2.1. Variables.

En esta lección se tratan aspectos generales de las variables.

¿Qué es una variable?

Variables en Matemáticas:

El concepto de "variable" proviene de las Matemáticas. En Matemáticas, una variable es un símbolo que forma parte de una expresión o de una fórmula. Normalmente las variables se representan mediante letras del alfabeto latino (x, y, z, n, i, j, etc.). Dependiendo del contexto, las variables significan cosas distintas. Por ejemplo:

En el caso del Álgebra, una variable representa una cantidad desconocida que se relaciona con otras y que en algunos casos podemos averiguar.

Consideremos por ejemplo la ecuación:

x + 3 = 5

En este caso, la variable x representa una cantidad desconocida, pero de la que se sabe que si se le suma 3 se obtiene 5. Resolviendo la ecuación, obtenemos inmediatamente que la variable x estaba representando realmente el número 2.

En el caso del Análisis matemático, una variable no representa una cantidad determinada, sino que representa todo un conjunto de valores:

Consideremos por ejemplo la ecuación de la recta:

y = x + 1

En este caso, la variable x no representa ningún valor concreto, sino que puede tomar cualquier valor numérico positivo o negativo. Para cada valor de x podemos calcular el valor correspondiente de la variable y. Si interpretamos x e y como las coordenadas de puntos en un plano y dibujamos varios puntos, podríamos ver que todos los puntos se encuentran situados en una misma recta.

Variables en Programación

En Programación también existe el concepto de "variable", parecido, pero no idéntico al concepto matemático.

En Programación, las variables están asociadas a variables concretos. Además, cada lenguaje de programación tiene su forma de implementar el concepto de variable, por lo que lo que se explica a continuación es válido para muchos lenguajes de programación, aunque otros lenguajes de programación permiten otras posibilidades.

En muchos lenguajes de programación, una variable se puede entender como una especie de caja en la que se puede guardar un valor (por ejemplo, un valor numérico). Esa caja suele corresponder a una posición de memoria en la memoria del ordenador.

Las variables se representan también mediante letras o palabras completas: x, y, a, b, nombre, apellidos, edad, etc.

Cuando en esos lenguajes de programación escribimos la instrucción ...

a = 2

... lo que estamos pidiendo al programa es que guarde el valor 2 en una "caja" y que a la caja le llame a.

En esos lenguajes, el símbolo igualdad (=) hay que entenderlo como una asignación, no como una igualdad matemática. Al escribir una igualdad, le estamos pidiendo al programa que calcule lo que hay a la derecha de la igualdad y que lo guarde en la variable que hay a la izquierda de la igualdad.

En esos lenguajes, no estaría permitido escribir ...

2 = a

... porque 2 no es un nombre válido de variable. Tampoco estaría permitido escribir ...

x + 3 = 5

... porque en el lado izquierdo no pueden aparecer operadores.

Una vez hemos guardado un valor en una variable, podemos hacer referencia a él a lo largo del programa. Por ejemplo, el siguiente programa ...

a = 2

b = 3

c = a + b

guarda el valor 2 en la variable a

guarda el valor 3 en la variable b

La expresión toma los valores guardados en las variables a y b (2 y 3), los suma (2+3) y el resultado (5) lo guarda en la variable c.

La información que se guarda en una variable puede ser de muchos tipos:

Números (enteros, decimales, imaginarios, en notación científica, con precisión arbitraria, en base decimal o en otras bases, etc.),

Cadenas de texto (una sola letra o más letras, del juego de caracteres ASCII occidental o del juego de caracteres Unicode, etc),

Conjuntos de números o texto (matrices, listas, tuplas, etc.)

Estructuras más complicadas (punteros, diccionarios, etc.)

Cada tipo de información se almacena de forma distinta, por lo que existen diferentes tipos de variables para cada tipo de información.

