Capítulo 2

Primeros programas

Es muy habitual en el mundo de la programación, comenzar escribiendo un programa que visualice un mensaje por pantalla y de esta forma muestre como se hace una de las tareas más simples y habituales, la de mostrar información.

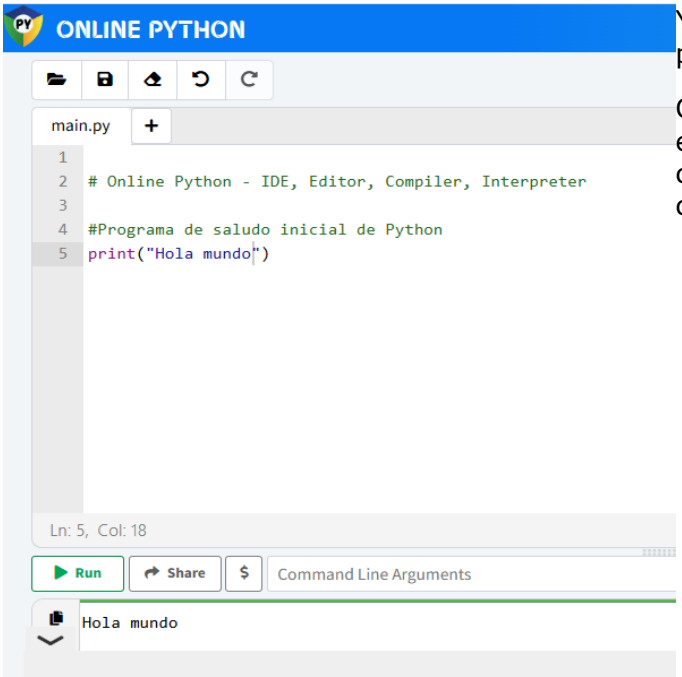
A este programa se le suele llamar ¡Hola Mundo!, que viene a representar un saludo a este nuevo lenguaje de programación.

A partir de este momento y salvo que digamos lo contrario, nuestras instrucciones las escribiremos en el editor de código, y así comenzaremos a escribir pequeños programas que ayudarán a desarrollar la habilidad y la lógica de programación.

En Python, como ya vimos en el capítulo anterior, mostrar el mensaje ¡Hola Mundo! es tan simple como escribir una única instrucción:

```
1      print('¡Hola Mundo!')
```

Simplemente escribiremos la línea de arriba en la zona de código y daremos al botón RUN, con lo que obtendremos la siguiente imagen:



Ya hemos conseguido hacer nuestro primer programa.

Como comentamos antes, podremos guardar el código en el equipo usando el icono correspondiente a guardar o compartir el código mediante una URL (Share)

En ventana de nuestro intérprete, que se denomina **Shell**, estamos viendo el resultado de la ejecución de nuestro programa. Es normal, que esta ventana se llene pronto de contenido, mensajes de error, pruebas, etc. En cualquier momento puedes limpiarla pulsando el botón correspondiente (Clear). También tenemos esta opción en la ventana del código (Reset Code)

Programa: saludo.py

Como ya sabemos como escribir un programa y también como mostrar información, vamos a seguir avanzando y nuestro próximo programa hará lo siguiente:

> 1º Mostrará un mensaje indicando que se trata de un programa para elaborar un saludo.
>
> 2º Solicitará el nombre de la persona que está usando el programa.
>
> 3º Mostrará un nuevo mensaje saludando a la persona con el nombre que haya escrito.

Para conseguir hacer este programa vamos a usar 2 conceptos expuestos en el capítulo anterior: la función input para introducir información al programa y el uso de una variable para almacenar la información introducida.

```
1     print('Este programa te saluda con tu nombre')
2     nombre = input('Escribe tu nombre: ')
3     print('Hola', nombre, 'encantado de conocerte.')
```

Aunque se trata de un programa muy simple, es importante que se comprenda perfectamente cada una de sus instrucciones:

