FUNDAMENTOS DE COMPUTACIÓN
GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES
Universidad de Cantabria

Práctica 3 (laboratorio)
Octubre 2019

Objetos númericos en PYTHON

En PYTHON los datos toman la forma de objetos (objects), y cada objeto tienen una clase (class) que define que tipo de operadores y métodos podemos usar con ellos. La función `type()` nos muestra la clase.

---

1. **Algunos objetos númericos en PYTHON :**

   - **enteros**: números enteros (INT).

   ```
   >>> 124
   124
   >>> type(-344)
   <class 'int'>
   ```

   - **números en coma flotante**: números decimales (FLOAT) : $\pm M \times B^E$ o $\pm M \times B^e$, donde M es la mantisa, B es la base y E o e es el exponente. La base en la representación decimal es 10 y el exponente se separa de la mantisa con la letra e o E. La codificación utilizada en PYTHON es el *IEEE standard 754* con precisión doble 53 bits de precisión (se explicará en clase de teoría). En español la parte fraccionaria de un número se separa de la parte entera por una coma, la norma anglosajona es con un punto

   ```
   >>> 56.3
   56.3
   >>> 45e2
   4500.0
   >>> 45E-3
   0.045
   >>> type(4.67)
   <class 'float'>
   >>> 2,1
   (2, 1)
   >>> type((2,1))
   <class 'tuple'>
   ```

   - **números complejos** (COMPLEX): La raíz imaginaria de la unidad (el número i tal que $i^2 = -1$) se representa por 1j.
     Los métodos `real` y `imag` extrae la parte real y la parte imaginaria (respectivamente) del número complejo.

   ```
   >>> 1j
   1j
   >>> 3+5j
   (3+5j)
   >>> (3+5j).real
   3.0
   >>> (3+5j).imag
   5.0
   >>> type(4.2+3.7j)
   <class 'complex'>
   >>> j
   Traceback (most recent call last):
     File "<stdin>", line 1, in <module>
   NameError: name 'j' is not defined
   ```

2. **Operadores aritméticos**:

   Una expresión es una combinación de objetos y operadores. La sintaxis para una expresión simple es:

   `<objeto>  <operador>  <objeto>`

Operador	Operación	Ejemplo	Resultado	Prioridad
**	Exponenciación	2**3	8	1
*	Multiplicación	8.5*2.5	21.25	2
/	División	11/3	3.6666666666666665	2
//	Cociente	11//3	3	2
%	Resto o módulo	29 %8	5	2
+	Suma	11.1+3	14.1	3
-	Resta	5-3j-19	(-14-3j)	3

El resultado de los operadores aritméticos depende del tipo de dato de los operandos. Analizamos el operador *módulo o resto* , % y el operador *división*, //:

El cociente y resto de la división Euclídea. *Dados dos números enteros a y b tales que $b > 0$, existe un único par de números enteros q y r tales que :*

$$a = b \times q + r,$$

verificando que $0 \leq r < b$. Se dice que q y r son el cociente y resto (respectivamente) de la división Euclídea de a entre b

Por ejemplo, $a = 45$ y $b = 7$, tenemos que $45 = 7 \times 6 + 3$, cociente 6 y resto 3. Si $a = -45$ y $b = 7$, entonces $-45 = 7 \times (-7) + 4$, cociente -7 y resto 4. En las condiciones de arriba los operadores // y % se obtiene el cociente y el resto (respectivamente) de la división Euclídea.

```
>>> -45%7
4
```

```
>>> -45//7
-7
```

Cuando en una expresión aparecen varias operadores, PYTHON las efectúa aplicando las reglas usuales de prioridad de las operaciones, como se ilustra en la tabla. Y con la misma prioridad se ejecutan de izquierda a derecha, excepto la exponenciación, que se ejecuta de derecha a izquierda. En caso de querer que las operaciones se realicen en otro orden, se pueden utilizar paréntesis.

```
>>> 2*4+5
13
>>> 2*(4+5)
18
>>> -2*2
-4
>>> -3//2
-2
>>> 12/3*2
8.0
>>> 2+3*1+2
7
```

```
>>> 12/(3*2)
2.0
>>> 2+3**2*5
47
>>> 2+(3**2)*5
47
>>> 2**1/3
0.6666666666666666
>>> 2**(1/3)
1.2599210498948732
>>> 2**1**3
2
```

