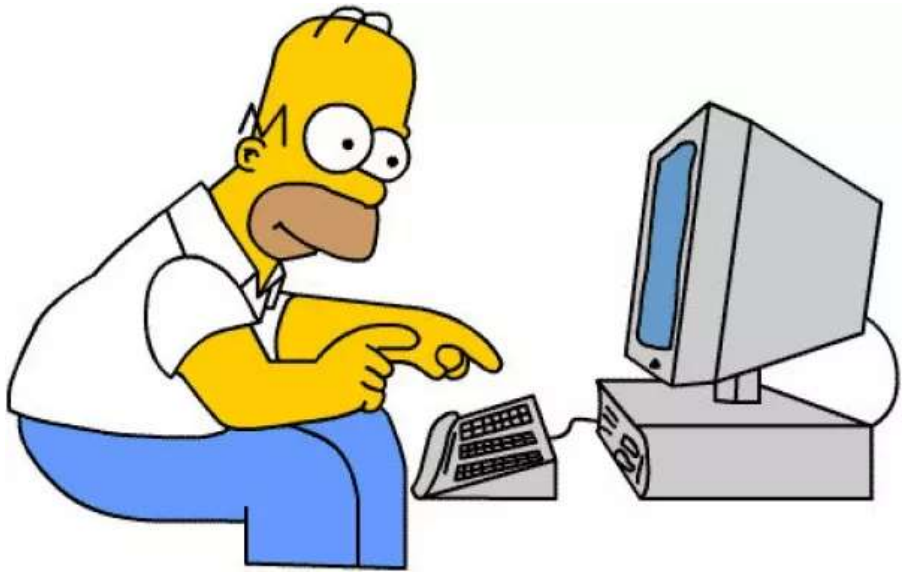
INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

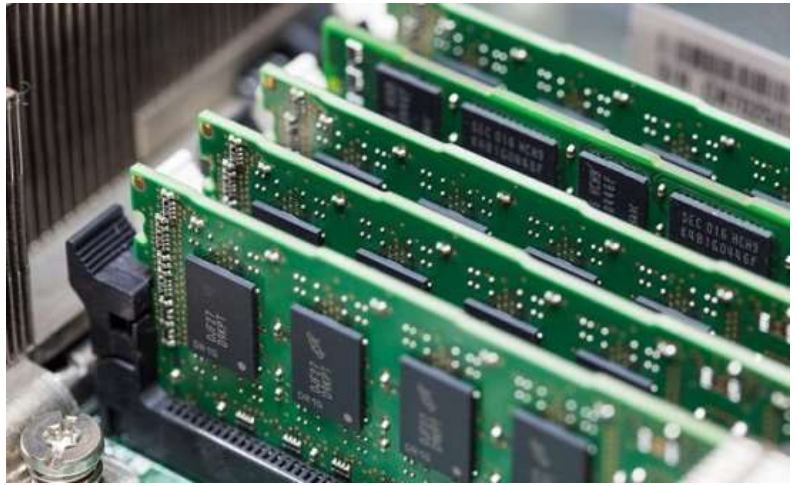


INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.



¿Qué es la memoria RAM ?

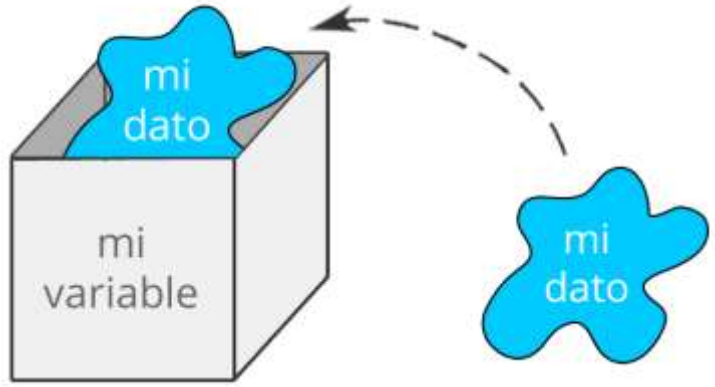
La memoria de la computadora donde se almacenan temporalmente tanto los datos como los programas que está utilizando (ejecutando) en ese momento.

El almacenamiento es considerado temporal por que los datos y programas permanecen en ella mientras el ordenador este encendido y el programa en ejecución. Al apagarse el ordenador los datos que hay en ella se pierden.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

¿Qué es una variable?



