



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

Introducción al Lenguaje Python

LAURA SEBASTIÁ

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Contenidos



- 1. Estructura de un programa**
- 2. Uso de entornos de desarrollo**
- 3. Tipos de datos básicos y operadores**
- 4. Entrada y salida por consola**
- 5. Uso y definición de funciones**

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Estructura de un programa

- Un programa de Python es un fichero de texto (normalmente guardado con el juego de caracteres UTF-8) que contiene expresiones y sentencias del lenguaje Python. Esas expresiones y sentencias se consiguen combinando los elementos básicos del lenguaje.
- El lenguaje Python está formado por elementos (tokens) de diferentes tipos:
 - palabras reservadas (keywords)
 - funciones integradas (built-in functions)
 - literales
 - operadores
 - delimitadores
 - identificadores

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Estructura de un programa

- El intérprete de Python no requiere un programa muy estructurado, sin embargo, un programa que diferencia claramente secciones de código distintas tiene mejor legibilidad.

Encabezado	<ul style="list-style-type: none">• Descripción general• Autor(es)• Fecha/versión
Definición de constantes	<ul style="list-style-type: none">• Constantes
Definición de funciones propias	<ul style="list-style-type: none">• Funciones
Bloque principal	<ul style="list-style-type: none">• Entrada de datos
	<ul style="list-style-type: none">• Procesamiento
	<ul style="list-style-type: none">• Entrega de los resultados (salida)

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Ejecución de programas: entornos de desarrollo

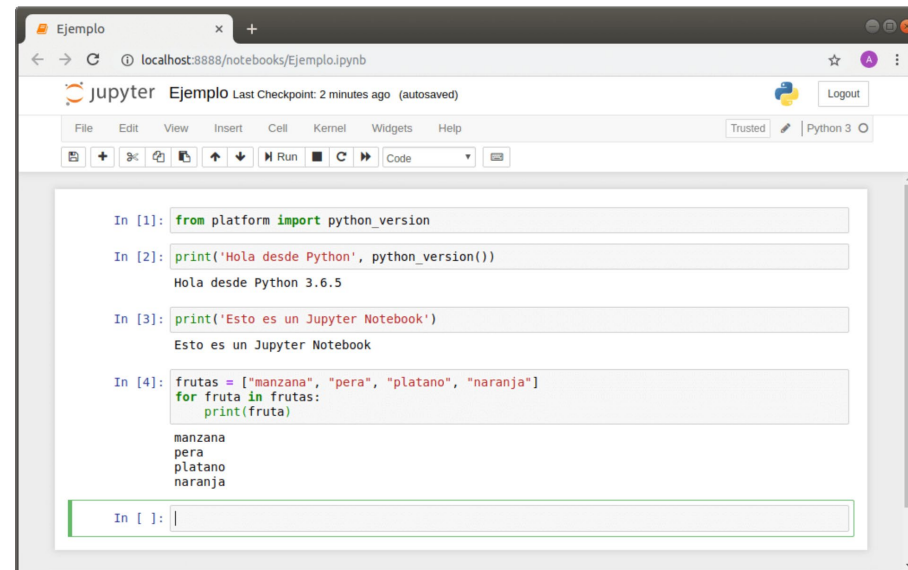
Entornos interactivos basados en web

Jupyter Notebook / Colab

Este entorno de desarrollo es una aplicación web que podemos ejecutar en un navegador como Chrome, Firefox, etc.

Esta aplicación facilita la creación de cuadernos (notebooks) compuestos por celdas. En estas celdas podemos desarrollar nuestro código Python e ir las ejecutando una a una.

Este entorno es muy popular en data science y machine learning, ya que permite visualizar gráficos y tener el código que los ha generado en un mismo documento.

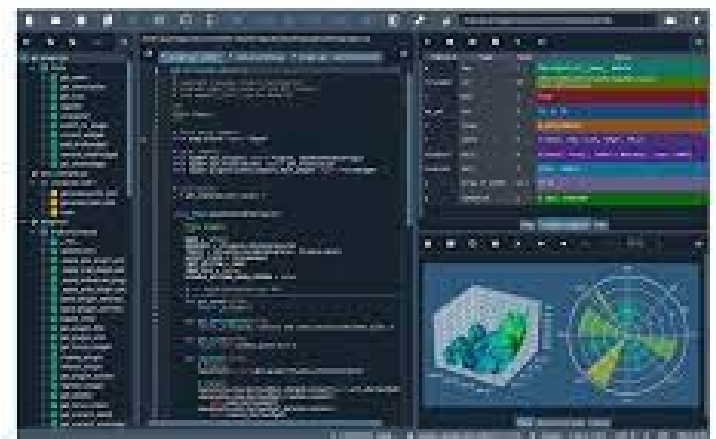


TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Ejecución de programas: entornos de desarrollo

Entornos de desarrollo integrados (IDE)

Los entornos de desarrollo integrado (IDEs en inglés) son un tipo de aplicaciones que están pensadas para facilitar la productividad cuando desarrollamos código, incorporando características como el resaltado de sintaxis. Hay una gran variedad de IDEs disponibles para Python.



TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Ejecución de programas: instalación

WinPython

WinPython is a free open-source portable distribution of the Python programming language for Windows 8/10 and scientific and educational usage.



<https://winpython.github.io/>



TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Ejecución de programas

- Errores:

```
>>> 1+2)
      File "<interactive input>", line 1
        1+2)
          ^
      SyntaxError: invalid syntax
```

La sintaxis de la instrucción es incorrecta

```
>>> 1/0
Traceback (most recent call last):
  File "<interactive input>", line 1, in <module>
ZeroDivisionError: integer division or modulo by zero
```

Error en la ejecución de la instrucción: división por cero

```
>>> b+7
Traceback (most recent call last):
  File "<interactive input>", line 1, in <module>
NameError: name 'b' is not defined
```

Se utiliza una variable 'b' que no está definida

```
>>> abs(3,5)
Traceback (most recent call last):
  File "<interactive input>", line 1, in <module>
TypeError: abs() takes exactly one argument (2 given)
```

La llamada a la función abs es incorrecta, se requiere un único parámetro

ETC...

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Tipos de datos

1. Numéricos:

1. Enteros: int (sólo limitado por la memoria de la máquina)

2. Reales: float (64 bits)

3. Complejos: complex

2. Strings

3. Booleanos: True, False

```
b=True  
c='Esto es un string'  
d="Esto tambien"  
e=2524  
f=1.79  
print (type(b) , type(c) , type(d) , type(e) , type(f) )
```

```
<class 'bool'> <class 'str'> <class 'str'> <class 'int'> <class 'float'>
```

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Asignación

`nombre = <expresión>`

- Crea la variable, si es necesario. Si ya existe, le asigna el nuevo valor
- Las variables no se declaran y no tienen tipo asignado (tipos dinámicos)

```
a=25
print ("Valor:", a, "Tipo:", type(a))
a="Mi primera clase de Python"
print ("Valor:", a, "Tipo:", type(a))
```

```
Valor: 25 Tipo: <type 'int'>
Valor: Mi primera clase de Python Tipo: <type 'str'>
```

```
>> whos
>> b=7
>> whos
>> del a
>> whos
```

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Operadores

Aritméticos	+	Suma
	-	Resta
	*	Multiplicación
	/	División
	%	Módulo
	**	Exponente
	//	División truncada

```
a=34+67
b=7-12
c=2*4
d=2**4
e=25/3
f=25//3
g=25%3
h=25.0/3
i=25.0//3
j=25.0%3
print (a,b,c,d,e,f,g)
print (h,i,j)
```

```
c1=""primera linea
      segunda linea""
print (c1)
a="uno"
b="dos"
c=a+b
d=a*3
print (a,b,c,d)
```

```
101 -5   8   16   8.33333333333334   8   1
8.33333333333334   8.0   1.0
```

```
primera linea
      segunda linea
uno dos unodos unounouno
```

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Operadores

Comparación	==	Igual
	!=	Distinto
	<>	Distinto
	>	Mayor
	<	Menor
	>=	Mayor o igual
	<=	Menor o igual

Se permiten expresiones del tipo $2 < 3 < 4$, que se evalúa como $(2 < 3)$ and $(3 < 4)$
También otras expresiones más “extrañas” como $2 < 3 > 1$ o $2 < 3 == 5$

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Operadores

=	Asignación simple	
+=	Suma y asignación	$c += a$ equivalente $c = c + a$
-=	Resta y asignación	$c -= a$ equivalente $c = c - a$
*=	Multiplicación y asignación	$c *= a$ equivalente $c = c * a$
/=	División y asignación	$c /= a$ equivalente $c = c / a$
%=	Modulo y asignación	$c \% = a$ equivalente $c = c \% a$
**=	Exponente y asignación	$c ** = a$ equivalente $c = c ** a$
//=	División truncada y asignación	$c //= a$ equivalente $c = c // a$

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Operadores

Lógicos	and	Y
	or	O
	not	Negación
Pertenencia	in	Incluido en
	not in	No incluido en
Identidad	is	Apunta al mismo objeto
	is not	No apunta al mismo objeto

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Resumen de operadores

Operación	Operador	Aridad	Asociatividad	Precedencia
Exponenciación	**	Binario	Por la derecha	1
Identidad	+	Unario	-	2
Cambio de signo	-	Unario	-	2
Multiplicación	*	Binario	Por la izquierda	3
División	/	Binario	Por la izquierda	3
Módulo (o resto)	%	Binario	Por la izquierda	3
Suma	+	Binario	Por la izquierda	4
Resta	-	Binario	Por la izquierda	4
Igual que	==	Binario	-	5
Distinto de	!=	Binario	-	5
Menor que	<	Binario	-	5
Menor o igual que	<=	Binario	-	5
Mayor que	>	Binario	-	5
Mayor o igual que	>=	Binario	-	5
Negación	not	Unario	-	6
Conjunción	and	Binario	Por la izquierda	7
Disyunción	or	Binario	Por la izquierda	8