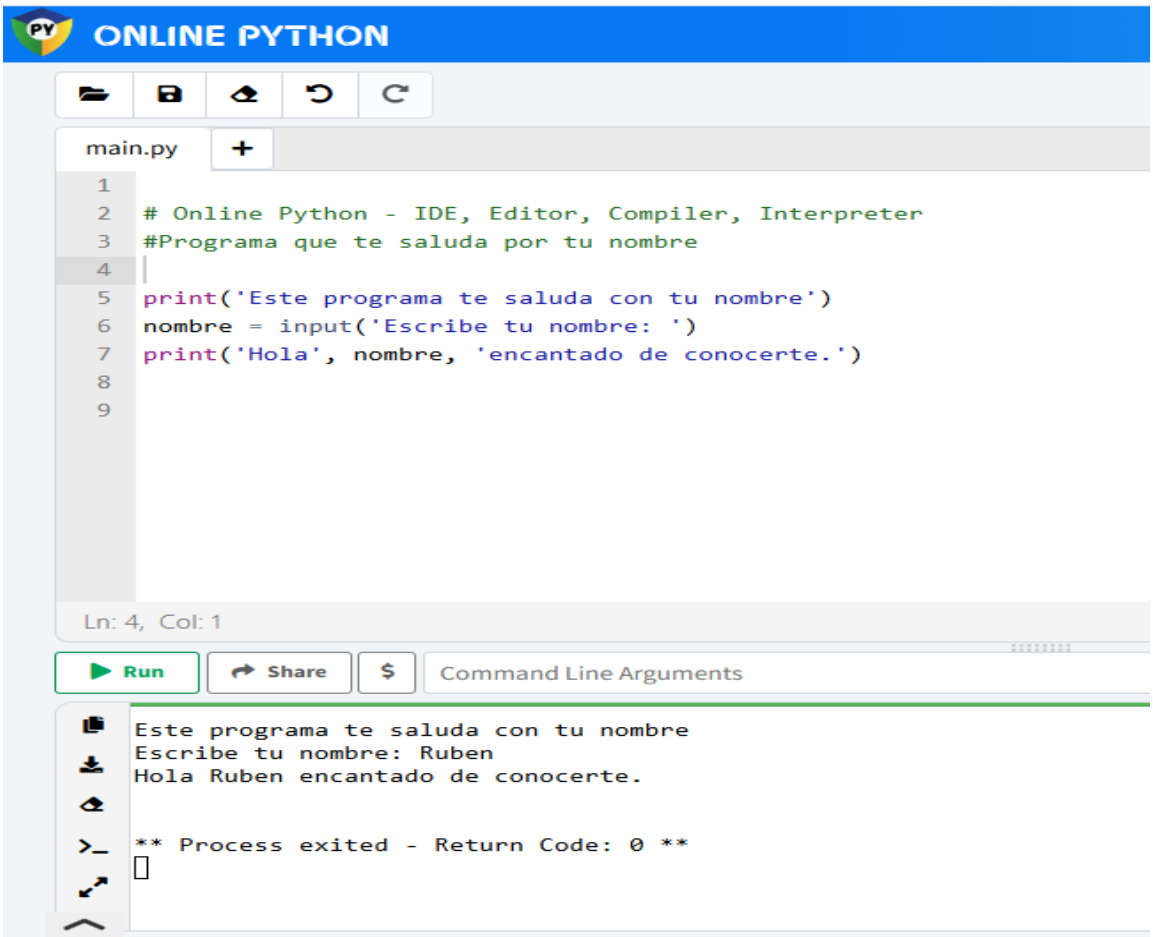
La línea 1 muestra un mensaje de texto, en este caso el uso de la función print no tiene mayor explicación, se trata de la funcionalidad habitual que tiene.

La línea 2 muestra un mensaje mediante la función input y queda a la espera de que el usuario introduzca alguna información y finalice con la pulsación de la tecla Intro (o Enter). Una vez que el usuario ha escrito su nombre, en este caso, la información introducida queda guardada en la variable **nombre** en formato **cadena de texto**.

La línea 3 vuelve a mostrar un mensaje, pero en esta ocasión a la función print se le han pasado varios parámetros separados por comas.

La función print cuando recibe varios parámetros separados por comas, los muestra introduciendo un espacio de separación entre los textos mostrados.

Nuestro programa generará una salida similar a la siguiente imagen:



Es importante resaltar que la información introducida quedará guardada en la variable **nombre** en **formato texto**, incluso si se introduce un valor numérico.

