
# Introducción a la Programación Segura

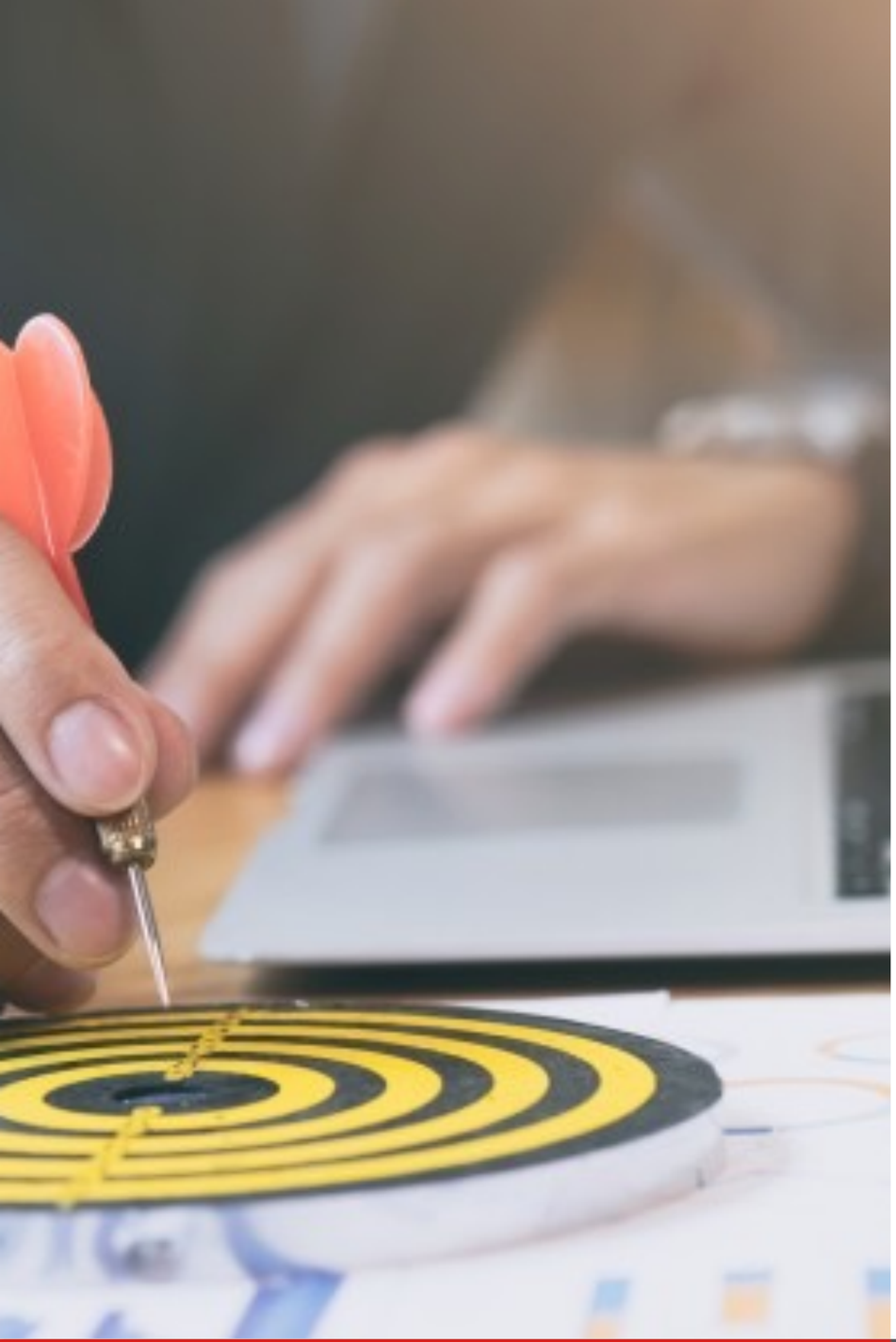
Estructuras de decisión, bucles y funciones del lenguaje Python.

# INTRODUCCIÓN A LA PROGRAMACIÓN SEGURA

UNIDAD 2: ESTRUCTURAS DE DECISIÓN, BUCLES Y FUNCIONES DEL LENGUAJE PYTHON.

## Contenidos

1. Variables.
2. Literales (enteros, flotantes, cadenas, valores booleanos).
3. La función print() y atributos de formato en la salida.
4. Operadores y expresiones.