Algunos lenguajes de programación (C, C++, Java, etc) exigen que antes de utilizar una variable se defina el tipo de información que se va a guardar en esa variable. Otros lenguajes de programación (Python, PHP, etc.) no lo exigen y es el intérprete del lenguaje el que decide el tipo de variable a utilizar en el momento que se guarda la información. Los lenguajes que requieren definir los tipos de las variables se denominan lenguajes tipificados y los que no se denominan lenguajes no tipificados. Python es un lenguaje no tipificado, aunque en Python 3.5 se ha introducido la posibilidad de indicar los tipos de datos de los argumentos de las funciones y de los valores devueltos.

2.2.1. Variables en Python

En algunos lenguajes de programación, las variables se pueden entender como "cajas" en las que se guardan los datos, pero en Python las variables son "etiquetas" que permiten hacer referencia a los datos (que se guardan en unas "cajas" llamadas objetos).

Python es un lenguaje de programación orientado a objetos y su modelo de datos también está basado en objetos.

Para cada dato que aparece en un programa, Python crea un objeto que lo contiene. Cada objeto tiene:

- Un identificador único (un número entero, distinto para cada objeto). El identificador permite a Python referirse al objeto sin ambigüedades.
- Un tipo de datos (entero, decimal, cadena de caracteres, etc.). El tipo de datos permite saber a Python qué operaciones pueden hacerse con el dato.
- Un valor (el propio dato).

Así, **las variables en Python no guardan los datos, sino que son simples nombres para poder hacer referencia a esos objetos.**

Si escribimos la instrucción:

a = 2

... **Python: Crea el objeto "2".** Ese objeto tendrá un identificador único que se asigna en el momento de la creación y se conserva a lo largo del programa. En este caso, **el objeto creado**

será de tipo número entero y guardará el valor 2 asocia el nombre “a” al objeto número entero 2 creado.

Así, **al describir la instrucción anterior no habría que decir 'la variable a almacena el número entero 2', sino que habría que decir 'podemos llamar a al objeto número entero 2'. La variable a es como una etiqueta que nos permite hacer referencia al objeto "2", más cómoda de recordar y utilizar que el identificador del objeto.**

Por otro lado, en **Python se distingue entre objetos mutables y objetos inmutables**:

Los objetos inmutables son objetos que no se pueden modificar. Por ejemplo, los números, las cadenas y las tuplas son objetos inmutables

Los objetos mutables son objetos que se pueden modificar. Por ejemplo, las listas y los diccionarios son objetos mutables.

En el caso de los objetos inmutables (como los números) no hay mucha diferencia entre considerar la variable como una caja o como una etiqueta, pero en el caso de los objetos mutables (como las listas) pensar en las variables como cajas puede llevar a error.

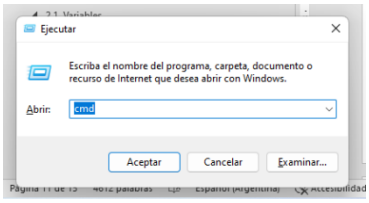
En estas clases utilizaremos expresiones como "guardar un valor en una variable" en vez de "guardar el valor en un objeto y asociar una variable al objeto", pero el alumno debe tener presente lo que ocurre en realidad en Python.

Más adelante profundizaremos en la idea de variable de Python.

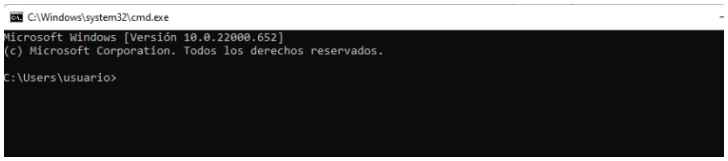
2.2.2. Definir una variable

Las variables en Python se crean cuando se definen por primera vez, es decir, cuando se les asigna un valor por primera vez. Para asignar un valor a una variable se utiliza el operador de igual (=). A la izquierda del igual se escribe el nombre de la variable y a la derecha el valor que se quiere dar a la variable.