En programación, una variable es un espacio reservado en la memoria de nuestro ordenador para almacenar un dato que puede ser usado o modificado tantas veces como se desee.

Las variables deben tener un nombre asociado al dato que almacenan. Es decir, podemos tener una variable llamada Nombre para almacenar el nombre (dato) de una persona, una variable llamada Edad para almacenar la edad (dato), etc.

Las variables pueden contener diferentes datos, lo que se conoce como **tipo de dato**.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

¿Qué es un tipo de dato?

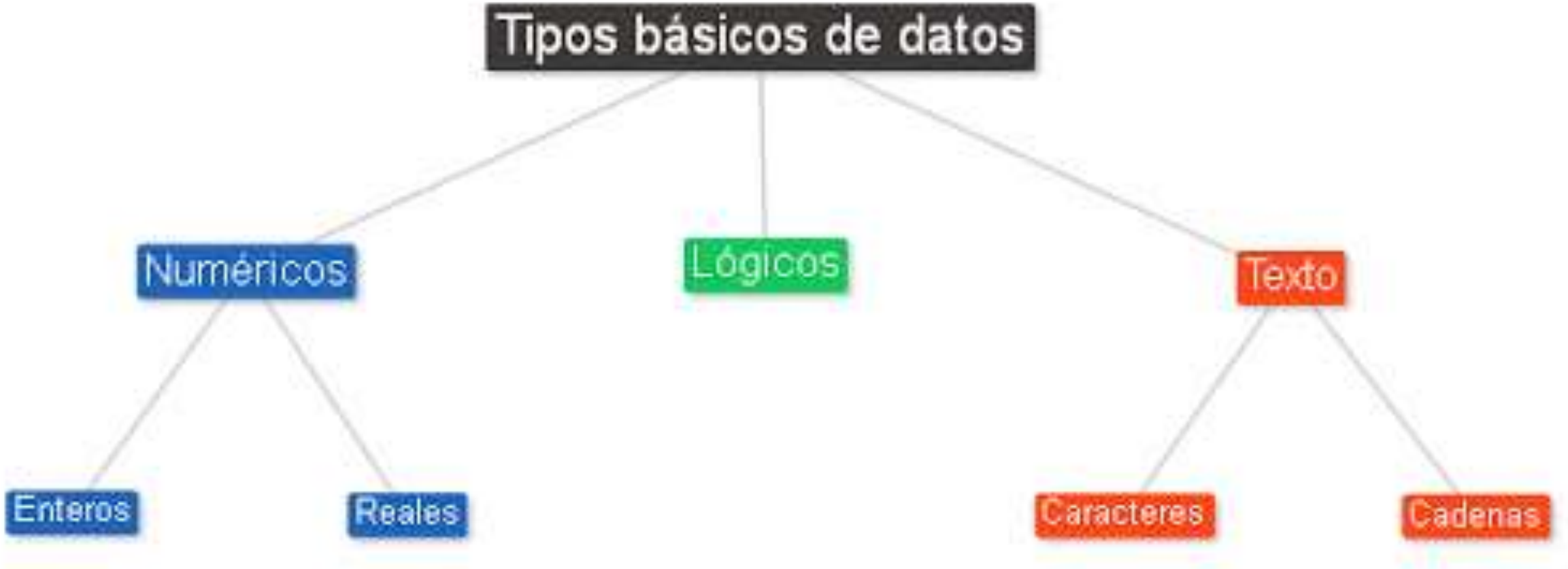
En programación, un tipo de dato es el atributo que especifica al ordenador la clase de dato que tiene que manejar, para saber qué valores puede tomar y qué operaciones realizar.

Los tipos de datos primitivos o elementales más comunes en los lenguajes de programación son los siguientes:

- ✓ Números enteros (y con signo negativo).
- ✓ Números en coma flotante (decimales).
- ✓ Caracteres (alfanuméricos).
- ✓ Cadenas de caracteres (textos).
- ✓ Estados lógicos (booleanos).

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.



INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Los **tipos de datos fundamentales** existentes en Python son los siguientes:

- **Entero** (*int*): Número sin decimales, tanto positivo como negativo, incluyendo el 0.
- **Real** (*float*): Número con decimales, tanto positivo como negativo, incluyendo el 0.
- **Booleanos** (*bool*): Pueden tener dos valores: True o False.
- **Cadenas de texto** (*str*): Texto.
- **Complejo** (*complex*): Número con parte imaginaria.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Identificador de una variable.

