USC
UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

2. PRIMEROS PASOS CON PYTHON

Programación I

Grado en Inteligencia Artificial

Curso 2022/2023

Contenidos

- Entorno de programación
- Operadores aritméticos
- Tipos de errores
- Tipos de datos
- Variables y sentencias de asignación
- Funciones predefinidas
- Módulos estándar
- Métodos

Entorno de programación

- **Editor de texto** para escritura de programas
- **Traductor** (compilador o **intérprete**)
  - Herramienta de **ejecución interactiva**
- Depurador
- Analizador en tiempo de ejecución
- Herramientas para ejecución de pruebas
- Analizador de cobertura
- Generador de documentación
- Analizador estático
- Sistema de Control de Versiones
- …

Instalación de Python

- Web oficial: https://www.python.org/
  - Sección *Downloads*
- Implementación tradicional (**CPython**):
  - Instalar herramienta IDLE
  - Complementar con herramientas como Notepad++ (Windows)
- Implementaciones alternativas
  - Anaconda
  - PythonAnywere (ejecución en la nube)

Entorno Integrado de Desarrollo

- **IDLE** (incluido con CPython)
- **Pycharm** de Jetbrains
- **Visual Studio Code** de Microsoft
- **Eclipse** + extensión **Pydev**
- **Spyder**
- **Anaconda**
- …

Documentación oficial de Python

- Página principal: https://docs.python.org
- Tutorial: https://docs.python.org/3/tutorial/
- Biblioteca estándar: https://docs.python.org/3/library/
- Referencia del lenguaje: https://docs.python.org/3/reference

Operadores aritméticos

Operación	Operador	Cardinal.	Asociativ.	Precedencia
Exponenciación	**	Binario	Derecha	1
Identidad	+	Unitario	–	2
Cambio de signo	-	Unitario	–	2
Multiplicación	*	Binario	Izquierda	3
División	/	Binario	Izquierda	3
Cociente	//	Binario	Izquierda	3
Resto (módulo)	%	Binario	Izquierda	3
Suma	+	Binario	Izquierda	4
Resta	-	Binario	Izquierda	4

Tipos de errores

- Errores **de sintaxis**
- Errores **en tiempo de ejecución** (excepciones)
- Errores **de tipo lógico**

Tipos de datos en Python

- **Simples**:
  - Entero: **int**
  - Bit: **bool**
  - Real: **float**
  - Complejo: **complex**
- **Compuestos** (*colecciones*):
  - Cadena de caracteres: **str**
  - Lista: **list**
  - Rango: **range**
  - Tupla: **tuple**
  - Conjunto: **set**
  - Diccionario: **dict**

Tipos numéricos

- Tipo **float** permite manejar *aproximaciones* a números reales (usar . para decimales)
- Se considera a **bool** un tipo de entero (**int**)
- Si en una expresión se usan datos de diferentes tipos, el tipo del resultado siempre es el más *expresivo*
- La función predefinida **type()** permite comprobar el tipo de cada dato
- Los enteros ocupan menos memoria y las operaciones entre ellos son más rápidas

Tipo booleano

- Mínima cantidad de información: *bit*
- El tipo **bool** puede tomar sólo los valores **True** y **False** (constantes predefinidas)
- Operadores lógicos:

Operación	Operador	Cardinalidad	Asociatividad	Precedencia
Negación	not	Unitario	–	Alta
Conjunción	and	Binario	Izquierda	Media
Disyunción	or	Binario	Izquierda	Baja

Operadores de comparación

- Devuelven valores booleanos

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

Resumen de operadores

Operación	Operador	Cardinalidad	Asociatividad	Precedencia	Tipo
Exponenciación	**	Binario	Derecha	1	Aritmético
Identidad	+	Unitario	–	2	Aritmético
Cambio de signo	-	Unitario	–	2	Aritmético
Multiplicación	*	Binario	Izquierda	3	Aritmético
División	/	Binario	Izquierda	3	Aritmético
Cociente	//	Binario	Izquierda	3	Aritmético
Resto (módulo)	%	Binario	Izquierda	3	Aritmético
Suma	+	Binario	Izquierda	4	Aritmético
Resta	-	Binario	Izquierda	4	Aritmético
Igual que	==	Binario	–	5	Comparación
Distinto de	!=	Binario	–	5	Comparación
Menor que	<	Binario	–	5	Comparación
Menor o igual que	<=	Binario	–	5	Comparación
Mayor que	>	Binario	–	5	Comparación
Mayor o igual que	>=	Binario	–	5	Comparación
Negación	not	Unitario	–	6	Lógico
Conjunción	and	Binario	Izquierda	7	Lógico
Disyunción	or	Binario	Izquierda	8	Lógico

