Pràctica 3: Funcions i Paquets

8 de desembre de 2020

1 Definir funcions i ús de paquets

1.1 Funcions

El **Python** permet definir funcions amb la instrucció `def`. Una funció té un nom, se li entra(en) un argument(s) (que pot ser buit) i ens retorna un resultat (que pot ser un simple valor, un gràfic, escriure a un fitxer, ...). Per fer aquest procediment, pot contenir una rutina molt llarga o bé tant sols una instrucció.

[1]:
```
def quadrat(x):
    return(x**2)
```

[2]:
```
quadrat(8)
```

[2]:
```
64
```

[3]:
```
quadrat(11.5)
```

[3]:
```
132.25
```

El nombre d'arguments pot ser més d'un i fins i tot podem fixar uns valors per defecte. Podem utilitzar el format llista (o conjunt) per a retornar més d'un valor:

[4]:
```
def f(x=2,y=5):
    return([x,y,x+y,x*y])
```

[5]:
```
f()
```

[5]:
```
[2, 5, 7, 10]
```

[6]:
```
f(10,20)
```

[6]:
```
[10, 20, 30, 200]
```

[7]:
```
f(y=20)
```

[7]:
```
[2, 20, 22, 40]
```

Si escrivim * davant el nom d'un argument, s'entèn que pot contenir més d'un valor:

[8]:
```
def suma(*sumands):
    s=0
    for i in sumands:
        s=s+i
    return s
```

[9]:
```
suma(23)
```

[9]: 23

[10]:
```
suma(1,3,5,7)
```

[10]: 16

Les variables que s'utilitzen dins d'una funció (per exemple, la `s` i la `i` en la definició de `suma`) són variables que es creen tant sols tenen valor dins de les fucions (variables *locals*).

[11]:
```
s
```

```
---------------------------------------------------------------------------

NameError                                 Traceback (most recent call last)

<ipython-input-11-ded5ba42480f> in <module>
----> 1 s


NameError: name 's' is not defined
```

[12]:
```
s=12
print(suma(2,4,6,8))
print(s)
```

```
20
12
```

Les variables que s'han definit abans (variables *globals* externes a la funció) es poden utilitzar dins de les funcions, però les modificacions que en fem dins de la funció no es reflecteixen al valor extern.

[13]:
```
def g():
    global x
    y=5
    print("La variable x val",x,"i és global")
    print("La variable y val",y,",és local i no importa si hi ha una altra y␣
 ↪definida fora de la funció")
```

```
    x=x*2
    y=y*2
    print("Hem doblat tant la x com la y",x,y)
    return
```

[14]:
```
x=10
y=99
```

[15]:
```
g()
```

```
La variable x val 10 i és global
La variable y val 5 ,és local i no importa si hi ha una altra y definida fora de
la funció
Hem doblat tant la x com la y 20 10
```

[16]:
```
print(x)
print(y)
```

```
20
99
```

1.1.1 Exemple

La funció següent retorna una llista amb els `n` primers números de la successió de Fibonacci:

[17]:
```
def fib(n):
    if n<=0: return
    if n==1: return [1]
    l=[1,1]
    for i in range(n-2):
        l.append(l[-1]+l[-2])
    return l
```