# UNIDAD DE APRENDIZAJE

UNIDAD N°2: ESTRUCTURAS DE DECISIÓN, BUCLES Y FUNCIONES DEL LENGUAJE PYTHON.

Aprendizaje Esperado:

2.1. Desarrolla scripts de Python, para entregar solución a problemas de mediana complejidad, considerando los estándares internacionales de programación y la evaluación de los resultados obtenidos en base a la evidencia.

# Variables:

En Python, las variables son identificadores que se utilizan para almacenar datos y representan una ubicación de memoria donde se guarda un valor. Cada variable tiene un nombre que la identifica y permite acceder a su contenido durante la ejecución del programa.

Reglas que debes considerar para los nombres de las variables:

1. Deben comenzar con una letra (mayúscula o minúscula) o un guion bajo (_).
2. Pueden contener letras, dígitos y guiones bajos.
3. No pueden comenzar con un dígito numérico.
4. Son sensibles a mayúsculas y minúsculas (por ejemplo, "miVariable" y "mivariable" son dos variables distintas)

# Asignación de variables:

En Python, para asignar un valor a una variable, se utiliza el operador de asignación '=' y la sintaxis es la siguiente:

```
nombre_de_la_variable = "Bienvenidos al curso de Python"
```

Nombre de la variable

Operador de asignación

Valor de la variable

Python es un lenguaje polimórfico, lo cual significa que no necesita definir el tipo de dato a una variable, sino que el intérprete determina su tipo de acuerdo a su valor.

# Literales (enteros, flotantes, cadenas, valores booleanos):

Los literales son valores constantes que se utilizan para representar datos de diferentes tipos, como enteros, flotantes, cadenas y valores booleanos. Estos valores se utilizan directamente en el código sin necesidad de realizar cálculos o evaluaciones.

El concepto de "literal" y "variable" se refiere más al contexto y al propósito del valor en el código que a una clasificación específica en Python. Para determinar si un valor es un literal o una variable, debes considerar cómo se utiliza ese valor en el código.

En resumen, aunque la sintaxis de literales y variables puede verse similar, los literales son valores constantes que se escriben directamente en el código, mientras que las variables son identificadores que se utilizan para almacenar valores que pueden cambiar durante la ejecución del programa. Para identificar si un valor es un literal o una variable, debes revisar el contexto en el que se utiliza en el código y si está asignado a un nombre de variable o no

# Literales (enteros, flotantes, cadenas, valores booleanos):

## Literales enteros:

Los literales enteros representan números enteros, es decir, números sin parte decimal. Pueden ser positivos o negativos. Se escriben simplemente como una secuencia de dígitos.

Código:

```
codigo = 731598
telefono = 2536148
```

Estos datos no deberían cambiar de valor, en tiempo de ejecución, según el contexto del programa.

# Literales (enteros, flotantes, cadenas, valores booleanos):

## Literales flotantes:

Los literales flotantes representan números decimales. Se escriben usando un punto para separar la parte entera de la parte decimal.

Código:

```
litros = 2.5
volumen = 0.55
```

Estos datos no deberían cambiar de valor, en tiempo de ejecución, según el contexto del programa.

# Literales (enteros, flotantes, cadenas, valores booleanos):

## Literales cadenas:

Los literales cadenas representan secuencias de caracteres, como texto. Se escriben entre comillas simples o dobles.

Código:

```
nombre = "Teclado"
marca = "Microsoft"
```

Estos datos no deberían cambiar de valor, en tiempo de ejecución, según el contexto del programa.

# Literales (enteros, flotantes, cadenas, valores booleanos):

## Literales valores booleanos:

Los literales booleanos representan los dos valores de verdad: Verdadero (True) y Falso (False). Estos literales se utilizan principalmente en expresiones condicionales y bucles.

Código:

```
disponible = True
vigente = False
```

Estos datos no deberían cambiar de valor, en tiempo de ejecución, según el contexto del programa.

# Uso de comentarios en Python:

Los comentarios en Python son líneas de código que se utilizan para proporcionar información adicional, explicaciones o notas sobre el código que estás escribiendo. Los comentarios no se ejecutan como parte del programa y son ignorados por el intérprete de Python. Su único propósito es ayudar a los programadores a entender y recordar el propósito o funcionamiento de ciertas partes del código.

