Introducción a Python

Tipos de datos y valores

{desafío} latam_

Distinguir los tipos de datos y sentencias básicas del lenguaje para la construcción de programas.

- Unidad 1:
  Introducción a Python
- Unidad 2:
  Sentencias condicionales e iterativas
- Unidad 3:
  Estructuras de datos y funciones

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Reconoce los tipos de datos usados en el entorno Python y su uso para la construcción de un programa.*

{desafío} latam_

¿Qué entendemos por reglas de programación?

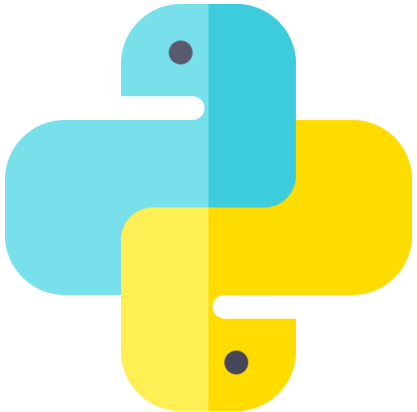
/* Tipos de datos */

Tipos de datos nativos en Python

Son distintas formas de mostrar información en el lenguaje.

En Python son:

- Valores numéricos
- Strings



Valores numéricos

Integer

```
# estos son valores integer
print(27)
print(-5)
print(0)
print(3 + 7)
print(31 - 7)
print(4 * 8)
```

Floats

```
# estos son valores float
print(27.6)
print(-5.9)
print(0.0)
print(3.2 + 7.456)
print(31.5 - 7.0)
print(4.190 * 8.12)
print(9.0 / 8.1)
```

- El separador decimal en Python es un “.”

- Cualquier operación de división entre integers, en Python el resultado será un float incluso si el resultado resulta en un valor entero. Ejemplo: 5.0 // 2 es 2.0

Es posible realizar operaciones matemáticas combinando integers y float de manera válida como se muestra a continuación:

```
print(3.5 + 12)
print(20 * 45.6)
```

Strings

- Son elementos encerrados entre comilla simple (‘) o comilla doble (“).
- Para definirse como un string válido, la comilla al inicio y al final debe ser la misma.
- Normalmente, un string es un texto, o una etiqueta como nuestros nombres de pila, pero también podría contener caracteres numéricos, sin embargo no se podrán realizar operaciones matemáticas con estos últimos.

```
# estos son valores string
print('hola')
print("150")
print('este es un texto más largo')
```

Strings

Las operaciones matemáticas entre un valor numérico y un string no son válidas.
Pero los strings sí tienen otras operaciones válidas.

Concatenación

```
# Concatenación
print("Carlos" + "Santana") # CarlosSantana
print("Carlos " + "Santana") # Carlos Santana
```

```
# Concatenación
print("15" + "23") # esto resulta en 1523
```

Duplicación

```
# Duplicación
print(3 * "Carlos") # CarlosCarlosCarlos
print(5 * "12") # 1212121212
```

/* Métodos */

.count()

Cuenta el número de veces que aparece un carácter alfanumérico.

```
print("Santana".count("a"))
# entrega un 3 ya que el string tiene 3 "a"
```

.upper()

Transforma el string a mayúsculas.

```
print("Santana".upper())
# resulta en SANTANA
```

.lower()

Transforma el string a minúsculas.

```
print("Santana".lower())
# resulta en santana
```

.title()

Permite colocar mayúsculas sólo a la primera letra de cada palabra.

```
print("carlos santana".title())
# resulta en Carlos Santana
```

len()

Permite contar el número de caracteres en un string. Este no es un método, por lo que para aplicarla, se debe incluir un string dentro de esta función.

```
print(len("Carlos Santana"))
# resulta en 14
```

join()

Permite unir muchos elementos separados por un string.

```
print(", "join(["a","b","c"]))
```

En este caso el separador (la coma) es quien recibe el método join, en el cual se agregarán los elementos a, b y c. Los elementos entre corchetes corresponden a una lista, la cual se explicará en detalle más adelante.

