TIPOS DE DATO, OPERADORES Y EXPRESIONES

10145

FUNDAMENTOS DE PROGRAMACIÓN PARA INGENIERÍA

INTRODUCCIÓN

- El objetivo del curso es aprender las nociones básicas de programación
- Programar significa escribir órdenes o instrucciones para que un computador pueda ejecutarlas
- Como los computadores no son máquinas inteligentes, necesitamos escribir estas instrucciones en un lenguaje que este pueda comprender
- Para ello usaremos el lenguaje de programación Python

OPERADORES Y PRECEDENCIA

OPERADORES ARITMÉTICOS

- Uno de los primeros pasos para aprender un lenguaje de programación, es el uso de este para realizar aritmética básica
- Para ello Python nos provee de los operadores matemáticos binarios básicos:
  - Suma (+) → `2 + 2 = 4`
  - Resta (-) → `4 - 5 = -1`
  - Multiplicación (*) → `3 * 4 = 12`
  - División (/) → `9 / 2 = 4.5`
- Y otros un poco más complejos…

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

OPERADORES ARITMÉTICOS

- Tales como:
  - Exponenciación (potencia) (**) → 2 ** 4 = 16
  - División entera (//) → 9 // 2 = 4
  - Módulo (resto de la división entera) (%) → 6 % 4 = 2

- Estos operadores, se les suele llamar operadores binarios puesto que requieren dos valores para operar

- Sin embargo, estos no son los únicos operadores que Python ofrece

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

PRECEDENCIA DE OPERADORES

- Cuándo existe una expresión con múltiples operadores, Python la evaluará respetando el orden de precedencia de la siguiente tabla:

OPERACIÓN	OPERADOR	ARIDAD	ASOCIATIVIDAD	PRECEDENCIA
EXPONENCIACIÓN	**	BINARIA	DERECHA	1
IDENTIDAD	+	UNARIA	-	2
NEGACIÓN	-	UNARIA	-	2
MULTIPLICACIÓN	*	BINARIA	IZQUIERDA	3
DIVISIÓN	/	BINARIA	IZQUIERDA	3
DIVISIÓN ENTERA	//	BINARIA	IZQUIERDA	3
MÓDULO	%	BINARIA	IZQUIERDA	3
SUMA	+	BINARIA	IZQUIERDA	4
RESTA	-	BINARIA	IZQUIERDA	4

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

PRECEDENCIA DE OPERADORES

- Cuándo existe una expresión con múltiples operadores, Python la evaluará respetando el **orden de precedencia** de la siguiente tabla:

OPERACIÓN	OPERADOR	ARIDAD	ASOCIATIVIDAD	PRECEDENCIA
ONENCIACIÓN	**	BINARIA	DERECHA	1
AD	+	UNARIA	-	2
	-	UNARIA	-	2
				3
				3
				3
				3
				4
				4

Además de las reglas de precedencia de operadores, Python tiene dos consideraciones importantes a la hora de evaluar expresiones:

- Cuándo existen operadores del mismo nivel de precedencia, siempre se da más **prioridad a la operación que esté más a la izquierda** de la expresión
- Las evaluaciones se realizan secuencialmente, es decir, Python siempre evalúa **una operación a la vez**

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

PRECEDENCIA DE OPERADORES

- Para revisar el funcionamiento de la precedencia de operadores, consideremos el siguiente ejemplo:

```
-2 * 9 / 6 +  4 ** 2 // 3 + 5 % 3 + 1
```

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

PRECEDENCIA DE OPERADORES

- Para revisar el funcionamiento de la precedencia de operadores, consideremos el siguiente ejemplo:

```
-2 * 9 / 6 +  4 ** 2 // 3 + 5 % 3 + 1
```

- Si escribimos esta expresión en Python, al evaluarla nos entrega como resultado el valor 5.0

- El detalle de cómo Python realiza el cálculo, podemos observarlo a continuación

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

```
- 2 * 9 / 6 +  4 ** 2 // 3 + 5 % 3 + 1
  - 2 * 9 / 6 +  16 // 3 + 5 % 3 + 1
    -2 * 9 / 6 +  16 // 3 + 5 % 3 + 1
      -18 / 6 +  16 // 3 + 5 % 3 + 1
        -3.0 +  16 // 3 + 5 % 3 + 1
          -3.0 +  5 + 5 % 3 + 1
            -3.0 + 5 + 2 + 1
              2.0 + 2 + 1
                4.0 + 1
                  5.0
```

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

PRECEDENCIA DE OPERADORES

- Al igual que en las matemáticas, es posible utilizar paréntesis para alterar la precedencia de operadores. Por ejemplo, la expresión:

$$\frac{-5+4}{2 * 8}$$

- Se puede escribir en Python como:

```
(-5 + 4) / (2* 8)
```

- Es importante destacar que para alterar la precedencia, Python sólo acepta paréntesis redondos, dado que las llaves `{, }` y los corchetes `[, ]` están reservados para operaciones más complejas

