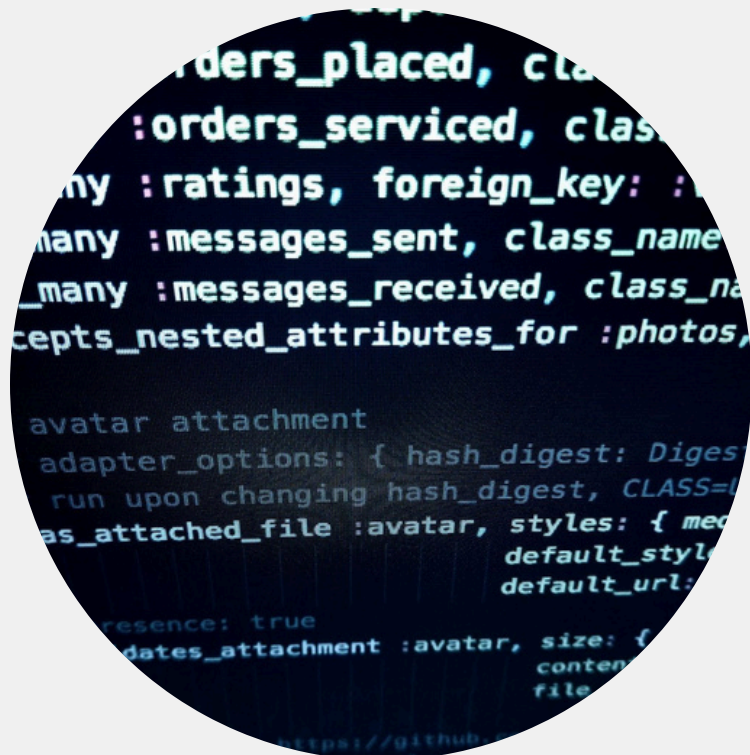




Módulo 1 – Unidad 2 – Lección 1

Actividad 3

Variables en Python

Variables en Python

La función `print()` en Python, se utiliza para imprimir o mostrar información en la salida estándar que comúnmente es la consola. Proporciona una forma sencilla de mostrar mensajes, resultados de cálculos u otra información relevante mientras ejecuta el programa.

Aquí tienen una explicación básica de cómo funciona:

```
# Ejemplo básico de uso de print  
mensaje = "Hola, mundo!"  
print(mensaje)
```

1

2

3

i

Formato de cadenas:

Pueden utilizar formateo de cadenas para incluir valores de variables en el texto que imprimen. Por ejemplo:

```
# Formateo de cadenas
nombre = "Ana"
edad = 30
print(f"Mi nombre es {nombre} y tengo {edad} años.")
```

Esto imprimirá: Mi nombre es Ana y tengo 30 años.

La función `print()` es una herramienta versátil y fundamental para mostrar información en la consola durante el desarrollo y ejecución de programas en Python.



Fin de línea:

Por defecto, `print()` añade un carácter de nueva línea (`\n`) al final de la salida. Pueden cambiar este comportamiento utilizando el parámetro `end`. Por ejemplo:

```
# Cambiando el carácter al final de la salida
print("Hola", end=" ")
print("mundo!")
```

Esto imprimirá: "Hola mundo!" en la misma línea.



Sintaxis:

La función `print()`, se llama con uno o más argumentos que representan los elementos que deseen imprimir. Pueden imprimir variables, strings, números u otras expresiones.

Separadores:

Por defecto, `print()` utiliza un espacio como separador entre los elementos. Pueden especificar un separador diferente utilizando el parámetro `sep`. Por ejemplo:

Por defecto, `print()` utiliza un espacio como separador entre los elementos. Pueden especificar un separador diferente utilizando el parámetro `sep`. Por ejemplo:

```
# Especificando un separador diferente
nombre = "Juan"
edad = 25
print(nombre, edad, sep="-")
```

Esto imprimirá: Juan-25.

Explicación de cómo declarar y asignar valores a variables en Python.

Declaración de Variables:

En Python, no es necesario declarar el tipo de variable antes de asignarle un valor, pueden nombrar una variable y asignarle un valor directamente.

```
# Ejemplo de declaración y asignación de variables
nombre = "John" # Una variable de cadena
edad = 25 # Una variable de entero
altura = 1.75 # Una variable de punto flotante
es_estudiante = True # Una variable booleana
```



Explicación de cómo declarar y asignar valores a variables en Python.

Asignación de Valores:

La asignación se realiza utilizando el operador de igualdad =. Pueden asignar un valor a una variable en la misma línea en que se declara, o pueden asignar valores a variables ya existentes.

```
# Asignación de valores en la declaración
ciudad = "Nueva York"
poblacion = 8500000

# Reasignación de valores
ciudad = "Los Ángeles"
poblacion = 4000000
```


Explicación de cómo declarar y asignar valores a variables en Python.