¿Qué ocurre si quiero tratar la información introducida como un valor numérico?

Si leíste atentamente el capítulo 1, ya tendrás la respuesta.

Disponemos de varias funciones para convertir texto en su correspondiente valor numérico:

int	Interpreta el texto introducido como un valor numérico **entero.**
float	Interpreta el texto introducido como un valor numérico **decimal.**
eval	Evalúa matemáticamente el texto introducido. **Es la forma más versátil.**

Ten en cuenta que si usas estas funciones y la expresión introducida no se puede convertir a dicho tipo numérico, el programa generará un error. Más adelante veremos cómo podemos capturar los errores para mostrar algún mensaje cuando se produzcan y que el programa no se interrumpa.

Como práctica del asunto que estamos tratando, vamos a hacer un sencillo programa donde apliquemos lo explicado en los últimos párrafos.

Programa: sumar2numeros.py

Queremos hacer un programa que nos pida 2 números y nos muestre la suma de dichos números.

1º El programa mostrará un mensaje indicando para que sirve.

2º Solicitará el primero de los números.

3º Solicitará el segundo de los números.

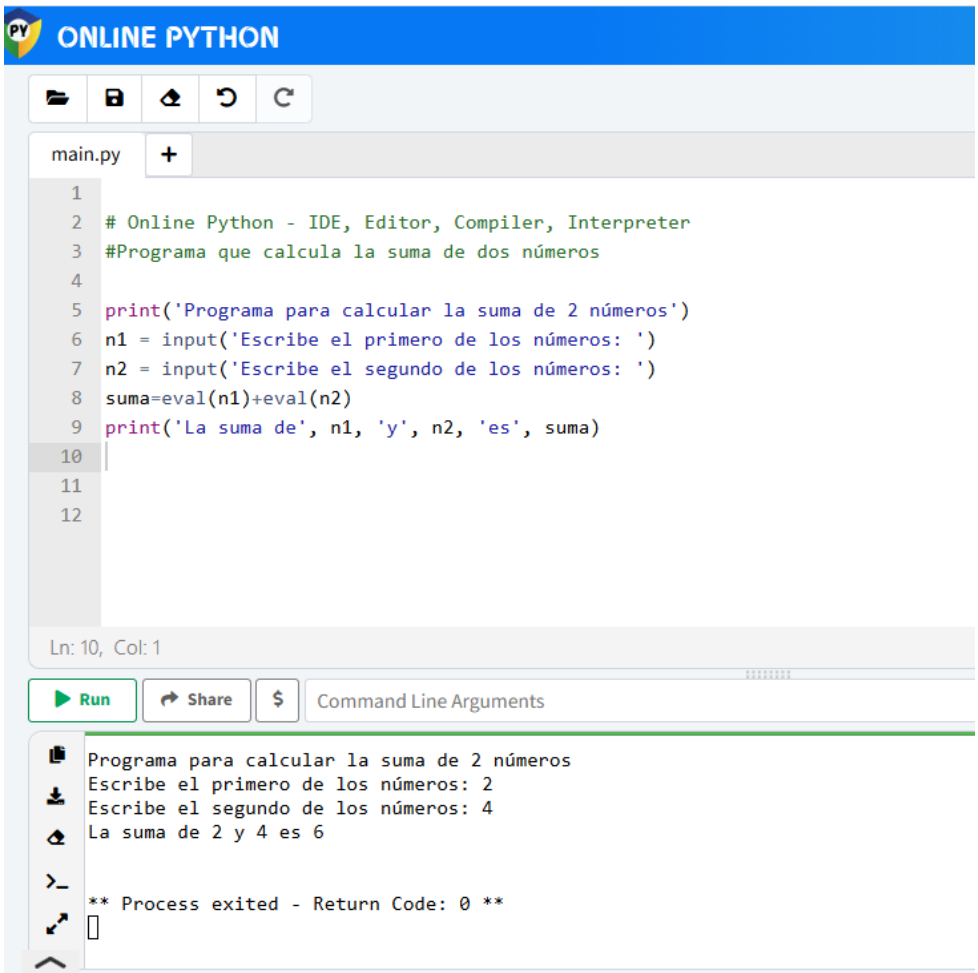
4º Mostrará los 2 números introducidos y el resultado de la suma.