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

TIPOS DE DATOS

TIPOS DE DATOS

- En el ejercicio anterior, como pudimos ver, Python entregó como resultado el valor 5.0 en vez de 5
- Esto se debe a que los computadores **no manejan los números del mismo modo que los manejamos los humanos**
- El computador, necesita conocer de antemano qué operaciones puede realizar con cada dato, por lo que asigna un **tipo de dato** a cada valor
- Los tipos de dato no solo distinguen y discriminan números, sino que también se usan para reconocer texto, valores de verdad y otros elementos importantes del lenguaje

TIPOS NUMÉRICOS

- En Python existen tres tipos de datos para representar números: `int`, `float` y `complex`
  - **Enteros** (`int`): Representan números enteros, positivos y negativos
  - **Números de punto flotante** (`float`): Representan números no enteros, no son una representación perfecta del conjunto matemático $\mathbb{R}$, también se les llama flotantes
  - **Números complejos** (`complex`): Representan números que tienen parte real e imaginaria, sin embargo, tampoco son una representación totalmente fiel del conjunto matemático $\mathbb{C}$

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

TIPOS NUMÉRICOS

- En el caso del ejercicio anterior, el resultado 5.0 fue un número de punto flotante, debido a que al llegar a la división:

```
-18 / 3
```

- Python obliga que el resultado sea un flotante, para asegurar que la división sea un número matemáticamente coherente
- Luego, el resto de las operaciones que involucran un número flotante (`3.0`) resultan en flotante, pues Python opta por mantener el tipo de dato que represente un conjunto más amplio de valores, en este caso, los flotantes

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

OTROS TIPOS DE DATO

- Cómo se mencionó anteriormente, no existen solamente tipos de dato para representar números, sino otros objetos del lenguaje
- Veremos varios de ellos durante el semestre, pero es importante conocer inicialmente otros dos que serán importantes en las próximas clases, el booleano y el string

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

OTROS TIPOS DE DATO

- Booleano (`bool`): Se utiliza para representar valores lógicos
  - Un Booleano sólo puede ser `True` o `False` dependiendo del caso
  - Internamente se representan como números, 0 para `False` y 1 para `True`
- String (`str`): Se utiliza para almacenar cadenas de texto
  - Se identifican fácilmente porque inician y cierran con comillas. Por ejemplo: `'HOLA', "2.7",'EL5398'`
  - Se utilizan para capturar entradas para el programa o para informar de resultados a un usuario

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

CONVERSIÓN DE TIPOS

- A pesar de que Python busca mantener el tipo de dato más general al evaluar una expresión y en los cálculos aritméticos realizará la conversión de tipo de forma implícita, existe la posibilidad de transformar a un tipo explícitamente
- Para ello, existen comandos especiales, llamados funciones que realizan la conversión de tipo (*typecast*)
- Python no siempre sabrá como realizar la conversión de tipo, por lo que, una mala utilización de estos comandos podría llevar a errores en el programa

FUNCIONES DE *TYPECASTING* (1/2)

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

FUNCIÓN	EJEMPLO	RESULTADO

`int()`	`int(3.8)`
	`int('450')`
	`int(True)`
	`int('A12')`
		`3`
		`450`
		`1`
		`Error`

`float()`	`float(3)`
	`float(" -3.5 ")`
	`float(False)`
	`float((2+3j))`
		`3.0`
		`-3.5`
		`0.0`
		`Error`

`complex()`	`complex(2,3)`
	`complex('2-4.5j')`
	`complex(False, True)`
	`complex('2','3.5')`
		`(2+3j)`
		`(2-4.5j)`
		`(0+1j)`
		`Error`

FUNCIONES DE *TYPECASTING* (2/2)

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

FUNCIÓN	EJEMPLO	RESULTADO

`bool()`	`bool(3.8)`
	`bool('True')`
	`bool(0.0)`
	`bool((0+0j))`
	`bool('CHAO')`
	`bool(2,3)`
		`True`
		`True`
		`False`
		`False`
		`True`
		`Error`

`str()`	`str(3)`
	`str(4.3440000)`
	`str(False)`
	`str((2+3j))`
	`str(3,4)`
		`'3'`
		`'4.344'`
		`'False'`
		`'(2+3j)'`
		`Error`

VARIABLES Y EXPRESIONES

OPERADOR ASIGNACIÓN

- En Python, al igual que en la mayoría de los lenguajes de programación populares, existe un operador especial, llamado asignación
- Este operador permite almacenar un valor o el resultado de una expresión con un nombre determinado por el programador para usarlo posteriormente
- La asignación tiene la siguiente estructura:
  - `<identificador> = <expresión>`

¡Recuerden que no es una comparación ni una equivalencia matemática!