Toda variable tiene que tener un **nombre** o identificador que hace referencia al dato almacenado en la memoria del ordenador.

El nombre de la variable estará compuesto por caracteres alfanuméricos sin tildes, carácter subrayado, o números.

Lo habitual es que haya una serie de restricciones:

- ✓ No puede tener espacios.
- ✓ No puede empezar con un número.
- ✓ No puede ser una palabra reservada.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Identificador de una variable.

Otra cosa a tener en cuenta es que hay lenguajes que distinguen mayúsculas y minúsculas (**case sensitive**). Por ejemplo, la variable **total** serían una variable diferente de **Total** en **Python**. Pero en Visual Basic, no.

Nombres válidos:

nombre
Apellido
apellido2
fecha_de_nacimiento
FechaDeNacimiento
_dia

Nombres inválidos:

el nombre
2apellido
fecha-de-nacimiento
programación

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Como se declara una variable en programación.

La mayoría de lenguajes de programación obliga a declarar la variable (decir como se va a llamar) con el tipo de datos asociado.

Por ejemplo, en C++

int total;

En este caso, la variable se llama total, y sabemos que su tipo de datos es un número, al poner int. Esta variable no puede almacenar otro tipo de datos.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Como se declara una variable en programación.

Por ejemplo, para almacenar un número podríamos poner:
total = 10;
A esto se le llama **asignación de valores a una variable**.

También se podría poner todo en una sola línea:
int total = 10;
Aquí tenemos el tipo de datos, la declaración y un valor inicial, todo condensado.
En muchos otros lenguajes de programación no es necesario declarar el tipo de datos.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Como se declara una variable en programación.

Python, es un lenguaje de programación de **tipado débil**, lo que significa que cualquier variable puede contener **cualquier tipo de dato** en cualquier momento, sin especificarlo en la creación de la variable. Esto lo hace un lenguaje de programación más sencillo en los comienzos del aprendizaje de la programación.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Como se declara una variable en programación.

En **Python**, una variable se define con la sintaxis:

```
nombre_de_la_variable = valor_de_la_variable
```

Cada variable, tiene un **nombre** y un **valor**, el cual define a la vez, el **tipo de datos** de la variable.

Las variables en Python no se declaran. las variables se crean cuando se les da valor por primera vez.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

```
texto = "Hola Mundo"
numero = 13
decimal = 3.14
booleano = True
```

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Constantes

Existe un tipo de variable, denominada **constante**, la cual se utiliza para definir valores fijos, que no requieran ser modificados.

MI_CONSTANTE = 12

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Las **palabras reservadas de Python** son las que forman el núcleo del lenguaje Python. Estas palabras no pueden utilizarse para nombrar otros elementos (variables, funciones, etc.), aunque pueden aparecer en cadenas de texto.

