Programación con

python™

Tema 2: Conceptos básicos de Python

Alma Mallo Casdelo - alma.mallo@udc.es

Tipos de datos

- Los tipos de datos básicos en Python son:
  - *bool:* Se usa para representar valores lógicos.
  - *int:* Se usa para representar enteros (precisión arbitraria).
  - *float:* Se usa para representar números reales.
  - *complex:* Se usa para representar números complejos.

Tipos de datos

- Strings o cadenas de caracteres: no son tipos de datos tipos de datos simples pero los usaremos de forma sencilla en este primer tema.
- Las cadenas de caracteres (texto) pueden escribirse entre comillas dobles o simples.
- Se pueden escribir cadenas de caracteres de varias líneas utilizando triples comillas.
- Ejemplos de strings: “casa”, “34”, ‘Introduce un número’

Tipos de datos

- type(valor): devuelve el tipo de dato de ese valor.
- Existen funciones de conversión de tipo:
  - De string a entero: int
  - De string a float: float
  - De int o float a string: str

Valores y variables

- Una variable es una etiqueta que referencia a un dato (denominado valor) almacenado en memoria RAM a partir de una determinada posición.

- En Python ese valor tiene un tipo de dato, la variable no.

- Una variable tiene en cada momento el tipo de dato del último valor asignado a ella.

- El tipo de dato de una variable se determina, por tanto, en tiempo de ejecución y puede cambiar de la misma forma que el dato en sí mismo.

Valores y variables

- Notación snake_case: estilo de escritura en el cual las palabras en el nombre de la variable están separadas por el carácter subrayado “_”.
    - Ejemplos: numero_de_alumnos, “diametro_circulo
- Las palabras reservadas del lenguaje como “print”, “input”, “for”, “while”, “break”, “continue”, “True”, “False”, etc. no deben utilizarse como nombres de variable.
- No se pueden utilizar espacios, guiones o números al principio.

Valores y variables

- Las variables pueden cambiar de tipo simplemente con asignarles otro valor de diferente tipo..

```
x = 1
x = "string value"
```

- Se puede asignar un valor a múltiples variables simultáneamente.

```
a = b = c = 1
```

- También se pueden asignar múltiples objetos a múltiples variables al mismo tiempo.

```
a, b, c = 1, 2, "john"
```

Operadores y operandos

- Los operadores son símbolos que representan operaciones (aritméticas o de otro tipo) sobre datos (que llamamos operandos).

- Un operando puede ser:

  - Literal => Valor definido explícitamente en el propio código fuente y que no cambia durante la ejecución (ej: constante numérica, cadena de caracteres literal…)

  - Variable => En Python, puntero a un objeto (referencia a un valor) que sí puede ser modificada.

  - Llamada a función / método => Implica la ejecución del código de dicha función / método y la obtención de un resultado.

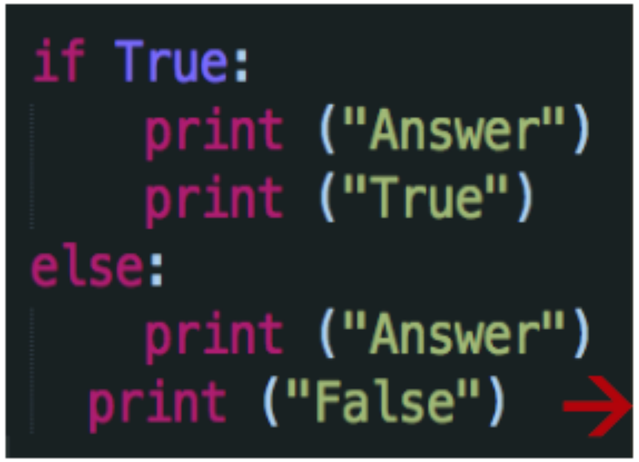
Operaciones numéricas

Principales operadores aritméticos:

Símbolo	Operación	Ejemplos
+	Suma	3 + 2, 2.4 + 1.1
-	Resta	5 - 2, 4.2 - 0.2, 4.2 - 2
*	Multiplicación	5 * 3, 2.5 * 2
/	División	5 / 2 (= 2.5)
//	Cociente división entera	5 // 2 (= 2)
%	Resto división entera	5 % 2 (= 1)
**	Potencia (elevado a un número)	5 ** 2 (= 25), 2.5**2 (= 6.25)

Indentación

- Se utiliza indentación consistente para marcar los bloques.
  - Los elementos del mismo bloque tienen la misma indentación.
  - El bloque empieza en el primer elemento con esa indentación y termina cuando aparece una indentación menor.



Comentarios

- En Python los comentarios comienzan con el carácter #:

```
# La siguiente línea muestra hola
print("hola")
```

- También existe otra forma de comentario, llamado *docstring* y se marca con tres comillas simples o dobles. Se utiliza para documentar funciones, clases, módulos, …

```
"""
Esto es un doctring y puede incluir
varias líneas
"""
print("hola")
```

Entrada / Salida

- `print()` escribe texto en pantalla.
- `print()` admite múltiples argumentos, por defecto los separa por un espacio en blanco y finaliza con un salto de línea.
- Las cadenas de caracteres (texto) pueden escribirse entre comillas dobles o simples.
- Se puede formatear la salida del print. f-srings: cadenas con formato Una cadena con formato va precedida de `f`.
- Ejemplo de print:
  - print(“Hola, esto es un ejemplo”)
  - print(f“El resultado es {resultado}”)

Entrada / Salida

- `input()` lee datos por teclado:
  - Devuelve una cadena de caracteres.
  - Admite un argumento que es el texto que se desea que salga por pantalla para informar al usuario.
- Un cadena de caracteres puede convertirse a entero con la función `int()`, a número real con `float()`, etc.
- Ejemplo: `n1 = int(input("Dime el valor de n1: ")`

Seguimos con ejemplos en el cuaderno Jupyter del tema…