Literales de entero

- Aparte de base 10, los enteros se pueden especificar en base 2, 8 y 16:
  - **Binario**: prefijos 0b y 0B
  - **Octal**: prefijos 0o y 0O
  - **Hexadecimal**: prefijos 0x y 0X
- Para proporcionar resultados en alguna de estas bases se necesita conocimiento sobre *strings*

Variables y asignaciones

- Las **variables** son *contenedores* de datos
- Las sentencias de **asignación** permiten introducir un valor en una variable
- En Python **no se declaran variables** de forma explícita, sino que se crean al asignarles un valor por primera vez
- El **operador de asignación** es **=** (no confundir con el operador **==**)

Sentencias de asignación

- Siguen el patrón: **variable = expresión**
- **No reversibles**: en el *lado izquierdo* sólo puede aparecer una variable, mientras que en el *lado derecho* puede figurar una expresión de complejidad arbitraria
- Se evalúa la expresión presente en el lado derecho y el resultado se almacena en la variable del lado izquierdo

Sentencias de asignación

- Se puede **reasignar** el valor de una variable cuantas veces se necesite
- Se admiten **asignaciones múltiples**
- Asignaciones **con operador**: +=, -=, *=, /=, //=, %=, **=
- No se puede hacer uso de variables que antes no hayan sido **inicializadas** por medio de una sentencia de asignación

Cadenas de caracteres (*strings*)

- Secuencias de caracteres encerradas entre **comillas simples** o dobles

  'ejemplo'

  'con espacios'

  "usando comillas dobles"

  '123' (ojo, esto no es un número)
- Operadores básicos:
  - **Concatenación**: +
  - **Repetición**: *

Funciones predefinidas

- Se pueden **usar directamente** sin necesidad de *importar* ningún *módulo*
- Reciben *uno o más* **parámetros** (*argumentos*) y devuelven *un* **resultado** (*valor de retorno*)
- Los argumentos se colocan entre paréntesis
- Python proporciona una amplia variedad de funciones muy útiles que operan sobre números o cadenas de caracteres

Funciones *sobre* números

- Valor absoluto: **abs()**
- Redondeo: **round()**
- Conversión de tipo: **int()**, **float()**, **str()**
- Representación en otras bases: **bin()**, **oct()**, **hex()**
- División entera: **divmod()**
- Potencia: **pow()**

Funciones para texto

- Las cadenas también son comparables en base al **orden lexicográfico**
- Valor numérico de un carácter: **ord()**
- Carácter que corresponde a un nº: **chr()**
- Funciones **int()** y **float()** también se pueden alimentar con una cadena
- Funciones de E/S: **print()** e **input()**
- Longitud de un string: **len()**

Módulos estándar

- Proporcionan funciones y variables a través del mecanismo de **importación**:

  from módulo **import** elemento1, elemento2, …

  from módulo **import** *

  import módulo

- Algunos módulos de interés:
  - Matemáticas: **math**, **cmath**, **random**, **statistics**
  - Fecha y hora: **datetime**, **calendar**
  - Sistema: **sys**
  - Persistencia: **csv**, **pickle**, **sqlite3**
  - Internet: **http**, **ftplib**, **poplib**, **imaplib**, **smtplib**
  - Interfaz gráfica: **turtle**, **tkinter**

Módulo matemático

- Trigonométricas: **sin()**, **cos()**, **tan()**, **asin()**, **acos()**, **atan()**, **sinh()**, **cosh()**, **tanh()**…
- Exponencial: **exp()**
- Redondeo: **ceil()**, **floor()**
- Logaritmo: **log()**, **log2()**, **log10()**
- Raíz cuadrada: **sqrt()**
- Factorial: **fact()**
- Divisibilidad: **gcd()**, **lcm()**
- Constantes **pi** y **e**

Métodos

- Sintaxis para **invocar funciones**:

  función(arg1, arg2, arg3, …)

- Un método es un **tipo especial** de función:

  arg1.método(arg2, arg3)

- La petición recae sobre un *objeto* que juega un papel análogo al del sujeto en una oración (el método hace las veces de predicado)

Métodos para manipular cadenas

- Paso a minúsculas: **lower()**
- Paso a mayúsculas: **upper()**
- Iniciales en mayúsculas: **title()**
- Reemplazo de caracteres: **replace()**
- Salida con formato: **format()**
  - Marcas, campos y formatos
- […] Ver documentación de la clase str