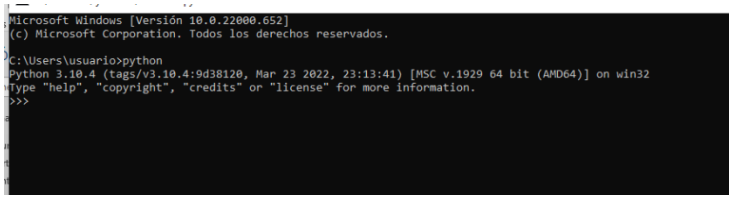
Para comenzar con un ejemplo vamos a abrir el intérprete de Python. Para ello entramos en el procesador de comandos (CMD) de Windows, presionando la tecla de Windows+ la R, tipiamos CMD y damos Enter.



Esto no abrirá una ventana similar a esto:

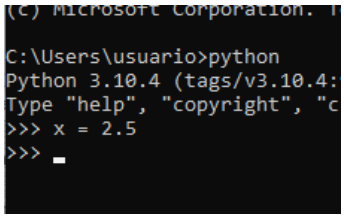


Allí tipiamos Python y damos Enter y nos tiene que aparecer así:

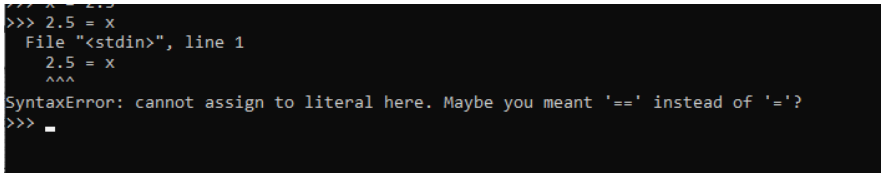


Vean que en la última línea nos aparece >>> esto es signo que estamos dentro del procesador de Python. Aquí podemos tipear comando que Python ejecutar cada vez que demos Enter.

Con el ejemplo siguiente haremos que Python almacene el número decimal 2.5 en una variable de nombre x (como se comenta en el apartado anterior, realmente habría que decir que se crea la etiqueta x para hacer referencia al objeto número decimal 2.5). Fíjese en que los números decimales se escriben con punto (.) y no con coma (,).



La variable se escribe siempre a la izquierda del igual. Si se escribe al revés, Python genera un mensaje de error:



Nos dará el error: SyntaxError: can't assign to literal

Para que IDLE muestre el valor de una variable, basta con escribir su nombre (en este ejemplo ponemos x):

```
>>> x = 2.5
>>> x
2.5
>>>
```

Si una variable no se ha definido previamente, escribir su nombre genera un mensaje de error:

```
>>> x = -10
>>> y
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'y' is not defined
>>>
```

Nos dará el error: Traceback (most recent call last):

File "<pyshell#1>", line 1, in <module>

y

NameError: name 'y' is not defined

Una variable puede almacenar números, texto o estructuras más complicadas (que se verán más adelante). Si se va a almacenar texto, el texto debe escribirse entre comillas simples (') o dobles ("), que son equivalentes. A las variables que almacenan texto se les suele llamar cadenas (de texto).

```
>>> nombre = "Juan Manuel"
>>> nombre
'Juan Manuel'
>>>
```

Si no se escriben comillas, Python supone que estamos haciendo referencia a otra variable (que, si no está definida, genera un mensaje de error):

```
>>> nombre = Juan
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'Juan' is not defined
>>> nombre
'Juan Manuel'
>>>
```

Nos dará el error: NoTraceback (most recent call last):

File "<pyshell#0>", line 1, in <module>

nombre = Pepe

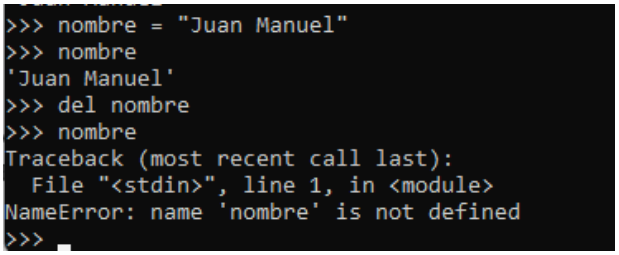
NameError: name 'Juan' is not defined

>>> nombre = Juan Manuel

SyntaxError: invalid syntax

2.2.3. Borrar una variable

La instrucción “del” borra completamente una variable.



