4.1.10. Evaluación con cortocircuitos

La evaluación de expresiones lógicas es algo especial. Observa la condición de este **if**:

```
if a == 0 or 1/a > 1:
    ...
```

¿Puede provocar una división por cero? No, nunca. Observa que si *a* vale cero, el primer término del **or** es *True*. Como la evaluación de una *o lógica* de *True* con cualquier otro valor, *True* o *False*, es necesariamente *True*, Python *no evalúa* el segundo término y se ahorra así un esfuerzo innecesario.

Algo similar ocurre en este otro caso:

```
if a != 0 and 1/a > 1:
    ...
```

Si *a* es nulo, el valor de *a* **!=** 0 es falso, así que ya no se procede a evaluar la segunda parte de la expresión.

Al calcular el resultado de una expresión lógica, Python evalúa (siguiendo las reglas de asociatividad y precedencia oportunas) lo justo hasta conocer el resultado: cuando el

primer término de un **or** es cierto, Python acaba y devuelve directamente cierto y cuando el primer término de un **and** es falso, Python acaba y devuelve directamente falso. Este modo de evaluación se conoce como *evaluación con cortocircuitos*.

EJERCICIOS

▶ 89 ¿Por qué obtenemos un error en esta sesión de trabajo con el intérprete interactivo?

```
>>> a = 0 ↲
>>> if 1/a > 1 and a != 0: ↲
...     print a ↲
... ↲
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
ZeroDivisionError: integer division or modulo by zero
```