[18]:
```
fib(1)
```

[18]:
```
[1]
```

[19]:
```
fib(10)
```

[19]:
```
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```

1.1.2 Exercicis

1. Feu una funció anomenada `mcd` que, donats dos nombres enters `m` i `n` calculi:
2. Primer la resta de dividir `m` per `n` amb el nom `r`.
3. Consideri `m=n` i `n=r` i torni a fer la mateixa divisió d'abans, fins que trobi `r=0`.
4. Retorni el valor anterior a d'`r` (l'últim abans de valer 0). Comproveu-ne el funcionament fent `mcd(4539,5049)`.

5. Feu una funció anomenada `divisors` que, donat un nombre enter `n` retorni una llista amb tots els divisors d'`n`. Comproveu-ne el funcionament fent `divisors(1024)` i `divisors(4410)`.
6. Feu una funció anomenada `separa` que, donada una cadena `s` i un caràcter `c`, retorni una llista amb les subcadenes d'`s` separades pel caràcter `c`. Per exemple, `separa("Hola a tothom"," ")` hauria de retornar `["Hola","a","tothom"]`.

1.2 Ús de paquets

Es poden carregar paquets per ampliar el nombre de funcions que venen implementades al **Python**. Dependrà de la instal·lació que es pugui carregar un paquet o no, però les que veurem a aquest curs es poden trobar a les instal·lacions més utilitzades, com **Anaconda**.

La instrucció per a cridar les funcions d'un paquet és la `import` i accepta arguments per a carregar totes les funcions d'un paquet concret, o bé tant sols unes quantes i fins i tot amb quin nom o prefix volem treballar.

En general, quan carreguem un paquet, el nom de les funcions utilitzen el nom del paquet com a prefix o bé l'alias que li entrem.

A aquesta pràctica tant sols carregarem algunes funcions per a veure'n el funcionament. Algunes de les pràctiques posteriors estaran dedicades a alguns paquets específics.

1.3 Exemple: el paquet `os`

El paquet `os` (*operating system*) permet carregar funcions que interactuen amb el sistema operatiu de l'ordinador:

[20]:
```
import os
```

[21]:
```
os.getcwd() # Ens mostra el directori on estem treballant
```

[21]:
```
'/home/albert/Cloud/owncloudTom/Shared/EIE-GEA/2020-21/Python'
```

Si utilitzem el nom del paquet que hem carregat com a instrucció, ens mostra la informació que té (en particular, quin s'ha carregat si tenim més d'una instal·lació de **Python**):

[22]:
```
os
```

[22]:
```
<module 'os' from '/usr/lib/python3.8/os.py'>
```

La instrucció `dir` permet obtenir un llistat de les funcions que conté un paquet, mentre que podem utilitzar la funció `help` per obtenir-ne ajuda:

[23]:
```
dir(os)
```

[23]:
```
['CLD_CONTINUED',
 'CLD_DUMPED',
 'CLD_EXITED',
 'CLD_TRAPPED',
 'DirEntry',
```

```
'EX_CANTCREAT',
'EX_CONFIG',
'EX_DATAERR',
'EX_IOERR',
'EX_NOHOST',
'EX_NOINPUT',
'EX_NOPERM',
'EX_NOUSER',
'EX_OK',
'EX_OSERR',
'EX_OSFILE',
'EX_PROTOCOL',
'EX_SOFTWARE',
'EX_TEMPFAIL',
'EX_UNAVAILABLE',
'EX_USAGE',
'F_LOCK',
'F_OK',
'F_TEST',
'F_TLOCK',
'F_ULOCK',
'GRND_NONBLOCK',
'GRND_RANDOM',
'MFD_ALLOW_SEALING',
'MFD_CLOEXEC',
'MFD_HUGETLB',
'MFD_HUGE_16GB',
'MFD_HUGE_16MB',
'MFD_HUGE_1GB',
'MFD_HUGE_1MB',
'MFD_HUGE_256MB',
'MFD_HUGE_2GB',
'MFD_HUGE_2MB',
'MFD_HUGE_32MB',
'MFD_HUGE_512KB',
'MFD_HUGE_512MB',
'MFD_HUGE_64KB',
'MFD_HUGE_8MB',
'MFD_HUGE_MASK',
'MFD_HUGE_SHIFT',
'MutableMapping',
'NGROUPS_MAX',
'O_ACCMODE',
'O_APPEND',
'O_ASYNC',
'O_CLOEXEC',
'O_CREAT',
```