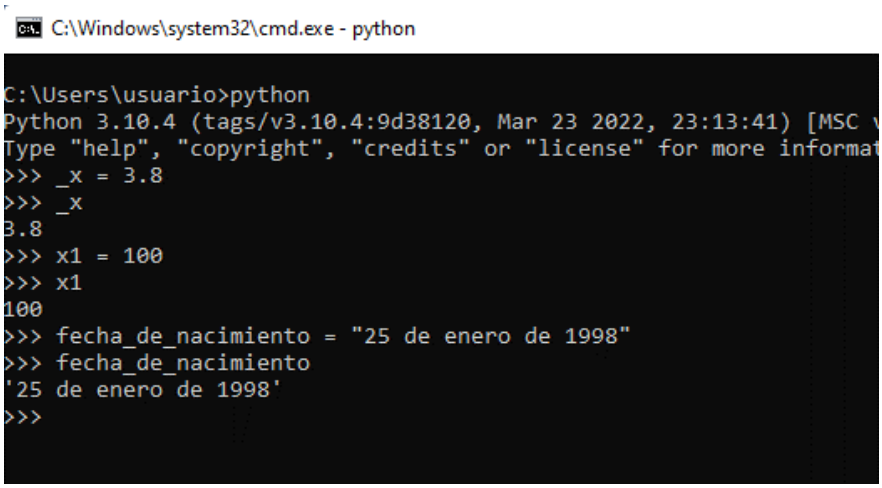
En la ultimas líneas vemos que nos da error al consultar por la variable nombre porque en la línea anterior la borramos con la instrucción “del”.

2.2.4. Nombres de variables

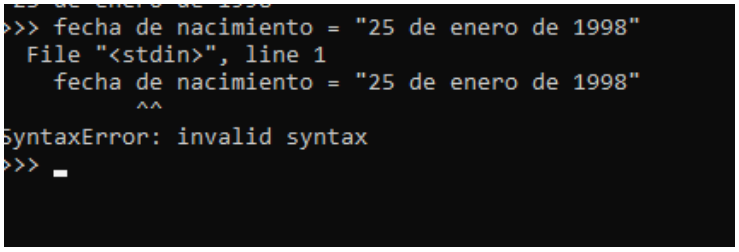
Aunque no es obligatorio, se recomienda que el nombre de la variable esté relacionado con la información que se almacena en ella, para que sea más fácil entender el programa. Si el programa es trivial o mientras se está escribiendo un programa, esto no parece muy importante, pero si se consulta un programa escrito por otra persona o escrito por uno mismo, pero hace tiempo, resultará mucho más fácil entender el programa si los nombres están bien elegidos.

También se acostumbra a utilizar determinados nombres de variables en algunas ocasiones, como se verá más adelante, pero esto tampoco es obligatorio.

El nombre de una variable debe empezar por una letra o por un guion bajo (_) y puede seguir con más letras, números o guiones bajos.