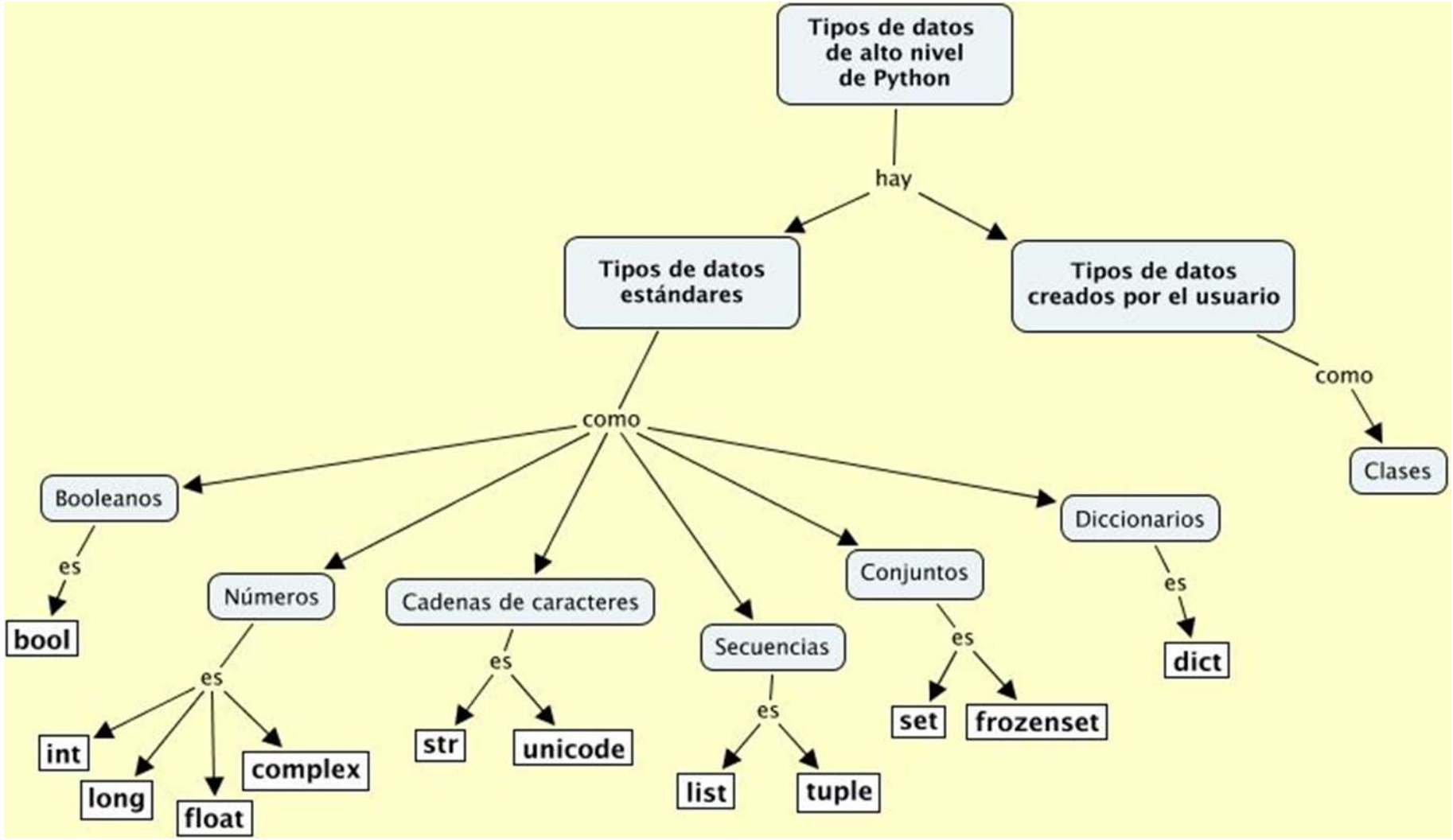
Palabras reservadas de Python 3

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

Y con Python 3.7:
async
await

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.



INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Enteros

Los números enteros son aquellos números positivos o negativos que no tienen decimales (además del cero). En Python se pueden representar mediante el tipo **int** (de integer, entero) o el tipo **long** (largo).

La única diferencia es que el tipo **long** permite almacenar números más grandes. Es aconsejable no utilizar el tipo long a menos que sea necesario, para no malgastar memoria.

edad = 35

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Reales

Los números reales son los que tienen decimales. En Python se expresan mediante el tipo **float**.

precio = 7435.28

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Complejos

Los números complejos, llamado **complex** en Python, son aquellos que tienen parte real y parte imaginaria.

La mayor parte de lenguajes de programación carecen de este tipo, aunque sea muy utilizado por ingenieros y científicos en general.

Los números complejos en Python se representan de la siguiente forma:

complejo = 2.1 + 7.8j

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Operadores aritméticos

Símbolo	Significado	Ejemplo	Resultado
+	Suma	a = 10 + 5	a es 15
-	Resta	a = 12 - 7	a es 5
-	Negación	a = -5	a es -5
*	Multiplicación	a = 7 * 5	a es 35
**	Exponente	a = 2 ** 3	a es 8
/	División	a = 12.5 / 2	a es 6.25
//	División entera	a = 12.5 / 2	a es 6.0
%	Módulo	a = 27 % 4	a es 3

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Un ejemplo sencillo con variables y operadores aritméticos:

monto_bruto = 175
tasa_interes = 12
monto_interes = (monto_bruto * tasa_interés) / 100
tasa_bonificacion = 5
importe_bonificacion = (monto_bruto * tasa_bonificación) / 100
monto_neto = (monto_bruto - importe_bonificacion) + monto_interes

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Booleanos

Una variable de tipo booleano sólo puede tener dos valores: **True** (cierto) y **False** (falso). Estos valores son especialmente importantes para las expresiones condicionales y los bucles.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Operadores de comparación

Los operadores de comparación comparan dos valores y se evalúan a un valor Booleano **True** o **False**

Signo del Operador	Nombre del Operador
<	Menor que
>	Mayor que
<=	Menor o igual a
>=	Mayor o igual a
==	Igual a
!=	Diferente a

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Operadores lógicos

De la misma manera que se puede operar entre números mediante las operaciones de suma, resta, etc., también existen tres operadores lógicos para combinar expresiones booleanas: **and** (y), **or** (o) y **not** (no).

