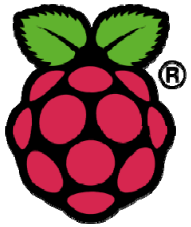

Programación en Python con Raspberry Pi

Tipos básicos

Carolina Mañoso, Ángel P. de Madrid y Miguel Romero

Índice

- Palabras reservadas
- Variables
- Tipo Número
  - Operadores
  - Práctica
- Tipo Booleano
  - Operadores
  - Práctica
- Tipo Cadena de caracteres
  - Práctica
- Referencias

Palabras reservadas

- Hay 33 palabras reservadas:
  - No se puede declarar variables, objetos o clases con estos nombres:

```
False, None, True, and, as, assert, break,
  class, continue, def, del, elif, else,
  except, finally, for, from, global, if,
  import, in, is, lambda, nonlocal, not,
  or, pass, raise, return, try, while,
  with, yield
```

```
>>> help()
Help> keywords
```

Variables (1/2)

- El signo igual = se utiliza para asignar un valor (números, booleanos, cadenas, ...) a una variable.
- El tipo de la variable será el del dato asignado, no se declara el tipo de la variable al crearla.

```
>>>a = 5 #Esta variable es de tipo entero
>>>c = "Hola mundo" #Esta variable es de tipo cadena
```

- Si una variable no está definida (no tiene un valor asignado) al intentar usarla producirá un error.

```
>>>b
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    b
NameError: name 'b' is not defined
```

- Por convención el nombre comienza en minúscula y si son varias palabras se unen por guión bajo.

Variables (2/2)

- En modo interactivo, la última expresión impresa se asigna a la variable "_"

```
>>> n_1 = 5
>>> n_2 = 6
>>> n_1 + n_2
11
>>> _
11
>>> total = _
>>> total
11
```

- Lo tipos básicos de Python son:
  - Números (entero, real y complejo)
  - Valores booleanos
  - Cadena de caracteres

Tipo Número

- Basta asignar un número a una variable para que ésta sea de tipo `int`

```
>>> num_int = 15
```

- Basta asignar un número con parte decimal a una variable para que ésta sea de tipo `float`

```
>>> num_real = 15.4
```

- Se puede expresar en notación científica añadiendo una `e` para indicar el exponente en base 10:

```
>>> num_cient = 0.15e-3
```

- Se puede asignar números complejos:

```
>>> num_complejo = 1.4+5.3j
```

- Para saber el tipo de dato asociado a una variable:

```
>>> type(num)
```

- `int()` convierte a entero y `float()` a float

Operadores aritméticos

Operador	descripción
+	Suma
-	Resta
*	Multiplicación
**	Potencia
/	División
//	División, se obtiene parte entera del cociente
%	Módulo, se obtiene el resto

```
>>> suma = 3 + 2 #el valor de la var suma es 5
>>> cont += 1 #equivalente a cont = cont + 1
```

Operadores binarios

Operador	descripción
&	and
|	or
~	not
<<	Desplazamiento a la izquierda
>>	Desplazamiento a la derecha

```
>>> b1 = 3 & 2 #11+10=10 (igual a 2)
>>> b2 = 3 >> 1 #b2 es igual a 1
```

Práctica tipo Número

- El interprete actúa como calculadora

```
>>> 16 / 3
>>> 12 / 4 #la división siempre retorna un float
>>> 16 // 3
>>> 16 % 3
>>> 5 * 4
>>> 5 ** 4
>>> (40 - 2*6)/4 # reglas de precedencia
>>> suma = 3 + 2 #el valor de la var suma es 5
>>> suma += 1 #equivalente a cont = cont + 1
>>> type(suma)
>>> float(suma)
>>> type(suma)
>>> suma =float(suma)
>>> 10/5e-3
```

Tipo Booleano (1/2)

- Una variable de tipo booleano (`bool`) sólo puede tomar dos valores: `True` (cierto) y `False` (falso).
- Para trabajar con ellas están los operadores lógicos:

Operador	descripción
and	Y
or	O
not	No

```
>>> r = y and x # si y es true y x false, r es false
```

Tipo Booleano (2/2)

- Los valores booleanos son el resultado de expresiones con operadores relacionales:

Operador	descripción
==	Igual a
!=	Distinto de
<	Menor que
>	Mayor que
<=	Menor o igual que
>=	Mayor o igual que

```
>>> r = 5 > 3 # r es true
```

Práctica tipo Booleano

- Introduzaca en el intérprete de comandos:

```
>>> True and False
>>> not True
>>> True or True
>>> r = 5 > 3 # r es true
>>> 2 == 1
>>> 1 != 3
>>> 2 < 1
```

Tipo Cadena de caracteres (1/5)

- Las cadenas de caracteres son texto encerrado entre comillas simples o dobles.

```
>>> cadena1 = "hola"
>>> cadena2 = 'hola'
```

