**VARIABLES Y TIPOS DE DATOS**

# Asignar un valor a una variable

variable = valor

**# Tipos de datos**

texto = "**Hola Mundo**" # string

numero = **42** # integer

flotante = **3.14** # float

booleano = **True** # boolean

nulo = **None** # NoneType

# Imprimir en pantalla

**print**(variable)

**Operadores**

**# Operadores aritméticos:**

**+** : Suma. Suma dos valores.

**x = 5**

**y = 10**

**print (x + y)**

**→ 15**

**-** : Resta. Resta un valor de otro.

**x = 5**

**y = 10**

**print (x - y)**

**→ -5**

**\*** : Multiplicación. Se usa para multiplicar valores.

**x = 5**

**y = 10**

**print (x \* y)**

**→ 50**

**/** : División. Divide un valor por otro.

**x = 10**

**y = 5**

**print (x / y)**

**→ 2**

**%** : Módulo. Se utiliza para obtener el resto de la división entre dos valores

**x = 10**

**y = 3**

**print (x % y)**

**→ 1**

**\*\*** : Potencia. Se usa para elevar un valor a una potencia.

**x = 2**

**y = 3**

**print (x \*\* y)**

**→ 8**

**# Operadores de asignación:** **x** *op.asig* **valor**

**=** : El operador de asignación básico, que *asigna* el valor de la expresión a la variable.

**x = 5**

**print (x)**

**→ 5**

**+=** : El operador de suma y asignación, que *suma* el valor de la expresión a la variable.

**x** = **5**

**x** += **3**

**print (x)**

**→ 8**

**-=** : El operador de resta y asignación, que *resta* el valor de la expresión de la variable.

**x** = **5**

**x** -= **3**

**print (x)**

**→ 3**

**\*=** : El operador de multiplicación y asignación, que *multiplica* la variable por el valor de la expresión.

**x** = **5**

**x** \*= **3**

**print (x)**

**→ 15**

**/=** : El operador de división y asignación, que *divide* la variable por el valor de la expresión.

**x** = **5**

**x** /= **3**

**print (x)**

**→ 1.66**

**//=** : El operador de división entera y asignación, que realiza una *división entera* de la variable por el valor de la expresión.

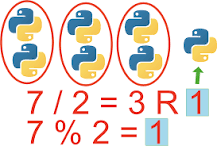
**x** = **5**

**x** //= **3**

**print (x)**

**→ 1.66**

**%=** : Este operador de módulo divide la variable del lado izquierdo por el valor del lado derecho y *asigna el resto* a la variable del lado izquierdo.

**x** = **7**

**x** %= **2**

**print (x)**

**→ 1**

**\*\*=** : El operador de exponente y asignación, que eleva la variable a la potencia del valor de la expresión.

**x** = **7**

**x** \*\*= **2**

**print (x)**

**→ 49**

**# Operadores de comparación:** Los operadores de comparación en Python se utilizan para comparar dos valores y producir un *valor booleano* (verdadero o falso) como resultado.

**==** : Comprueba si dos valores son iguales.

**a** = **7**

**b** = **7**

**print (a == b)**

**→ True**

**!=** : Comprueba si dos valores son diferentes.

**a** = **5**

**b** = **6**

**print (a != b)**

**→ True**

**>** : Comprueba si el valor de la izquierda es mayor que el de la derecha.

**a** = **6**

**b** = **5**

**print (a > b)**

**→ True**

**<** : Comprueba si el valor de la izquierda es menor que el de la derecha.

**a** = **5**

**b** = **6**

**print (a < b)**

**→ True**

**>=** : Comprueba si el valor de la izquierda es mayor o igual que el de la derecha.

**a** = **6**

**b** = **5**

**print (a >= b)**

**→ True**

**<=** : Comprueba si el valor de la izquierda es mayor o igual que el de la derecha.

**a** = **6**

**b** = **5**

**print (a <= b)**

**→ True**

**# Operadores lógicos:**

**and** : El operador "and" devuelve True si ambos operandos son verdaderos; en cualquier otro caso, devuelve False.

**a** = **5**

**b** = **1**

**if a > 0 and b < 20:**

**print (“Ambas condiciones son verdaderas”)**

**Este código imprimirá "Ambas condiciones son verdaderas" ya que ambas condiciones en el operador "and" son verdaderas.**

**or** : El operador "or" devuelve True si al menos uno de los operandos es verdadero; en cualquier otro caso, devuelve False.

**a** = **5**

**b** = **25**

**if a > 0 or b < 20:**

**print (“Al menos una de las condiciones es verdadera”)**

**Este código imprimirá "Al menos una de las condiciones es verdadera" ya que la primera condición en el operador "or" es verdadera.**

**not** : El operador "not" devuelve True si al menos uno de los operandos es verdadero; en cualquier otro caso, devuelve False.

**a** = **5**

**b** = **25**

**if not (a > 0 and b < 20):**

**print (“La condición es falsa”)**

**Este código imprimirá "La condición es falsa" ya que la condición dentro del operador "not" es verdadera, pero el operador "not" invierte su valor de True a False.**

**Estructuras de control de flujo**

**# Estructura if/else**

**if** condicion: Este operador permite ejecutar un bloque de código si una determinada condición es verdadera.

num = 10

**if** num > 0:

**print**("**El número es positivo**")

**else**: Este operador se utiliza en conjunto con **if** y permite ejecutar un bloque de código si la condición del **if** es falsa.

num = -5

**if** num > 0:

**print**("**El número es positivo**")

**else**:

**print**("**El número es negativo o cero**")

**# Estructura if/elif**

Este operador se utiliza en conjunto con **if** y permite evaluar múltiples condiciones en una estructura de control.

num = 0

**if** num > 0:

**print**("**El número es positivo"**)

**elif** num < 0:

**print**(**"El número es negativo"**)

**else**:

**print**(**"El número es cero"**)

**# Estructura ‘for’**

**for** variable **in** iterable:

Este operador se utiliza para iterar sobre una secuencia de elementos, como una lista o una cadena de caracteres.

frutas = [**"manzana", "banana",** **"cereza"]**

**for** fruta in frutas:

**print**(fruta)

**# Estructura ‘while’**

**while** condicion:

Este operador permite ejecutar un bloque de código mientras una determinada condición sea verdadera.

i = 0

**while** i < 10:

**print**(i)

i += 1

Otros son: break, continue, pass.

**Funciones**

# Definir una función

**def** **nombre\_de\_funcion**(argumento\_1, argumento\_2):

# código de la función

**return** resultado

# Llamar a una función

nombre\_de\_funcion(valor\_1, valor\_2)

**Listas**

# Crear una lista

lista = [**1**, **2**, **3**, **4**, **5**]

# Acceder a un elemento de la lista

primer\_elemento = lista[**0**]

ultimo\_elemento = lista[-**1**]

# Agregar un elemento a la lista

lista.append(**6**)

# Eliminar un elemento de la lista

lista.remove(**5**)

**Diccionarios**

# Crear un diccionario

diccionario = {'**llave\_1'**: valor\_1, '**llave\_2'**: valor\_2}

# Acceder a un valor en el diccionario

valor\_1 = diccionario['**llave\_1'**]

# Agregar una entrada al diccionario

diccionario['**llave\_3'**] = valor\_3

# Eliminar una entrada del diccionario

**del** diccionario['**llave\_2'**]