Como es habitual en todos los programas, se desarrolla un bloque que interactúa con el usuario en donde se muestra información y se solicita información. Esto ya sabemos hacerlo con las funciones **print** e **input**, por lo que ya no prestaré mucha más atención a dichas funciones salvo que sea necesario aclarar algún aspecto.

Veamos el código de nuestro programa, más bien una de las formas de hacerlo. Ten en cuenta que generalmente siempre existirán varias formas de realizar un programa que cubra nuestro objetivo:

```
1    print('Programa para calcular la suma de 2 números')
2    n1 = input('Escribe el primero de los números: ')
3    n2 = input('Escribe el segundo de los números: ')
4    suma=eval(n1)+eval(n2)
5    print('La suma de', n1, 'y', n2, 'es', suma)
```

Al ejecutar nuestro programa, obtendremos una salida similar a esta:



De nuevo y aunque el programa sea simple, es conveniente que todos los conceptos queden

totalmente claros puesto que estamos estableciendo los cimientos para el futuro.

En la líneas 2 y 3, hemos solicitado al usuario que introduzca 2 números y se han almacenado en las variables n1 y n2, pero se han almacenado como cadenas de texto. Si en nuestro caso '32' y '47', concatenamos dichas cadenas, obtendríamos '3247'.

En la línea 4 hemos realizado la operación indicándole que lo haga de la siguiente forma:

Evalúa como un número la cadena de texto que contenga **n1**, evalúa también la cadena que contenga **n2** y suma el resultado que te hayan dado. Además, asigna el resultado de la suma a la variable **suma**.

En la línea 5 únicamente hemos construido una forma de mostrar el resultado, podríamos haberlo hecho también así: **print(n1,'+',n2,'=',suma)**

Aunque hemos comprobado que nuestro programa funciona correctamente, vamos a ver otras posibles maneras de resolverlo.

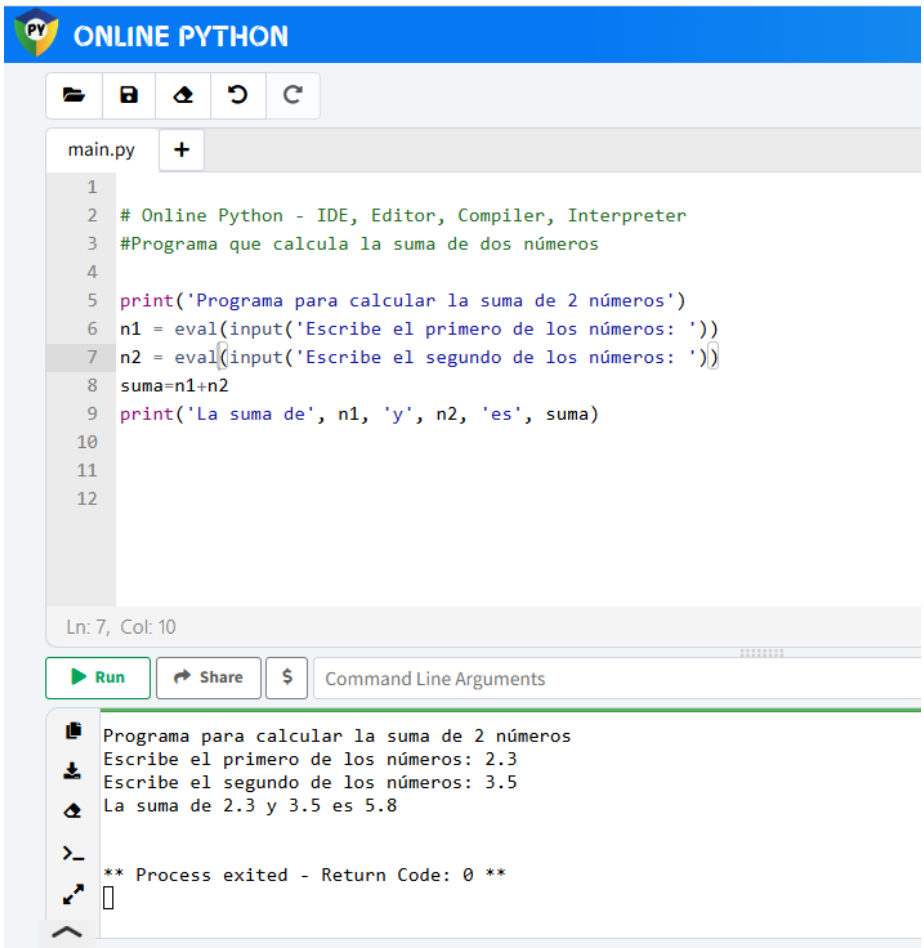
Las líneas 2 y 3 podríamos hacerlas escrito así:

```
2      n1 = eval(input('Escribe el primero de los números: '))
3      n2 = eval(input('Escribe el segundo de los números: '))
```

