DMC | INSTITUTE

Módulo Python for Analytics – Sesión 02

Diploma Data Scientist

Docente: Geider Nuñuvero

REGLAS

Se requiere **puntualidad** para un mejor desarrollo del curso.

Para una mayor concentración **mantener silenciado el micrófono** durante la sesión.

Las preguntas se realizarán **a través del cha**t y en caso de que lo requieran **podrán activar el micrófono**.

Realizar las actividades y/o tareas encomendadas en **los plazos determinados**.

Identificarse en la sala Zoom con el primer nombre y primer apellido.

ITINERARIO

09:00 AM – 07:00 PM	***Soporte técnico DMC***
07:30 PM – 08:50 PM	***Agenda***
08:50 PM – 09:00 PM	***Pausa Activa***
09:00 PM – 10:30 PM	***Agenda***

Horario de Atención Área Académica y Soporte

Lunes a Viernes 09:00 am a 10:30 pm/ Sábados 09:00 am a 02:00 pm

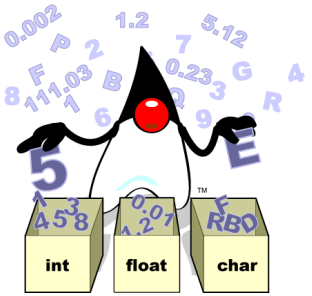
Sesión 2

- Variables y tipos de datos en Python
- Operaciones básicas
- Listas
- Tuplas: Indexing y Slicing.
- Diccionarios.
- Conjuntos o Sets.

Entendiendo los principales tipos de datos en Python

1. **Booleano:** Toma el valor de **Verdadero** o **Falso**
   - b1 = **True**
   - b2 = **False**

2. **Numérico:** Podemos clasificarlos en 3 grupos:
   - Entero: -18 1 5 100
   - Decimal: -7.5 2.4 3.252 8.0
   - Complejo: 0.5 + 2j



3. **String:** Es una cadena de texto, compuesta por un conjunto de caracteres:
   - s1 = ‘ Data Science ’
   - s2 = “ Python ”

Operaciones aritméticas básicas

Expresiones

Operaciones entre int y floats

- Suma → i + j
- Diferencia → i - j
- Producto → i * j
- División → i / j
- División entera → i //j
- Resto → i % j
- Potencia → i ** j

Precedencia

Parentesis
Potencia
Producto
Suma
Izquierda a derecha

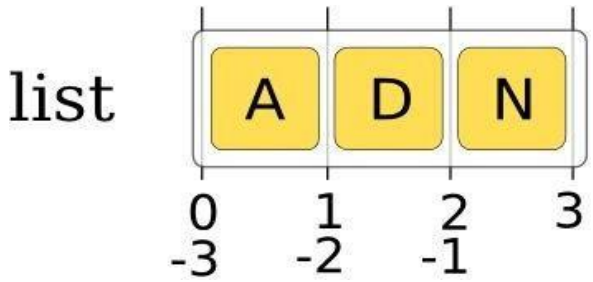
¿Cuál es el resultado en Python?

```
7 - 6 + 2 ** 3 / 4 * 5
```

Lista (1/2)

Colección ordenada y mutable de elementos del mismo o diferente tipo de dato.

- Contiene elementos de diferentes clases (incluso otras listas).
- Es del tipo "**list**".
- Cada elemento de la secuencia se le asigna un número que corresponde a su posición en la secuencia (índice). El primer índice es cero (0), el segundo uno (1), y así sucesivamente.



Lista (2/2)

Funciones frecuentes

- **append** -> añade un elemento a la lista
- **len** -> longitud de la lista

Operadores

- **+** -> concatenar dos listas
- ***** -> repetición de elementos
- **in** -> pertenencia

```
['soltero','casado']+['viudo', 'separado']
>> ['soltero', 'casado', 'viudo', 'separado']

['oh!', 1]*4
>> ['oh!', 1,'oh!', 1,'oh!', 1,'oh!', 1]

3 in [1, 2, 3]
>> True
```

Tupla (1/2)

Colección ordenada e inmutable de elementos del mismo o diferente tipo de dato.