Se tiene que los número enteros es un subconjunto de los reales y, estos últimos de los números complejos. Por lo tanto, en PYTHON si en la expresión `<objeto> <operador> <objeto>` uno de los objeto es de una `class` digamos más grande, entonces el valor de la expresión es un objeto de esa clase:

```
>>> 1.1 + 2
2.1
```

3. **Errores de tecleo y excepciones.** Si introducimos una expresión incorrectamente, PYTHON nos lo indicará con un mensaje de error.

```
 0+1)
  File "<stdin>", line 1
    0+1)
       ^
SyntaxError: invalid syntax
>>> 1/0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
```

El primero es un error de sintaxis (*SyntaxError*) y el segundo es de naturaleza distinta, se trata de un error de división por cero.

4. El entero más grande está limitado por las características del computador (memoria, etc.). Sin embargo, el número decimal más grande que acepta PYTHON es 10^{308} y el más próximo a cero es 10^{-323}

```
>>> 10**308+0.1
1e+308
>>> 10.02**308
Traceback (most recent call last):
  File "<pyshell#79>", line 1, in <module>
    10.02**308
OverflowError: (34, 'Result too large')
>>> 10**308+1
100000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000001
```

5. El módulo o librería MATH es un conjunto de funciones matemáticas de PYTHON . Para cargar todas las funciones del modulo:

```
>>> from math import*
>>> sin(34)
0.5290826861200238
>>> cos(34)
-0.8485702747846052
>>> sin(34)**2+cos(34)**2
1.0
>>> log(256,2)
8.0
>>> log(256)
5.545177444479562
```

Para cargar solamente las funciones `sin` y `cos`.

```
 >>> from math import sin, cos
 >>> sin(34)
0.5290826861200238
```

Otra forma de ejecutar las funciones del modulo MATH:

```
 >>> import math
 >>> math.sin(34)
0.5290826861200238
```

Para obtener ayuda sobre un módulo.

```
>>> import math
>>> help(math)
```

6. **Ejercicios**

- 4-1 ¿Qué resultará de ejecutar las siguientes expresiones en PYTHON ? Haz los cálculos a mano y comprueba el resultado con el ordenador. Algunos de ellas pueden producir errores de tecleo y/o excepciones.
  - a-1) $2+3*2+2$ a-2) $(2+3)*2+2$ a-3) $+---2$ a-4) $-+-+5$ a-5) $1+*3$
  - b-1) $1+2**2*3$ b-2) $1+((2**3)*5$ b-3) $1+2+3+0.5$ b-4) $1.0+3/2$ b-5) $1.0+3\,\%2$
  - c-1) 1/2/4.0 c-2) 1/2.0/4 c-3) 1//2/4 c-4) 4.0**(1/2)+1//2 a-5) 4.0**(1.0/2)+1/2.0
  - d-1) 1.0/2//4 d-2) 4**.5 d-3) 4.0**(1//2) d-4) 3e3//10 d-5) 10//5E-3
  - e-1) 3/2 +1 e-2) 1j**2 e-3) 4. %3 e-4) 1-3*y+4 d-5) 3.2+1
- 4.2 Traduce las siguientes expresiones matemáticas a PYTHON y evalúalas. Trata de utilizar el menor número posible de paréntesis:
  - a) $2+(3(6/2))$
  - b) $\frac{4+6}{2+3}$
  - c) $(4/2)^5$
  - d) $(4/2)^{4+2^2}$

- e) $(-3)^2$
- f) $-(3^2)$

- Calcular:
  - a) $\log_4(234)$
  - b) $\cos(2.34\pi)$
  - c) $\sqrt{678}$
  - d) $e + \pi$, donde e es el número de Euler o constante de Napier.
  - e) mcd(90765,135139) (el máximo común divisor)