Salto de línea

Carácter especial que nos permite agregar un salto de línea dentro de un string.

```
# El caracter para denotar un salto de línea es \n
print("hola\na\ntodos")
```

Nótese que se utiliza el símbolo de slash invertido y seguidamente la letra n. \n

```
hola
a
todos
```

Valores booleanos

- Son valores lógicos que pueden tomar sólo dos valores:
  - True
  - False

- Estos valores son el resultado de alguna prueba lógica o de una asignación directa; donde este tipo de dato se revisará en detalle al momento de aprender Control de Flujo.

/* Variables */

¿Qué es una variable?

Se utiliza para guardar el resultado de operaciones o almacenar valores para ser reutilizados en el futuro. Podemos entender las variables como contenedores.

Una variable se compone de:

- Un nombre
- Un valor

Consideraciones
de variables

- El nombre de una variable comienza con minúscula.
- Si se requiere utilizar más de una palabra, éstas deben estar unidas por un guión bajo o underscore; a esto se le conoce como snake_case.
- Los nombres de las variables no pueden tener espacios ni comenzar con un número.
- Es importante utilizar nombres que sean representativos del valor que se quiere almacenar.

Por ejemplo, si queremos crear una variable que almacene la cantidad de alumnos de un curso, su nombre podría ser **cant_alumnos**.

Tipos de datos

de una variable

Una variable poseerá un tipo de dato asociado, según el valor que se le asigne, donde los tipos de datos pueden ser, entre otros, **integer**, **string**, **float** o **bool**.

Python dispone de la función **type()** para poder identificar el tipo de dato de una variable:

```
entero = 2
decimal = -6.5
texto = "Hola Mundo"

print(type(entero)) # int
print(type(decimal)) # float
print(type(texto)) # str
```

Es importante señalar que la función type se aplica a la variable, y no al `print()`.

Ejecutar `type(print(entero))` devolverá un tipo None, ya que print no es un tipo de dato, sino una función para imprimir en pantalla valores o información.

Manipulando variables

- Es posible trabajar con variables previamente definidas. Estas variables contendrán el valor asignado a lo largo de la ejecución de todo el código o hasta que este sea reasignado.
- Una variable de un tipo específico conserva todas las propiedades de su tipo de dato original.

```
a = 2 # se asigna el valor 2 a la variable a
a = a + 1 # se suma 1 al valor de a y se reasigna
print(a) # se muestra el resultado final
```

```
3
```

```
nombre = "Carlos"
# se asigna el valor Carlos a la variable nombre
apellido = "Santana"
# se asigna el valor Santana a la variable
apellido

print(nombre + " " + apellido)
# se muestra el resultado final
```

```
"Carlos Santana"
```

{desafío} latam_

Transformando los datos

Interpolación

Permite introducir una variable, un dato o una operación válida dentro un String.

```
nombre = 'Carlos'
apellido = 'Santana'
# Interpolación
print("Mi nombre es {} {}".format(nombre,
apellido))
```

```
nombre = 'Carlos'
apellido = 'Santana'
# Interpolación
print(f"Mi nombre es {nombre} {apellido}")
```

```
Mi nombre es Carlos Santana
```

Precisión de datos

Permite controlar la precisión de los decimales de manera muy sencilla.

```
# Operación que arroja varios decimales
print(f'El resultado es {1/9}')
```

```
El resultado es 0.1111111111111111
```

```
# Muestro sólo 2 decimales
print(f'El resultado es {1/9:.2f}')
```

```
El resultado es 0.11
```

/* Ingresando datos */

Ingresando datos

con input

- Input abre un prompt o un cuadro de diálogo que permite al usuario ingresar texto que puede ser utilizado en el programa.
- Adicionalmente, si dentro del Input agregamos un texto, este texto será utilizado como mensaje adjunto al cuadro de diálogo.
- El valor ingresado con input() siempre será de tipo string.

```
>>> input("Ingrese su Nombre: ")
Ingrese su Nombre: Carlos Santana
```

