educación IT
CODE YOUR FUTURE

# Introducción a Python

Módulo 1

# Tipos de dato

# Tipos de dato

Los tipos de dato permiten **clasificar** la información que contiene una variable.

En Python, distinguimos cuatro tipos de dato básicos:

- Números enteros.
- Números de coma flotante (racionales).
- Cadenas (conjunto de caracteres para representar un texto).
- Booleanos (aquellos que solo pueden contener dos valores: verdadero o falso).

Ya hemos estado trabajando con números enteros y cadenas al hacer:

```
>>> a = 12
>>> saludo = "Hola mundo"
```

### Recuerda

"**>>>**" indica que estamos ejecutando el código en la consola interactiva. En esta clase vamos a estar haciendo bastante uso de ella.

Siempre que creamos una variable tendrá un tipo de dato —que es inferido por Python— asignado según el valor que le asignemos.

Creemos un par de variables más con los dos tipos de dato restantes:

```
>>> pi = 3.14
>>> encendido = True
```

**La sintaxis de Python para representar números decimales es usando la coma por un punto** (tal como ocurre por defecto en las calculadoras y otros lenguajes de programación).

En el caso de los *booleanos*, los únicos dos valores que aceptan son **`True`** y **`False`**.

> Nótese que **Python distingue entre minúsculas y mayúsculas;** por ende, `true` o `TRUE` son incorrectos.

También Python trae soporte por defecto para los números complejos *(complex)*, donde la parte imaginaria va a estar representada por la letra "**`j`**" en lugar de utilizar la "**`i`**" como en la notación matemática.

```
>>> c = 3 + 5j
```

Usando la instrucción `type` podemos saber en tiempo de ejecución a qué tipo de dato corresponde el contenido de una variable.

```
>>> type(a)
<class 'int'>

>>> type(saludo)
<class 'str'>

>>> type(pi)
<class 'float'>

>>> type(encendido)
<class 'bool'>

>>> type(c)
<class 'complex'>
```

Lo que nos interesa aquí es lo que aparece entre comillas simples luego de cada llamada a **`type()`**, que es cómo Python nomina a los tipos de datos:

- **`int`**: números enteros.
- **`str`**: cadenas.
- **`float`**: números de coma flotante.
- **`bool`**: booleanos.
- **`complex`**: complejos.

Una misma variable puede contener distintos tipos de dato durante la ejecución de un mismo programa.

## Por ejemplo

```
>>> b = 1
>>> type(b)
<class 'int'>

>>> b = "Hola mundo"
>>> type(b)
<class 'str'>
```

Esto se denomina ***tipado dinámico***. Un lenguaje de programación es dinámicamente tipado si una variable puede tomar valores de distinto tipo. La mayoría de los lenguajes de tipado dinámico son lenguajes interpretados, como Python.

# Comentarios (#)

# Comentarios (#)

Los comentarios son **líneas dentro de un archivo de código** que Python ignorará por completo.

Son útiles para hacer anotaciones en el código, y la sintaxis para crearlos es vía el **#** (numeral), como vemos en el primer ejemplo de la derecha.

Ocasionalmente, aunque poco recomendado, se pueden colocar en la misma línea que una instrucción de Python, como vemos en el segundo ejemplo.

```
# Imprimir un mensaje en pantalla.
print("Hola mundo")
```

```
x = 1  # Asignar 1 a la variable x.
```

El comentario generado con **#** es de **un renglón,** pero existe la posibilidad de generar comentarios tipo párrafo.

Los **comentarios multilínea** nos permiten generar comentarios de varios renglones, se generan con triple comilla simple o doble para abrir y cerrar.

```
'''''
Esto es
Un
Comentario
'''''
```

educación IT
CODE YOUR FUTURE

# ¡Sigamos trabajando!