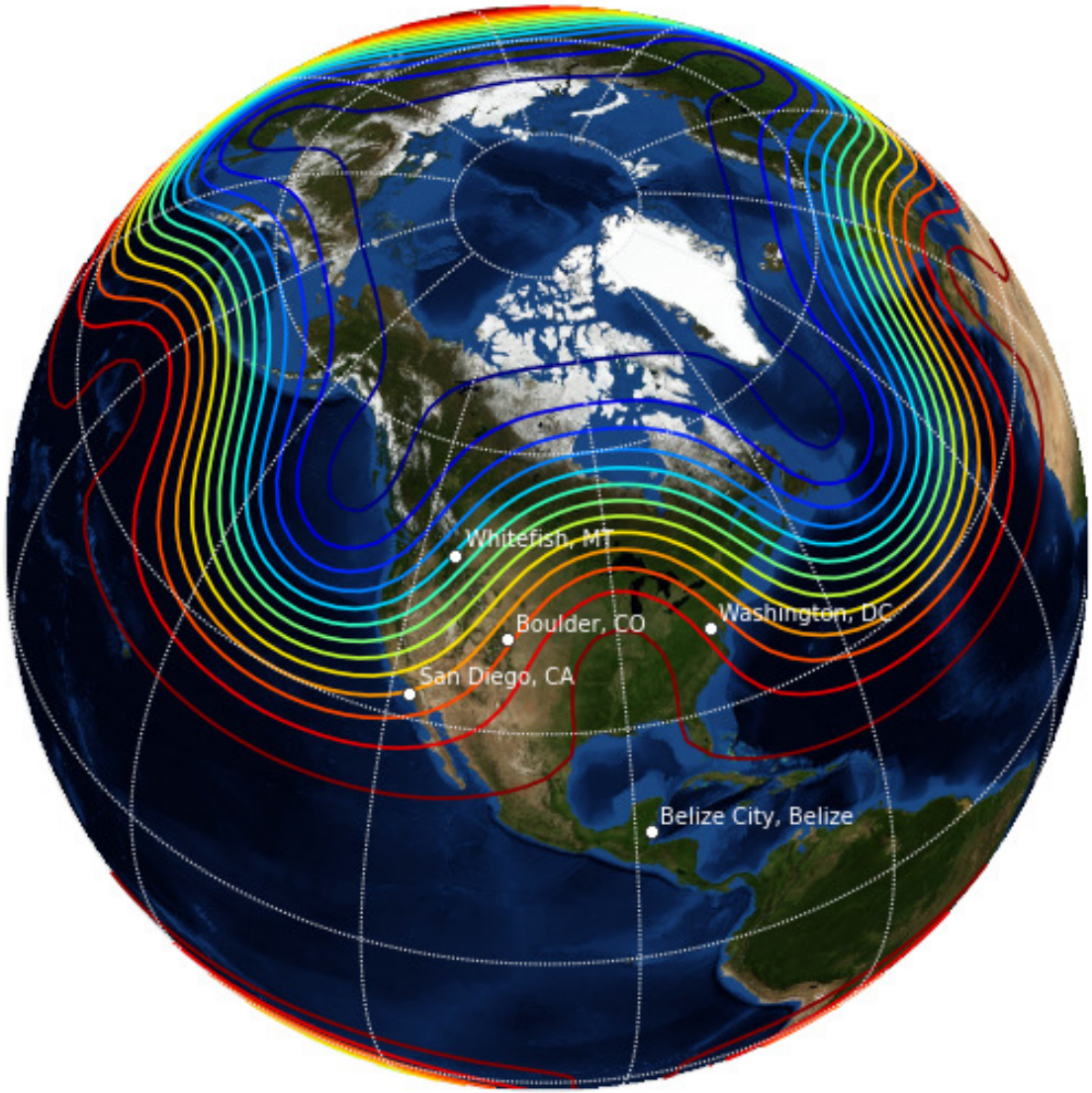
- NumPy: paquete fundamental para computación científica.
- SciPy: librería para computación científica (extiende a NumPy)
- matplotlib: librería para gráficos 2D (soporta gráficos 3D también)
- Mayavi: librería para gráficos y visualización de datos 3D.

Complementos para Python

- NumPy: paquete fundamental para computación científica.
- SciPy: librería para computación científica (extiende a NumPy)
- matplotlib: librería para gráficos 2D (soporta gráficos 3D también)
- Mayavi: librería para gráficos y visualización de datos 3D.
- ipython: consola interactiva para Python.


About as simple as it gets, folks
1.0
0.5
0.0
−0.5
−1.0
voltage (mV)
0.0
0.5
1.0
1.5
2.0
time (s)


1.01
0.79
0.56
0.34
0.11
−0.11
−0.34
−0.56
−0.79
−1.01
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−4
−2
0
2
4
−4
−2
0
2
4


Whitefish, MT
Boulder, CO
Washington, DC
San Diego, CA
Belize City, Belize

Contenido

Trabajando con Python

Hay dos modos de trabajo en Python, cada uno de ellos depende de nuestra habilidad:

- **Modo rudo**: trabajo directo en la consola (uso del intérprete interactivo), sirve para probar pequeñas instrucciones, depurar programas, o buscar ayuda de funciones y métodos.

Trabajando con Python

Hay dos modos de trabajo en Python, cada uno de ellos depende de nuestra habilidad:

- **Modo rudo**: trabajo directo en la consola (uso del intérprete interactivo), sirve para probar pequeñas instrucciones, depurar programas, o buscar ayuda de funciones y métodos.
- **Modo amigable**: a través de una interface IDLE (Entorno de Desarrollo Integrado), usando el código como un programa o script independiente, en el caso de un programa en su forma final o que ya tenga definidas funciones y clases propias.

¿Qué podemos hacer en el modo interactivo?

- El prompt >>> indica que `Python` está listo para recibir instrucciones.
- Si tecleamos una expresión, ésta se evalúa y se muestra directamente el resultado.
- La ayuda para una función se obtiene con `help` y el nombre de la función (entre paréntesis)
  `>>> help(float)`
- La ayuda para una palabra clave se obtiene con `help` y la palabra (entre comillas simples y paréntesis)
  `>>> help('while')`
- Llamar a `help()` (sin argumentos) accede a la ayuda interactiva.

Programa o *script* independiente

```
1  # Este es un programa o script en Python
2  # En una linea , todo lo que sigue a
3  # continuacion de un caracter # es comentario
4
5  print "Hola , Mundo !"
6
7  x = 42
8
9  print x + 8 # imprime 50
10
11 print 'Programar es divertido de nuevo'.split
       ()
12
13 print ([x**2 for x in range (15)])
14 print ([x**2 for x in range (15) if x %2 ==
       0])
```