- Es del tipo "**tuple**".

Las tuplas se diferencian de las listas en que:

1. No se puede modificar sus elementos (**inmutable**).
2. Se usan paréntesis en lugar de corchetes para construir una tupla.

```
tupla[0] = 3
>> TypeError: 'tuple' object does not support item assignment
```

Tupla (2/2)

Colección ordenada e inmutable de elementos del mismo o diferente tipo de dato.

- Igual que en las listas, cada elemento de la secuencia se le asigna un número que corresponde a su posición en la secuencia (**índice**). El primer índice es cero (0), el segundo uno (1), y así sucesivamente.
- Igual que con las listas, para la selección se **usan los corchetes [ ].**
- Igual que con las listas, se pueden concatenar las tuplas.

Indexing

Las secuencias ordenadas se acceden vía índices entre corchetes

- Los elementos de las secuencias ordenadas (strings, listas, y tuplas) se acceden vía índices que se indican entre corchetes después del nombre del objeto que contiene la secuencia.

0	1	2	3	4	5	6	7
'teresa'	'andrea'	'pablo'	'isabel'	'alvaro'	'angelo'	'mirian'	'carolina
-8	-7	-6	-5	-4	-3	-2	-1

```
nombres = ['teresa', 'andrea', 'pablo', 'isabel', 'alvaro', 'angelo',
'mirian', 'carolina']
nombres[0]
>> 'teresa'

nombres[-1]
>> 'carolina'
```

Slicing

Se puede obtener una parte de la secuencia usando: secuencia[x:y:z] con x, y, z como números enteros

- Devuelve una nueva secuencia tal que:
  - Es del mismo tipo que la original (si era una lista, devuelve una lista, e igual con tuplas y cadenas)
  - Contiene los elementos desde secuencia[x] hasta secuencia[y] (sin incluirla), saltando z elementos cada vez (se puede omitir si z=1)

```
nombres = ['teresa', 'andrea', 'pablo', 'isabel', 'alvaro', 'angelo', 'mirian', 'carolina']
nombres[2:4]
>> ['pablo', 'isabel']

nombres[1:5:3]
>> ['andrea', 'alvaro']
```

Diccionario (1/2)

Colección no ordenada y mutable de elementos del mismo o diferente tipo de dato.

- Los diccionarios definen una relación uno a uno entre claves y valores.

```
{ key1:value1 ,  key2:value2 ,  key3:value3 , . . . }
```

- En un diccionario las **claves (keys) son únicas** y no se pueden modificar.
- En cambio, los valores (values) no tienen que ser únicos y son actualizables.
- Es del tipo **dict**.

Diccionario (2/2)

- Para la selección, se usan los **corchetes** [ ] y la **clave** correspondiente.
- Acceso a claves inexistentes devuelve error **KeyError**.
- También podemos devolver un valor por defecto con el método **get**. El primer parámetro del método get es la clave de búsqueda, el segundo el valor por defecto.

```
d={'jose': 24, 'luis': 34, 'maria': 21}
d['jose']
>> 24

d['paco']
>> KeyError: 'paco'

d.get('paco',-1)
>> -1
```

Set (1/2)

Colección no ordenada de elementos del mismo o diferente tipo de dato, sin elementos duplicados.

- Es del tipo "**set**"
- Es la implementación del concepto matemático de conjunto, por lo que permite diversas operaciones como unión, intersección, diferencia, etc.
- Los sets no aceptan objetos mutables, por lo que no se le pueden pasar listas como elementos pero sí tuplas.

Set (2/2)

Funciones frecuentes

- **add**() -> Añadir un elemento.
- **difference**() -> Operación de diferencia. (-)
- **union**() -> Unión de dos conjuntos. (|)
- **intersection**() -> Intersección de dos conjuntos. (&)
- **in** -> Operador de pertenencia.

```
a = set(' abcdefg hh ')
b = set('defghijklmn')
a
>> {'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h'}

a & b
>> {'d', 'e', 'f', 'g', 'h'}
```

GRACIAS