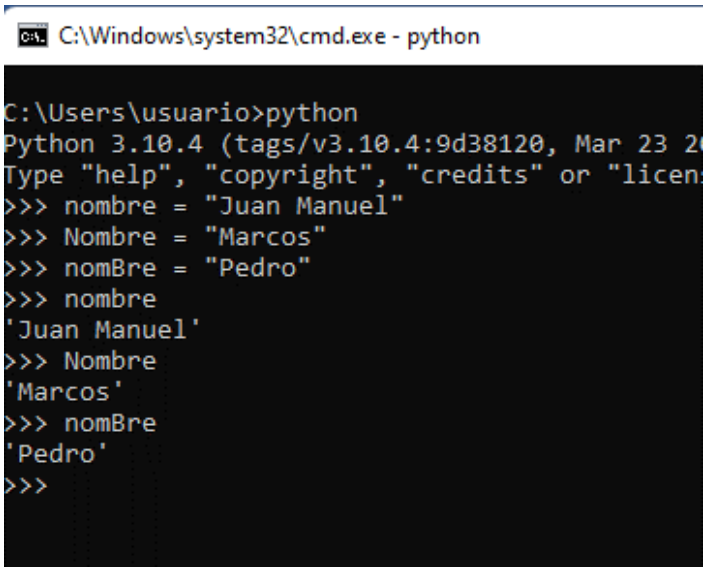
Los nombres de variables no pueden incluir espacios en blanco.



En Python los nombres de variables pueden contener cualquier carácter alfabético (los del alfabeto inglés, pero también ñ, ç o vocales acentuadas), aunque se recomienda utilizar únicamente los caracteres del alfabeto inglés.

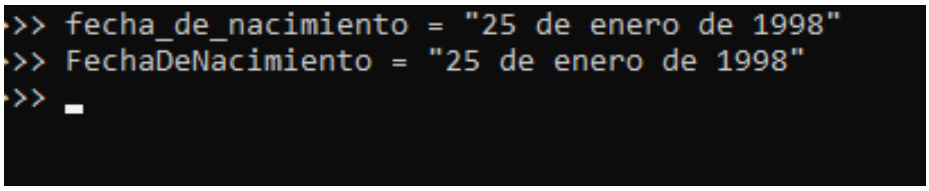
Nota: En Python 2 los nombres de variables no podían contener caracteres no ingleses.

Los nombres de las variables pueden contener mayúsculas, pero tenga en cuenta que Python distingue entre mayúsculas y minúsculas (en inglés se dice que Python es case-sensitive).

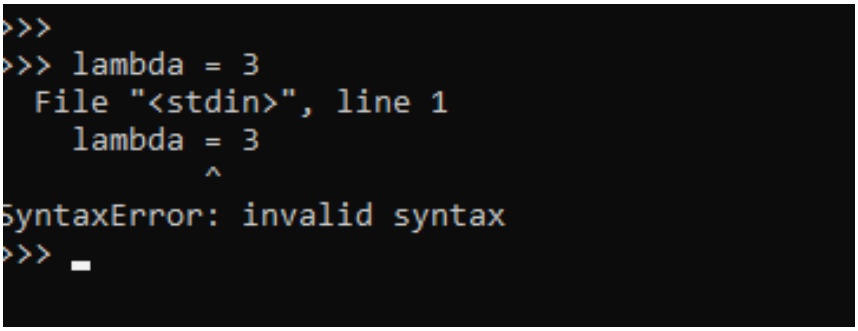


Cuando el nombre de una variable contiene varias palabras, se aconseja separarlas con guiones bajos para facilitar la legibilidad, aunque también se utiliza la notación CamelCase (*), en las que las palabras no se separan, pero empiezan con mayúsculas (salvo la primera palabra).

(*CamelCase es un estilo de escritura que se aplica a frases o palabras compuestas. El nombre se debe a que las mayúsculas a lo largo de una palabra en CamelCase se asemejan a las jorobas de un camello. El nombre CamelCase se podría traducir como Mayúsculas/Minúsculas Camello.)



Las palabras reservadas del lenguaje están prohibidas como nombres de variables:



Los nombres de las funciones integradas sí que se pueden utilizar como nombres de variables, pero más vale no hacerlo porque a continuación ya no se puede utilizar la función como tal:

```
>>>
>>> print = 3
>>> print("hola")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'int' object is not callable
>>>
```

Borrando con “del” la variable con nombre de función, se recupera la función.

```
>>>
>>> print = 3
>>> print("hola")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'int' object is not callable
>>> del print
>>> print("hola")
hola
>>>
```