Expresión	Significado
`a` **`and`** `b`	El resultado es `True` solamente si `a` es `True` y `b` es `True` de lo contrario el resultado es `False`
`a` **`or`** `b`	El resultado es `True` si `a` es `True` o `b` es `True` de lo contrario el resultado es `False`
`not` `a`	El resultado es `True` si `a` es `False` de lo contrario el resultado es `False`

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

Una cadena de texto es una secuencia inmutable de caracteres que en Python se representa con el tipo de dato **str**.

Las cadenas no son más que texto encerrado entre comillas simples (‘cadena’) o dobles (“cadena”). Las cadenas se deben cerrar con el mismo tipo de comillas usado en la apertura.

```
mi_cadena1 = "Hola Mundo!"

mi_cadena2 = ‘Hola Mundo!’
```

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

También es posible encerrar una cadena entre triples comillas (simples o dobles).

De esta forma podremos escribir el texto en varias líneas, y al imprimir la cadena, se respetarán los saltos de línea que introdujimos sin tener que recurrir al carácter \n, así como las comillas sin tener que escaparlas.

```
mi_cadena_multilinea = """
Esta es una cadena
de varias líneas
"""
```

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

Dentro de las comillas se pueden añadir caracteres especiales escapándolos con \

- **\n** Salto de línea.
- **\t** Tabulación.
- **\”** Comilla doble.
- **\’** Comilla simple.
- **** Barra diagonal invertida.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

Existen cuatro operadores que nos permiten operar con cadenas de texto:

+ Permite concatenar cadenas . Si queremos concatenar cadenas de caracteres junto con otros tipos de datos podemos hacerlo realizando un **casting** de las variables al tipo de dato **str**.

* Nos permite repetir la cadena tantas veces como lo indique el número utilizado como segundo operando.

+= El operador de asignación de suma permite añadir cadenas al final de una cadena de caracteres.

% El operador modulo permite interpolar variables dentro de una cadena

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

Concatenar

```
a = “Hola”
b = “Mundo”
c = a + “ “ + b
print(c) # “Hola Mundo”
```

Multiplicar

```
a = “Hola ”
b = “Mundo”
c = a *2 + b
print(c) # “Hola Hola Mundo”
```

Formatear con %

```
a = 2
b = 3
print('El resultado de %s y %i es: %s' % (a, b, a + b ))
# ‘El resultado de 2 y 3 es 5’
```

Añadir

```
mensaje = 'Hola'
mensaje += ' '
mensaje += 'Mundo'
print(mensaje) # ‘Hola Mundo’
```

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Obtener el largo de una cadena.

Se puede averiguar la longitud de una cadena utilizando **len().**

```
len("programas ") # Devuelve: 10
```

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Acceder a una posición de la cadena

Las distintas posiciones de una cadena se llaman **índices**. Los índices son números enteros.

Se puede acceder a cada uno de las posiciones de una cadena mediante su índice correspondiente, siendo 0 (cero), el índice del primer elemento.

Si se utiliza un número negativo como índice, esto se traduce en que el índice empieza a contar desde el final, hacia la izquierda:

a[-1] es el último carácter de a, a[-2] es el penúltimo carácter de a, [-3] es el antepenúltimo y así sucesivamente.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Acceder a una posición de la cadena

Queremos averiguar cuál es el carácter que está en la posición i-ésima de una cadena. Para ello escribiremos **a[i]** para ver el carácter de la posición i-ésima de la cadena a.