- No existe el tipo de datos `caracter`.
- El caracter de escape `\` puede ser usado para escapar de las comillas o usar el otro tipo de comillas

```
>>> 'doesn\'t' >>> "doesn't"
```

- La función `print()` produce como salida la cadena que encierra entre los paréntesis, omitiendo las comillas que la encierran.
  - En Python 2 no hace falta paréntesis
  - `\n` indica cambio de línea

```
>>> print('primera línea\n segunda línea')
```

Tipo Cadena de caracteres (2/5)

- Una cadena *raw* debe interpretarse tal como se escribe, es decir, se omiten los caracteres especiales expresados con la barra invertida `\`. Las cadenas raw se escriben entrecomilladas y con el carácter `r` precediéndolas:

```
>>> print(r'primera línea\n segunda línea')
```

- Si la cadena ocupa varias líneas se encierra entre tripe entrecomillado.

```
>>> print("""primera línea
segunda línea""")
```

- Las cadenas de caracteres se pueden concatenar (`+`) y repetir (`*`)

```
>>> cadena = 'hola' + 'hola' #cadena = "holahola"
>>> cadena = 'hola' * 3 #cadena = "holaholahola"
```

Tipo Cadena de caracteres (3/5)

- Las cadenas son secuencias y como tal se gestionan:
  - `len()`, devuelve la longitud de la cadena
  - Las cadenas se pueden recorrer, indexándolas. El primer índice: 0

```
"abcde"[1] # es igual a "b"
```

  - Se pueden usar índices negativos (-1, es el último)

```
"abcde"[-1] # es igual a "e"
```

  - Se pueden utilizar intervalos [i : j], donde i está incluida y j excluida

```
"abcde"[1:3] # es igual a "bc"
```

  - El intervalo [: y], todos los caracteres hasta el y

```
"abcde"[:3] # es igual a "abc"
```

  - El intervalo [x:], todos los caracteres desde x al final

```
"abcde"[3:] # es igual a "de"
```

Tipo Cadena de caracteres (4/5)

- Este tipo de dato es inmutable: no se puede sobreescribir. Hay que renombrar. Toda operación sobre cadenas está definida para crear otra nueva como resultado.

```
>>> cadena = "inmutable"
>>> cadena[0] = "n"
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    cadena[0]= "n"
TypeError: 'str' object does not support item
  assignment
>>> cadena = "n" + cadena[1:]
```

- Para convertir al tipo Cadena de caracteres usamos `srt()`

```
>>> cadena = str(3)+str(3)
```

Tipo Cadena de caracteres (5/5)

- Además, las cadenas de caracteres poseen *métodos* específicos para su gestión:
  - `upper()` devuelve una cadena en mayúsculas
  - `lower()` devuelve una cadena en minúsculas
  - `swapcase()` convierte mayúsculas a minúsculas y viceversa
  - `find("subcadena")` devuelve un entero que representa la posición donde se inicia la subcadena dentro de otra
  - `count("subcadena")` devuelve la cantidad de apariciones de la subcadena dentro de otra
  - `replace("subcad busqueda ", "subcad nueva")` devuelve la cantidad de apariciones de la subcadena dentro de otra
  - `strip("caracter")` elimina caracteres a la izquierda y derecha de una cadena

Práctica tipo Cadenas de caracteres

- Introduce en el interprete de comandos:

```
>>> cad_e = " esta cadena es muy larga "
>>> len(cad_e)
>>> cad = cad_e.strip(" ")
>>> len(cad)
>>> cad.find("muy")
>>> CAD = cad.upper()
>>> cad = CAD.lower()
>>> type(cad)
>>> cad[4]
>>> cad[-4]
>>> cad[1:3]
>>> cad[:3]
>>> cad[3:]
>>> cad + CAD
```

Referencias

- La documentación oficial, *Python Tutorial Release 3.5.2*. Guido van Rossum and the Python development team. Python Software Foundation. Junio 25, 2016. https://docs.python.org/3/download.html
- *Learning Python 5th edition*. Mark Lutz. O'Reilly 2013
- *Practical Programming: An Introduction to computer Science using Python 3*. P. Gries, J. Campbell & J. Montojo. Edited by Lynn Beighley. 2nd Edition.The Pragmatic Bookshelf. 2013
- *Learning Python with Raspberry Pi*. Alex Bradbury, Ben Everard. Wiley. 2014
- Tutorial: *Python 3 para impacientes*. Antonio Suárez Jiménez. 2014

http://python-para-impacientes.blogspot.com.es/p/indice.html

- *Raspberry Pi, User Guide*. Eben Upton, Gareth Halfacree. Wiley. 2012

Aviso

Esta colección de transparencias se distribuye con fines meramente docentes.

Todas las marcas comerciales y nombres propios de sistemas operativos, programas, hardware, etc. que aparecen en el texto son marcas registradas propiedad de sus respectivas compañías u organizaciones.