```
'O_DIRECT',
'O_DIRECTORY',
'O_DSYNC',
'O_EXCL',
'O_LARGEFILE',
'O_NDELAY',
'O_NOATIME',
'O_NOCTTY',
'O_NOFOLLOW',
'O_NONBLOCK',
'O_PATH',
'O_RDONLY',
'O_RDWR',
'O_RSYNC',
'O_SYNC',
'O_TMPFILE',
'O_TRUNC',
'O_WRONLY',
'POSIX_FADV_DONTNEED',
'POSIX_FADV_NOREUSE',
'POSIX_FADV_NORMAL',
'POSIX_FADV_RANDOM',
'POSIX_FADV_SEQUENTIAL',
'POSIX_FADV_WILLNEED',
'POSIX_SPAWN_CLOSE',
'POSIX_SPAWN_DUP2',
'POSIX_SPAWN_OPEN',
'PRIO_PGRP',
'PRIO_PROCESS',
'PRIO_USER',
'P_ALL',
'P_NOWAIT',
'P_NOWAITO',
'P_PGID',
'P_PID',
'P_WAIT',
'PathLike',
'RTLD_DEEPBIND',
'RTLD_GLOBAL',
'RTLD_LAZY',
'RTLD_LOCAL',
'RTLD_NODELETE',
'RTLD_NOLOAD',
'RTLD_NOW',
'RWF_DSYNC',
'RWF_HIPRI',
'RWF_NOWAIT',
```

```
'RWF_SYNC',
'R_OK',
'SCHED_BATCH',
'SCHED_FIFO',
'SCHED_IDLE',
'SCHED_OTHER',
'SCHED_RESET_ON_FORK',
'SCHED_RR',
'SEEK_CUR',
'SEEK_DATA',
'SEEK_END',
'SEEK_HOLE',
'SEEK_SET',
'ST_APPEND',
'ST_MANDLOCK',
'ST_NOATIME',
'ST_NODEV',
'ST_NODIRATIME',
'ST_NOEXEC',
'ST_NOSUID',
'ST_RDONLY',
'ST_RELATIME',
'ST_SYNCHRONOUS',
'ST_WRITE',
'TMP_MAX',
'WCONTINUED',
'WCOREDUMP',
'WEXITED',
'WEXITSTATUS',
'WIFCONTINUED',
'WIFEXITED',
'WIFSIGNALED',
'WIFSTOPPED',
'WNOHANG',
'WNOWAIT',
'WSTOPPED',
'WSTOPSIG',
'WTERMSIG',
'WUNTRACED',
'W_OK',
'XATTR_CREATE',
'XATTR_REPLACE',
'XATTR_SIZE_MAX',
'X_OK',
'_Environ',
'__all__',
'__builtins__',
```

```
'__cached__',
'__doc__',
'__file__',
'__loader__',
'__name__',
'__package__',
'__spec__',
'_check_methods',
'_execvpe',
'_exists',
'_exit',
'_fspath',
'_fwalk',
'_get_exports_list',
'_putenv',
'_spawnvef',
'_unsetenv',
'_wrap_close',
'abc',
'abort',
'access',
'altsep',
'chdir',
'chmod',
'chown',
'chroot',
'close',
'closerange',
'confstr',
'confstr_names',
'copy_file_range',
'cpu_count',
'ctermid',
'curdir',
'defpath',
'device_encoding',
'devnull',
'dup',
'dup2',
'environ',
'environb',
'error',
'execl',
'execle',
'execlp',
'execlpe',
'execv',
```

```
'execve',
'execvp',
'execvpe',
'extsep',
'fchdir',
'fchmod',
'fchown',
'fdatasync',
'fdopen',
'fork',
'forkpty',
'fpathconf',
'fsdecode',
'fsencode',
'fspath',
'fstat',
'fstatvfs',
'fsync',
'ftruncate',
'fwalk',
'get_blocking',
'get_exec_path',
'get_inheritable',
'get_terminal_size',
'getcwd',
'getcwdb',
'getegid',
'getenv',
'getenvb',
'geteuid',
'getgid',
'getgrouplist',
'getgroups',
'getloadavg',
'getlogin',
'getpgid',
'getpgrp',
'getpid',
'getppid',
'getpriority',
'getrandom',
'getresgid',
'getresuid',
'getsid',
'getuid',
'getxattr',
'initgroups',
```

```
'isatty',
'kill',
'killpg',
'lchown',
'linesep',
'link',
'listdir',
'listxattr',
'lockf',
'lseek',
'lstat',
'major',
'makedev',
'makedirs',
'memfd_create',
'minor',
'mkdir',
'mkfifo',
'mknod',
'name',
'nice',
'open',
'openpty',
'pardir',
'path',
'pathconf',
'pathconf_names',
'pathsep',
'pipe',
'pipe2',
'popen',
'posix_fadvise',
'posix_fallocate',
'posix_spawn',
'posix_spawnp',
'pread',
'preadv',
'putenv',
'pwrite',
'pwritev',
'read',
'readlink',
'readv',
'register_at_fork',
'remove',
'removedirs',
'removexattr',
```

