Taller de Física Computacional

Funciones

Cristián G. Sánchez y Carlos J. Ruestes

2021

Funciones

Funciones

- Ya hemos visto algunas funciones que son parte de la biblioteca estándar como `abs()` o `round(x,[n])`.
- Las funciones en el marco de nuestro modelo de lo que es un programa son sub-programas.
- Sirven para reducir el problema a porciones más pequeñas o simples.
- Es una buena práctica ir armando una biblioteca de funciones para tenerlas a mano.
- En Python las bibliotecas de funciones se guardan en módulos.

Funciones

Funciones en Python

- Las funciones pueden tener un número fijo o variable de argumentos que puede ser cero.
- Las funciones pueden tener argumentos opcionales (como el caso de round).
- Las funciones pueden tener parámetros con nombre.
- Las funciones pueden devolver un valor o no.
- Las funciones pueden devolver más de un valor.
- Las funciones pueden tener efectos colaterales.
- Las variables definidas dentro de una función sólo viven mientras la función se está ejecutando.

Definición de funciones, sintaxis

Usamos las palabras clave `def`, `return`, `None`, el delimitador `:` y los *tokens NUEVA LíNEA, INDENTACIÓN y DEDENTACIÓN*. Las siguientes son definiciones válidas de funciones

```
def identificador(p1,p2,p3=0):
    # pasan cosas
    # pasan más cosas
    return resultado

def identificador(p1,p2):
    # pasan cosas
    # pasan más cosas
    # al no tener "return" esta función devuelve None
```

Definición de funciones, sintaxis

Una función puede devolver más de un valor:

```
def identificador(p1,p2,p3=0):
    # pasan cosas
    # pasan más cosas
    return resultado1, resultado2
```

lo cual se invoca de la forma:

```
a,b = identificador(p1,p2,p3)
```

Alcance (Scope) y Efectos colaterales

- Las variables definidas dentro de una función sólo viven mientras la función se está ejecutando.
- Las funciones pueden tener efectos colaterales, de hecho puede ser lo único que hagan.
- Una función puede utilizar variables variables globales (OJO)

```
def identificador(a,b):
    c = 3.14 # c sólo vive mientras estoy aquí dentro
    c *= variable_global # Puedo usar una variable global
                         # si esté definida afuera
    print(c) # Este es un efecto colateral
    resultado = a + b + c
    return resultado
```

Definición de funciones, sintaxis

Las funciones pueden tener un número variable de argumentos tanto nombrados como sin nombre, para ello se utilizan las estructuras de datos *lista* y *diccionario* que veremos más adelante.

```
def identificador(*argumentos):
    # pasan cosas
    # pasan más cosas
    # los argumentos entran por una "lista"
    return resultado

def identificador(**argumentos):
    # pasan cosas
    # pasan más cosas
    # los argumentos entran por un "diccionario"
    return resultado
```

Síntesis y recursos:

- Manual de referencia de Python
- Manual de la Librería estándar de Python