Trataremos de averiguar con qué letra empieza una cadena.

```
a="Veronica"
a[0]    # Devuelve: ’V’
```

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Recorrer todos los caracteres de una cadena

Python nos permite recorrer todos los caracteres de una cadena de manera muy sencilla, usando directamente un ciclo definido:

```
for x in "programas ":
        print (x)
```

```
# Devuelve:
p
r
o
g
r
a
m
a
s
```

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Segmentos de cadenas

Python ofrece también una notación para identificar segmentos de una cadena. Si en lugar de un número escribimos dos números inicio y fin separados por dos puntos (inicio:fin) Python interpretará que queremos una lista que vaya desde la posición inicio a la posición fin, sin incluir este último.

```
a="Veronica"
a[0:2]          # Devuelve: ’Ve’
```

a[0:2] se refiere a la subcadena formada por los caracteres cuyos índices están en el rango [0,2):

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Segmentos de cadenas

Si se omite la posición de inicio, se considera que la porción a extraer comienza en el primer carácter de la cadena, mientras que, si se omite la posición límite, la porción se extrae hasta el final de la cadena:

serie = “Juego de tronos”

serie[:5] # Devuelve: “Juego”

serie[9:] # Devuelve: “tronos”

Si se omiten ambas posiciones, se devuelve la cadena completa. Sería equivalente a escribir el nombre de la variable.

Serie[:] # Devuelve: “Juego de tronos”

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

Las cadenas pueden ser modificadas a través de las funciones:

cadena.capitalize() Convertir a mayúscula la primera letra.
cadena.lower() Convertir una cadena a minúsculas.
cadena.upper() Convertir una cadena a mayúsculas.
cadena.swapcase() Convertir mayúsculas a minúsculas y viceversa.
cadena.title() Convertir una cadena en Formato Título.
cadena.center((longitud[, "caracter de relleno"]) Centrar un texto.
cadena.ljust(longitud[, "caracter de relleno"]) Alinear texto a la izquierda.
cadena.rjust(longitud[, "caracter de relleno"]) Alinear texto a la derecha.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

cadena.zfill(longitud) Rellenar un texto con ceros a la izquierda hasta alcanzar la longitud final indicada.

cadena.count(“subcadena”[, posicion_inicio, posicion_fin]) Contar la cantidad de apariciones de subcadena dentro de cadena.

cadena.find(“subcadena”[, posicion_inicio, posicion_fin]) Buscar una subcadena dentro de una cadena. Devuelve un entero representando la posición donde inicia la subcadena dentro de cadena. Si no la encuentra, retorna -1

cadena.startswith(“subcadena”[, posicion_inicio, posicion_fin]) Saber si una cadena comienza con una subcadena determinada. Devuelve True o False.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

cadena.endswith(“subcadena”[, posicion_inicio, posicion_fin]) Saber si una cadena finaliza con una subcadena determinada. Devuelve True o False.

cadena.isalnum() Saber si una cadena es alfanumérica. Devuelve True o False.

cadena.isalpha() Saber si una cadena es alfabética. Devuelve True o False.

cadena.isdigit() Saber si una cadena es numérica. Devuelve True o False.

cadena.islower() Saber si una cadena contiene solo minúsculas. Devuelve True o False.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

cadena.isupper() Saber si una cadena contiene solo mayúsculas. Devuelve True o False.

cadena.isspace() Saber si una cadena contiene solo espacios en blanco. Devuelve True o False.

cadena.istitle() Saber si una cadena tiene Formato De Título. Devuelve True o False.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Cadenas

cadena.replace(“subcadena a buscar”, “subcadena por la cual reemplazar”) Reemplazar texto en una cadena.
cadena.strip([“caracter”]) Eliminar caracteres a la izquierda y derecha de una cadena.
cadena.lstrip([“caracter”]) Eliminar caracteres a la izquierda de una cadena.
cadena.rstrip([“caracter”]) Eliminar caracteres a la derecha de una cadena.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Conocer el tipo de una variable

type() recibe como parámetro un objeto y te devuelve el tipo del mismo.

isinstance() recibe dos parámetros: un objeto y un tipo. Devuelve True si el objeto es del tipo que se pasa como parámetro y False en caso contrario.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Conversión de tipos (Casting)

str(): Devuelve la representación en cadena de caracteres del objeto que se pasa como parámetro.

int(): Devuelve un int a partir de un número o secuencia de caracteres.

float(): Devuelve un float a partir de un número o secuencia de caracteres.

complex(): Devuelve un complex a partir de un número o secuencia de caracteres.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

La zona del programa en la que una variable puede ser utilizada es lo que se conoce como su ámbito.