Ingresando datos

con input

- Al invocar **input()** la consola quedará bloqueada hasta que ingresemos una secuencia de caracteres y presionemos la tecla enter.
- Este valor puede almacenarse dentro de una variable y de esa manera ser utilizada en nuestro código:

```
>>> nombre = input("Ingrese su Nombre: ")
Ingrese su Nombre: Carlos Santana
```

Ejercicio guiado

Presentándome con Python

Un gran conferencista, que siempre tenía que presentarse ante el público que asistía a sus conferencias, debía siempre dar una pequeña reseña de su vida, para que lo pudieran conocer. Hacer esto cada vez, lo aburrió y decidió pedirte que pudieras automatizar esta tarea, ya que lo bueno de la programación es automatizar tareas que son tediosas o que son propensas a error. Para esto, el conferencista te muestra una presentación antigua de una sus conferencias y te solicita elaborar una solución que permita modificarla levemente pero de manera fácil y rápida:

```
Mi nombre es Bill, tengo 52 años y me desempeño como CEO en Microsoft.
Soy Creativo, Apasionado y Visionario.

Mi pasatiempo es jugar Golf y me gustaría poder jubilarme pronto.
```

Presentándome con Python

Solución - Paso 1

Entendamos el problema, esta es una plantilla, por lo tanto, podríamos determinar qué aspectos de la plantilla podrían cambiar. Algunos de estos aspectos son: **Nombre**, **Edad**, **Ocupación**, **Lugar de Trabajo**, **Características**, **Pasatiempos**, **Algo que hacer**.

Al entender esto, quizás podríamos decir que estos son aspectos variables, que podrían cambiar en el tiempo o podrían ser modificados dependiendo de la persona.

Presentándome con Python

Solución - Paso 1

Entonces, hagamos un código que solicite estas variables, **presentador.py**:

```
nombre = input('Ingrese su Nombre: ')
edad = input('Ingrese Edad: ')
ocupacion = input('Ingrese Ocupación: ')
lugar = input('En dónde?: ')
caracteristica_1 = input('Ingrese 3 características:\n1. > ')
caracteristica_2 = input('2. > ')
caracteristica_3 = input('3. > ')
pasatiempo = input('Cuál es tu pasatiempo: ')
hacer = input('¿Que quieres hacer? ')
```

Presentándome con Python

Solución - Paso 2

La idea es poder utilizarlas para construir el texto deseado, para ello, podríamos utilizar lo recién aprendido acerca de Interpolación. En este caso utilizaremos el string multilínea para poder escribir el texto de manera más ordenada:

```
print(f'''

Mi nombre es {nombre}, tengo {edad} años y me desempeño como {ocupacion} en {lugar}.
Soy una persona {caracteristica_1}, {caracteristica_2} y {caracteristica_3}.

Mi pasatiempo es {pasatiempo} y me gustaría poder {hacer}.''')
```

Presentándome con Python

Solución - Paso 3

Ahora, ejecutemos el programa:

```
Ingrese su Nombre: Bill
Ingrese Edad: 52
Ingrese Ocupación: CEO
En dónde?: Microsoft
Ingrese 3 características:
1. > Creativo
2. > Apasionado
3. > Visionario
Cuál es tu pasatiempo: jugar Golf
¿Que quieres hacer? jubilarme pronto


Mi nombre es Bill, tengo 52 años y me desempeño como CEO en Microsoft.
Soy una persona Creativo, Apasionado y Visionario.

Mi pasatiempo es jugar Golf y me gustaría poder jubilarme pronto.
```

Como se puede ver, se logró generar un programa en Python muy sencillo que permitirá generar presentaciones de manera muy rápida cambiando sólo algunos parámetros.

¿Cuáles son los tipos de datos nativos en Python?

¿De qué manera se permite al usuario ingresar datos en Python de manera interactiva?

Próxima sesión...

- *Reconoce los operadores matemáticos, lógicos y de comparación para la construcción de expresiones.*
- *Reconoce las sentencias básicas del lenguaje como condicionales y bucles para la construcción de programas.*

{desafío}
latam_

Academia de
talentos digitales