Convenciones de Nombres:

es buena práctica seguir algunas convenciones al nombrar variables. los nombres de variables suelen comenzar con una letra minúscula y usar la convención snake_case (palabras separadas por guiones bajos).

```
# Convenciones de nombres  
nombre_completo = "Jane Doe"  
temperatura_actual = 23.5
```

Explicación de cómo declarar y asignar valores a variables en Python.

Tipos de Variables Dinámicos:

Python es un lenguaje de tipos dinámicos, lo que significa que el tipo de una variable puede cambiar durante la ejecución del programa.



```
# Tipos de variables dinámicos
variable_dinamica = 10
print(variable_dinamica) # Salida: 10

variable_dinamica = "Hola"
print(variable_dinamica) # Salida: Hola
```

En Python, declarar y asignar valores a variables es un proceso intuitivo y flexible, el lenguaje se encarga de gestionar dinámicamente los tipos de variables, lo que facilita la escritura de código conciso y legible.

Tipos de datos básicos (enteros, flotantes, cadenas) y sus características.

Python cuenta con varios tipos de datos básicos que se utilizan para almacenar y representar información en programas. Aquí tienes una explicación de los tipos de datos más comunes:

Enteros

Flotantes

Cadenas

Enteros (int):

Representan números enteros sin decimales.

Ejemplos

5, -10, 1000.

En Python, los enteros pueden tener un tamaño ilimitado, lo que significa que puedes trabajar con números muy grandes sin preocuparte por desbordamientos.

```
# Enteros  
entero_positivo = 10  
entero_negativo = -5
```

Flotantes (float):

Representan números con decimales.

Ejemplos:

3.14, -0.5, 2.0.

En Python, los flotantes utilizan el estándar de punto flotante IEEE 754 y pueden representar números reales, aunque con ciertas limitaciones de precisión debido a la naturaleza finita de la representación en la computadora.

```
# Flotantes
flotante_1 = 3.14
flotante_2 = -0.5
```

```
# Operaciones con enteros y flotantes
resultado_suma = entero_positivo + flotante_1
resultado_multiplicacion = entero_negativo * flotante_2
```

```
# Impresión de resultados
print(resultado_suma)
print(resultado_multiplicacion)
```



Cadenas (str):

Representan secuencias de caracteres.

Ejemplos:

"Hola", 'Python', "123".

Pueden de nirse utilizando comillas simples (') o dobles ("). Ambas formas son equivalentes en Python.

Las cadenas son inmutables, lo que significa que no puedes modificar los caracteres individuales directamente, pero puedes realizar diversas operaciones y manipulaciones con ellas.

```
# Cadenas
cadena_simple = 'Hola, mundo!'
cadena_doble = "Python es divertido"
```

```
# Concatenación de cadenas
saludo_completo = cadena_simple + ' ' + cadena_doble
```

```
print(saludo_completo)
```

Ejercicios

Ejercicios relacionados con los tipos de datos básicos en Python (enteros, flotantes y cadenas). Estos ejercicios son útiles para practicar las operaciones comunes y mejorar la comprensión de las características de estos tipos de datos:

*Ejercicios sobre
Enteros y Flotantes*

*Ejercicios sobre
Cadenas de Texto*

Estos ejercicios proporcionan una variedad de situaciones para trabajar con números enteros, flotantes y cadenas de texto, permitiendo a los estudiantes practicar operaciones comunes y familiarizarse con las peculiaridades de cada tipo de datos.

Operaciones con Cadenas:

- Crea dos variables de cadena llamadas `cadena1` y `cadena2`.
- Concatena las dos cadenas y guarda el resultado en una nueva variable.
- Imprime la longitud de la cadena resultante.

Formato de Cadenas:

- Crea una variable con tu nombre.
- Utiliza el formato de cadena para imprimir un mensaje de bienvenida personalizado.

Subcadenas y Métodos:

- Crea una cadena larga y utiliza el método `split()` para dividirla en una lista de palabras.
- Encuentra e imprime la posición de una subcadena específica.

Métodos de Mayúsculas y Minúsculas:

- Crea una cadena en minúsculas y conviértela a mayúsculas usando los métodos `upper()` y `lower()`.



Operaciones Aritméticas:

- Crea dos variables, a y b, con valores numéricos. Realiza operaciones
- aritméticas básicas (suma, resta, multiplicación, división, exponente) con estas variables e imprime los resultados.

División y Módulo:

- Divide dos números enteros y guarda el resultado en una variable llamada
- resultado. Utiliza el operador módulo (%) para obtener el residuo de la división e imprímelo.

Precisión de Flotantes:

- Crea una variable c con un valor flotante.
- Realiza operaciones aritméticas que involucren flotantes y enteros.
- ¿Cómo se manejan las operaciones mixtas?