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Salida de datos

```
print (<expresión>)
```

```
print (<lista de expresiones separadas por comas>)
```

- Muestra por pantalla la lista de expresiones, que pueden incluir expresiones a evaluar o cadenas

```
import math
radio=1
volumen=4.0/3.0 * math.pi * radio ** 3
print ("El volumen de una esfera de radio", radio, "es", volumen)
print ("Volumen de una esfera de radio {0:.4f} = {1:.7f}".format(radio, volumen))
print (f"Volumen de una esfera de radio {radio:.4f} = {volumen:.7f}")
```

El volumen de una esfera de radio 1 es 4.1887902047863905

Volumen de una esfera de radio 1.0000 = 4.1887902

Volumen de una esfera de radio 1.0000 = 4.1887902

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Entrada de datos

```
variable = input('texto opcional a mostrar')
```

→ devuelve un string

- Se muestra el texto entre comillas, se solicita el dato correspondiente y se almacena el valor leído en la variable
- Podemos utilizar input y convertir a otro tipo de dato con las funciones: int(), float(), ...

```
import math
radio=float(input('Introduce el radio de la esfera:'));
volumen=4.0/3.0 * math.pi * radio ** 3
print ("El volumen de una esfera de radio", radio, "es", volumen)
```

El volumen de una esfera de radio 3.0 es 113.097335529

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

(Algunas) funciones predefinidas

(Todas las funciones predefinidas en: <https://docs.python.org/2/library/functions.html>)

Función	Significado	Ejemplo
abs(x)	Valor absoluto de x	abs(-3) → 3
float(x)	Conversión a float	float(3) → 3.0 float('3.2') → 3.2
int(x)	Conversión a int	int(2.1) → 2 int('2') → 2
str(x)	Conversión a cadena	str(2.1) → '2.1'
round(x)	Redondeo al float más próximo cuya parte decimal sea 0	round(2.1) → 2.0 round(-2.9) → -3.0
round(x,y)	Redondeo al float más próximo con y decimales	round(2.1451,2) → 2.15 round(2.1451,3) → 2.145

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Funciones definidas en módulos

`from <módulo> import <lista de funciones separadas por comas>`


- Importa la lista de funciones indicada definidas en “módulo”

```
from math import sin  
print ('Ejemplo seno: ', sin(0))
```

Ejemplo seno: 0.0

`from <módulo> import *`

- Importa todas las funciones definidas en “módulo”
- Tiene el inconveniente de que una función importada puede enmascarar alguna variable definida en el programa



```
sin=3  
from math import *  
print ('Ejemplo seno: ', sin)
```

Ejemplo seno: <built-in function sin>

```
import math  
print ('Ejemplo seno: ', math.sin(0))
```

Ejemplo seno: 0.0

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Métodos

- **Ciertos tipos de datos (de momento, sólo cadenas) permiten invocar funciones especiales: “métodos”**

`variable.método (<argumentos>)`

- **Todos los métodos se encuentran definidos en:**
<https://docs.python.org/3/library/stdtypes.html#string-methods>

Método	Significado	Ejemplo
lower()	Conversión a minúsculas	cadena='Un EJemplo' print(cadena.lower()) → 'un ejemplo'
upper()	Conversión a mayúsculas	cadena='Un EJemplo' print(cadena.upper()) → 'UN EJEMPLO'
title()	Pasa la inicial de cada palabra a mayúsculas	cadena='un EJemplo' print(cadena.title()) → 'Un Ejemplo'
replace(p,r)	Busca <i>p</i> en la cadena y sustituye sus apariciones por <i>r</i>	cadena='Un EJemplo' print(cadena.replace('E','e')) → 'Un eJemplo'

TEMA 1. INTRODUCCIÓN A LA PROGRAMACIÓN

Definición de funciones (básico)

- Una función es un fragmento de código con un nombre asociado que realiza una serie de tareas y devuelve un valor
- Cuando no se especifica un valor de retorno, la función devuelve el valor None

```
def nombreFuncion(param1, ..., param-n) :  
    instrucción-1  
    ...  
    instrucción-n
```

```
def suma(n1, n2):  
    return n1+n2  
  
a=suma(3,6.79)  
b=suma(5.25,4.75)  
c=suma('Buenos','dias')  
print (a)  
print (b)  
print (c)
```

```
9.79  
10  
Buenosdias
```

```
def miFuncion(p1, p2):  
    print (p1)  
    print (p2)  
  
miFuncion(3, 'Laura')  
a=miFuncion(6,8)  
print (a)
```

```
3  
Laura  
6  
8  
None
```