```
'rename',
'renames',
'replace',
'rmdir',
'scandir',
'sched_get_priority_max',
'sched_get_priority_min',
'sched_getaffinity',
'sched_getparam',
'sched_getscheduler',
'sched_param',
'sched_rr_get_interval',
'sched_setaffinity',
'sched_setparam',
'sched_setscheduler',
'sched_yield',
'sendfile',
'sep',
'set_blocking',
'set_inheritable',
'setegid',
'seteuid',
'setgid',
'setgroups',
'setpgid',
'setpgrp',
'setpriority',
'setregid',
'setresgid',
'setresuid',
'setreuid',
'setsid',
'setuid',
'setxattr',
'spawnl',
'spawnle',
'spawnlp',
'spawnlpe',
'spawnv',
'spawnve',
'spawnvp',
'spawnvpe',
'st',
'stat',
'stat_result',
'statvfs',
'statvfs_result',
```

```
 'strerror',
 'supports_bytes_environ',
 'supports_dir_fd',
 'supports_effective_ids',
 'supports_fd',
 'supports_follow_symlinks',
 'symlink',
 'sync',
 'sys',
 'sysconf',
 'sysconf_names',
 'system',
 'tcgetpgrp',
 'tcsetpgrp',
 'terminal_size',
 'times',
 'times_result',
 'truncate',
 'ttyname',
 'umask',
 'uname',
 'uname_result',
 'unlink',
 'unsetenv',
 'urandom',
 'utime',
 'wait',
 'wait3',
 'wait4',
 'waitid',
 'waitid_result',
 'waitpid',
 'walk',
 'write',
 'writev']
```

[24]:
```
help(os.mkdir)
```

```
Help on built-in function mkdir in module posix:

mkdir(path, mode=511, *, dir_fd=None)
    Create a directory.

    If dir_fd is not None, it should be a file descriptor open to a directory,
      and path should be relative; path will then be relative to that directory.
    dir_fd may not be implemented on your platform.
      If it is unavailable, using it will raise a NotImplementedError.
```

```
The mode argument is ignored on Windows.
```

1.3.1 Exemple: el paquet `numpy`

El paquet `numpy` té moltes funcions numèriques implementades. En particular, un valor de π i les funcions trigonomètriques:

[25]:
```
import numpy as np
```

[26]:
```
print(np.pi)
```

```
3.141592653589793
```

[27]:
```
print(np.sin(np.pi))
```

```
1.2246467991473532e-16
```

Si d'un paquet concret tant sols en volem unes funcions i no volem utilitzar els prefixos per a cridar-les, podem utilitzar la sintaxi següent:

[28]:
```
from numpy import sin,cos,pi
```

[29]:
```
print(pi)
```

```
3.141592653589793
```

[30]:
```
print(cos(pi))
```

```
-1.0
```

El paquet `numpy` també conté funcions que permeten definir vectors i matrius, afegint més opcions al fet de ser llistes:

[31]:
```
import numpy as np
```

[32]:
```
matriuidentitat=np.eye(5)
```

[33]:
```
print(matriuidentitat)
```

```
[[1. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0.]
 [0. 0. 1. 0. 0.]
 [0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 1.]]
```

[34]:
```
matriuuns=np.ones((5,5))
```

[35]:
```
print(matriuuns)
```

```
[[1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]]
```

[36]:
```
print(3*matriuuns-2*matriuidentitat)
```

```
[[1. 3. 3. 3. 3.]
 [3. 1. 3. 3. 3.]
 [3. 3. 1. 3. 3.]
 [3. 3. 3. 1. 3.]
 [3. 3. 3. 3. 1.]]
```

1.3.2 Exercicis

Suposem que hem carregat el paquet `numpy` com a `np`. 1. Què fa la funció `X = np.random.rand(5, 6)`? 1. I, si tenim definit `X` com a l'apartat anterior, què fan `X.mean(axis=0)` i `X.mean(axis=1)`?