El ámbito de una variable es el contexto en el que existe esa variable. Así, una variable existe en dicho ámbito a partir del momento en que se crea y deja de existir cuando desaparece su ámbito. Además, las variables son accesibles solo desde su propio ámbito.

Los principales tipos de ámbitos en Python son dos:

- ❑ Ámbito local.
- ❑ Ámbito global.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

Ámbito local: corresponde con el ámbito de una función, que existe desde que se invoca a una función hasta que termina su ejecución.
En un programa, el ámbito local corresponde con las líneas de código de una función. Dicho ámbito se crea cada vez que se invoca a la función. Cada función tiene su ámbito local. No se puede acceder a las variables de una función desde fuera de esa función o desde otra función.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

Ámbito global: corresponde con el ámbito que existe desde el comienzo de la ejecución de un programa.

Todas las variables definidas fuera de cualquier función corresponden al ámbito global, que es accesible desde cualquier punto del programa, incluidas las funciones. Si desde un módulo A importamos un módulo B mediante un **import**, desde A podremos acceder a las variables globales de B.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

En Python las **variables locales** son aquellas definidas dentro de una función. Solamente son accesibles desde la propia función y dejan de existir cuando esta termina su ejecución. Los parámetros de una función también son considerados como variables locales. La diferencia entre los parámetros de una función y las variables locales definidas dentro de una función radica en que los parámetros nos permiten comunicarnos con la función para introducir valores de entrada a través de ellos.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

Cualquier **variable local**, incluidos los parámetros, deja de existir en cuanto termina de ejecutarse la función en la que está definida.

Desde el cuerpo principal del programa no se puede acceder a las variables locales de ninguna función. Tampoco es posible acceder a las variables locales de una función desde otra.

Si intentamos acceder al valor de una variable local desde el cuerpo principal del programa o, en general, a una variable que no ha sido definida obtendremos un error típico de Python: NameError.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

```
def subrutina():
    a = 2
    print(a)
    return

a = 5
subrutina() # 2
print(a) # 5
```

```
def subrutina():
    a = 2
    print(a)
    return

subrutina() #2
print(a) # NameError: name 'a' is not defined
```

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

Las **variables globales** en Python son aquellas definidas en el cuerpo principal del programa fuera de cualquier función. Son accesibles desde cualquier punto del programa, incluso desde dentro de funciones. También se puede acceder a las variables globales de otros programas o módulos importados. Si en un módulo o programa determinado hay definida alguna variable global podremos acceder a ella importando dicho módulo.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

Las **variables globales** se pueden leer desde cualquier línea del programa simplemente haciendo referencia a ellas a través de su nombre. Pero **no** podemos modificar una variable global desde dentro de una función. Esto sucede así no por capricho, sino porque es un mecanismo de protección para evitar modificar sin querer una variable global que posiblemente terminaría alterando el funcionamiento normal de un programa.

Para modificarla es necesario utilizar el modificador **global**. Con esta declaración le estamos diciendo a Python que sabemos que vamos a utilizar una variable global y que queremos modificarla.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

```
contador = 10

def reiniciar_contador():
        global contador
        contador = 0



print(f'Contador antes es {contador}’)  # Contador antes es 10
reiniciar_contador()
print(f'Contador después es {contador}’) # Contador después es 0
```

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

A veces puede suceder que dentro de una función existe una variable local con el mismo nombre que una variable global, es decir, sus nombres entran en conflicto.

Cuando hacemos referencia a una variable Python busca primero en el ámbito local actual para ver si encuentra dicha variable y si no, la busca en el ámbito global. Es decir, en cierta forma, el ámbito local tiene preferencia sobre el global. Se puede decir que la variable local le hace sombra a la variable global.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

En general es mejor no utilizar variables globales. Normalmente son totalmente evitables y nos ahorramos muchos problemas y quebraderos de cabeza. Si necesitas que las funciones accedan a valores externos, ya sea para lectura o escritura, utiliza los parámetros de las funciones y su mecanismo de devolución de valores return.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

Ámbito de las variables.

El **ámbito de una variable** es la parte del programa donde esta variable puede ser utilizado, porque se conoce su existencia.

Variables globales: se definen en un lugar concreto del programa que no pertenezca a ninguna de las funciones.

Variables locales: deben estar definidas dentro de una función del programa y será en esta función donde tendrán su ámbito de validez.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

INTRODUCCIÓN A LA PROGRAMACIÓN

Variables y tipos de datos.

"Fin del tema"