Comentarios de una línea:

Estos comentarios comienzan con el símbolo de almohadilla (#) y abarcan todo el texto que sigue en la misma línea. Todo lo que está después del símbolo # es considerado un comentario y no se ejecutará.

```
# Esto es un comentario de una línea
edad = 25  # Esto es otro comentario de una línea que explica que edad es igual a 25
```

# Uso de comentarios en Python:

Los comentarios en Python son líneas de código que se utilizan para proporcionar información adicional, explicaciones o notas sobre el código que estás escribiendo. Los comentarios no se ejecutan como parte del programa y son ignorados por el intérprete de Python. Su único propósito es ayudar a los programadores a entender y recordar el propósito o funcionamiento de ciertas partes del código.

Comentarios de múltiples líneas:

Los comentarios de múltiples líneas también se conocen como docstrings. Estos comentarios se utilizan para describir funciones, módulos o bloques de código más grandes. Los comentarios de múltiples líneas comienzan y terminan con tres comillas (''' o """) y pueden abarcar varias líneas.

```
'''
Este es un comentario
de múltiples líneas que explica
el propósito de este bloque de código.
'''
edad = 25
```

python™

inacap

# La función print() y atributos de formato en la salida:

## Función Print():

La función print() en Python se utiliza para mostrar mensajes o datos en la consola o terminal. Es una de las formas más comunes de generar una salida para el usuario o para el desarrollo y depuración de programas.

La sintaxis básica de print() es la siguiente:

Código:

```
print("Aprendiendo Python")
```

Salida en Terminal:

```
Aprendiendo Python
```

# La función print() y atributos de formato en la salida:

Función Print():
Podemos imprimir directamente un texto o el valor de una o varias variables:

Código:

```
nombre = "Valentina"
apellido = "Araya"
edad = 27
print(nombre,apellido,edad)
```

Salida en Terminal:

```
Valentina Araya 27
```

Podemos imprimir múltiples valores separándolos por comas, esto agregará un espacio entre los valores impresos.

# La función print() y atributos de formato en la salida:

## Atributos de la función Print() – sep:

Python proporciona atributos de formato que se pueden utilizar con la función print() para controlar cómo se muestra la salida en la consola.

Código:

```
nombre = "Valentina"
apellido = "Araya"
edad = 27
print(nombre,apellido,edad,sep="---")
```

Salida en Terminal:

```
Valentina---Araya---27
```

En este ejemplo el atributo **sep** agregará **---** entre los valores impresos, pero puede ser cualquier carácter mientras se encuentre entre las comillas.

# La función print() y atributos de formato en la salida:

## Atributos de la función Print() – end:

Este atributo se utiliza para especificar el carácter que se agrega al final de la impresión.
Por defecto, end es un salto de línea ("\n").

Código:

```
nombre = "Valentina"
apellido = "Araya"
edad = 27
print(nombre,apellido,edad,sep="---")
print("TÉCNICAS DE PROGRAMACIÓN EN PYTHON")
```

```
nombre = "Valentina"
apellido = "Araya"
edad = 27
print(nombre,apellido,edad,sep="---",end="fin")
print("TÉCNICAS DE PROGRAMACIÓN EN PYTHON")
```

Salida en Terminal:

```
Valentina---Araya---27
TÉCNICAS DE PROGRAMACIÓN EN PYTHON
```

```
Valentina---Araya---27finTÉCNICAS DE PROGRAMACIÓN EN PYTHON
```

En este caso **cambiamos** el valor por defecto del atributo **end** (que es un salto de línea) **por** el texto **fin**.

python™

Inacap

# Salida frente a entrada e ingreso de datos con la función input ():

La función input() se utiliza para recibir información o datos ingresados por el usuario a través del teclado. Esta función pausa la ejecución del programa y espera a que el usuario ingrese algún texto y presione la tecla "Enter". Una vez que el usuario proporciona la entrada, input() captura el texto ingresado y lo devuelve como una cadena (string).

Código:

```
nombre = input("Por favor, ingresa tu nombre: ")
print("Hola " + nombre + ", bienvenido al Taller de Python.")
```

Salida en Terminal:

```
Por favor, ingresa tu nombre: Alex
Hola Alex, bienvenido al Taller de Python.
```

El programa imprimirá un saludo personalizado usando el nombre ingresado por teclado.

# Salida frente a entrada e ingreso de datos con la función input ():

Código:

```
nombre = "Javiera"
edad = 27

print("Nombre:", nombre)
print("Edad:", edad)
```

Salida en Terminal:

```
Nombre: Javiera
Edad: 27
```

El programa imprimirá los valores de las variables nombre y edad junto a los textos “Nombre:” y “Edad:”.

inacap.cl