En esta ocasión le estamos diciendo que al mismo tiempo de solicitar los números, el texto introducido sea evaluado matemáticamente y asignado a las correspondientes variables.

De haberlo hecho así, las línea 4 tendría que haber sido:

```
4      suma = n1 + n2
```



Vamos a ver también una última forma de resolverlo:

```
1     print('Programa para calcular la suma de 2 números')
2     n1 = input('Escribe el primero de los números: ')
3     n1 = eval(n1)
4     n2 = input('Escribe el segundo de los números: ')
5     n2 = eval(n2)
6     suma = n1 + n2
7     print(n1,'+',n2,'=',suma)
```

Esta última forma es interesante por lo siguiente:

En la línea 2, introducimos una cadena de texto en la variable n1, por lo que su contenido es de tipo string.

En la línea 3, a n1 le asignamos el resultado de evaluar numéricamente la misma cadena de texto n1, por lo que a partir de este momento el tipo de dato contenido en n1 será un valor numérico.

Observa también que los valores no tienen por qué ser números enteros, pueden ser valores decimales siempre que usemos **el punto (.) como separador decimal**. (Esta es una ventaja de usar **eval**, en vez de **int** o **float**)

Ejercicios:

Ejercicio 1:

Hacer un programa que solicite 2 números y a continuación muestre el resultado de la multiplicación de los números introducidos.

Ejercicio 2:

Hacer un programa que calcule potencias, es decir, solicite la base y el exponente, y después calcule el resultado de la base elevada al exponente.

Ejercicio 3:

Realiza un programa cuya salida sea similar a la que se muestra en la imagen siguiente.

```
Programa para realizar cálculos
Escribe un número: 23
Escribe un 2º número: 6
Operaciones con 23 y 6
23 + 6 = 29
23 - 6 = 17
23 x 6 = 138
23 / 6 = 3.8333333333333335
23 // 6 = 3 (División entera)
23 % 6 = 5 (Resto de la división)
```

Ejercicio 4:

Realiza un programa cuya salida sea similar a la que se muestra en la imagen siguiente.

```
Programa para calcular
el cuadrado y la raíz de un número
Escribe el número para los cálculos: 17
El cuadrado de 17 es 289
La raíz de 17 es 4.123105625617661
```

Ejercicio 5:

Realiza un programa cuya salida sea similar a la que se muestra en la imagen siguiente.

```
Programa que analiza una palabra
Escribe una palabra: Informática
La palabra Informática tiene 11 caracteres
La primera letra es I
La última letra es a
La 2 primeras letras son In
Las 2 últimas letras son ca
```

Ejercicio 6:

Pide la base y la altura de un rectángulo y muestra su área.

Ejercicio 7:

Pide la base y la altura de un triángulo y calcula el área.

Ejercicio 8:

Pide una cantidad de minutos y muéstrala expresada en horas y minutos.

Ejercicio 9:

Pide un número entero y muestra cuál es su última cifra.

Ejercicio 10:

Pide un número de dos cifras y muestra el número invertido.
Ejemplo: 42 → 24.

Ejercicio 11:

Pide dos números e intercambia sus valores (mostrando los resultados).

Ejercicio 12:

Pide un número entero y muestra cuál es la cifra de las decenas. (Pista: Divide 345672 entre 10 y fíjate en los cocientes y los restos…)