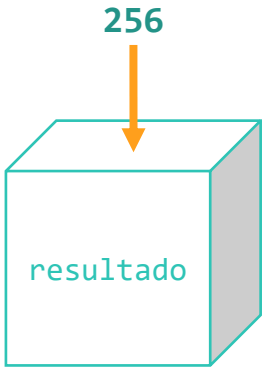
EJEMPLO DE ASIGNACIÓN

- Consideremos la expresión:

```
resultado = 2 ** 8
```

- Python primero evalúa la expresión, obteniendo el valor **256** y lo almacena en una variable llamada "**resultado**"

- Podemos imaginar una variable como una "**caja**" en la memoria del programa en la cuál el valor está guardado



ASIGNACIÓN

- Una asignación es una sentencia con la siguiente estructura:

```
<identificador> = <expresión>
```

- Reglas de un identificador:
  - El primer caracter no puede ser un dígito
  - Puede llevar letras, dígitos y el caracter subrayado (_)
  - No puede coincidir con las palabras reservadas de Python:

```
and, assert, break, class,
continue, def, del, elif, else,
except, exec, finally, for,
from, global, if, import, in,
is, lambda, not, or, pass,
raise, return, try, while,
yield
```

- Puede ser:
  - Un valor constante (un número o texto)
  - Una operación entre números
  - Una operación entre variables y constantes previamente declaradas
  - Mezcla entre operaciones, variables y números

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

CONSTANTES Y VARIABLES

- En Python podemos almacenar valores como constantes o variables
- En la práctica, diferenciamos las constantes de las variables, para indicarle a otros programadores que valores podrían ser modificados y qué valores no
- Por ejemplo, si hacemos un programa que calcule el área y perímetro de una circunferencia necesitamos definir dos valores iniciales:
  - El radio, que correspondería a una variable, pues su valor puede modificarse
  - El valor de π que corresponde a una constante, pues este nunca cambia

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

CONSTANTES Y VARIABLES

```
PI = 3.1415926535897931
radio = 2

perimetro = 2 * PI * radio
area = PI * radio ** 2
```

- En este caso los nombres `radio`, `perimetro` y `area` corresponderían a variables, pues son valores que dependen de otros o que podrían variar dependiendo del caso
- En cambio PI es un valor matemático definido y siempre debería ser igual

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

CONSTANTES Y VARIABLES

- En Python se utilizan reglas distintas para variables y constantes
  - Las **variables** se escriben en **minúsculas** y **separando palabras distintas con guiones bajos** (`lower_case_with_underscores`) buenos nombres de variables serían `valor`, `color`, `palabra_original`, `lista_estudiantes`
  - Las constantes en cambio se escriben con **mayúsculas** y también separando las variables con guiones bajos (`UPPER_CASE_WITH_UNDERSCORES`) buenos nombres para constantes serían: `VALOR_PI`, `NOMBRE_ARCHIVO`, `GRAVEDAD`, `ARCHIVO_SALIDA`

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

USO DE VARIABLES

- Como vimos en el ejemplo anterior, es posible **combinar variables con expresiones aritméticas dentro de una misma expresión** para usar los valores almacenados en ellas para el cálculo
- Además, como **siempre se realiza primero la evaluación de la expresión a la derecha y luego se hace la asignación**, es posible en algunos casos, utilizar la misma variable en ambos lados de la expresión
- ¿Con qué valor terminaría la variable resultado?

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

EJEMPLO

- Hagamos la evaluación paso a paso:

```
valor_original = 8

resultado = valor_original / 2

resultado = resultado ** 3

resultado = resultado + resultado / 2

resultado = int(resultado)
```

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

EJERCICIO PROPUESTO 1

- ¿Con qué valor termina la variable resultado en este caso?

```
valor = 12

resultado = valor + resultado

resultado = float(resultado ** 2)

resultado = resultado – valor % 3
```

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

EJERCICIO PROPUESTO 2

- Asumiendo que tengo el siguiente código y que no puedo definir nuevas variables, ni nuevos valores
  - `a = 1`
  - `b = 2`
- ¿Qué instrucciones debo escribir para que se cumpla que:
  - `a = 2` y
  - `b = 1`?

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

CONTENIDOS DE HOY

- Operadores matemáticos
- Precedencia de operadores
- Conversión de tipos
- Asignación
- Variables y constantes

TAREAS PARA TRABAJO AUTÓNOMO

1. Revisar el material del segundo tema del curso: Estructura de scripting (ppt y Colab)

2. Revisar el 'manual de legibilidad del código y buenas prácticas de programación' en Uvirtual.usach.cl

¿CONSULTAS?