Universidad **Ricardo Palma**

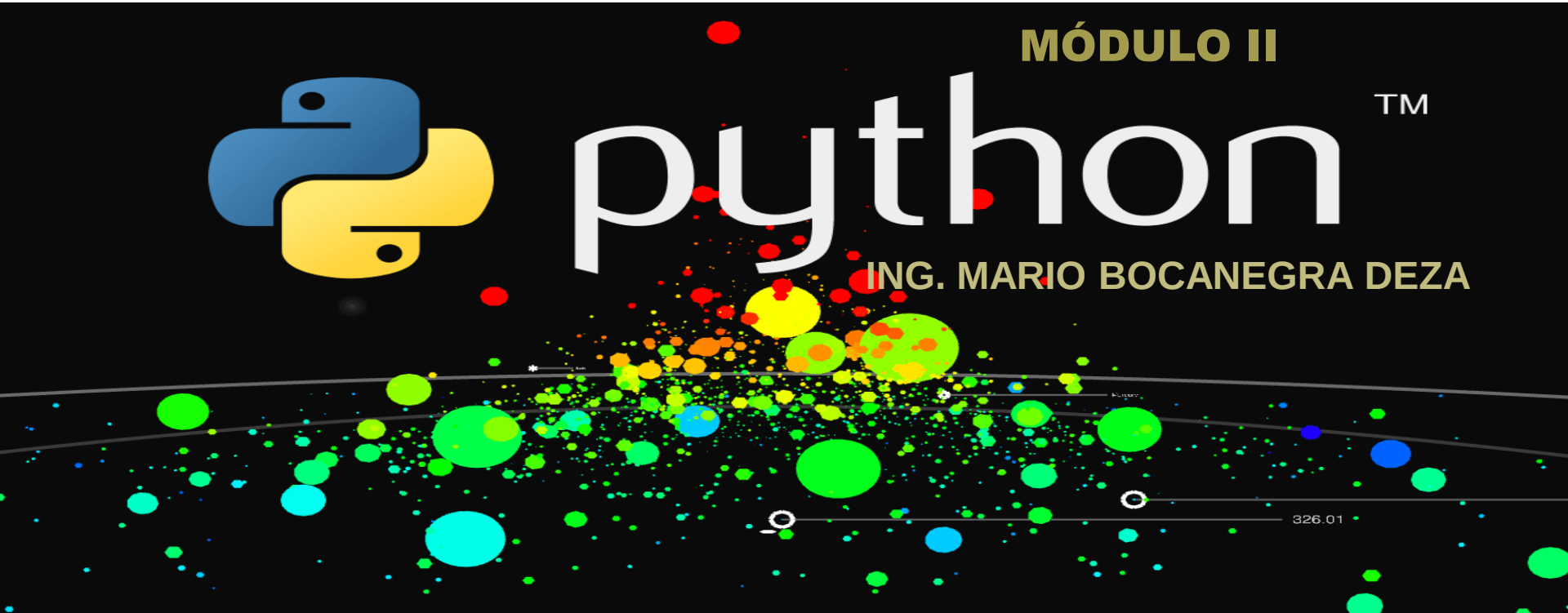
RECTORADO

Formamos seres humanos para una cultura de paz

Primer Programa de Especialización

INTRODUCCIÓN AL DATA SCIENCE

TIPO DE DATOS, VARIABLES Y E/S DE DATOS



Contenido

- ✓Tipo de enteros
- ✓ Entrada /Salida
- ✓Operadores aritméticos
- ✓Operadores relacionales
- ✓Tipo de booleanos
- ✓Tipo cadenas
- ✓Tipo de conjuntos
- ✓Tipo de tuplas
- ✓Tipo de listas
- ✓Tipo de diccionarios

Tipo de enteros

Números Enteros :

- Son aquellos que no tienen decimales, tanto positivos como negativos (además del cero).

```
>>> entero=3
>>> print(entero+2)
5
>>> print(entero*2)
6
>>> print(entero-1)
2
>>> print(entero+1+2+3)
9
```

Números Reales :

- Son los que tienen decimales.

real = 0.2703

- También se puede utilizar notación científica, y añadir una e (de exponente) para indicar un exponente en base 10

real = 0.1e-3

0.1x10-3 = 0.1 x 0.001 = 0.001

```
>>> 2e3
2000.0
>>> 0.1e-3
0.0001

>>> real=0.5
>>> print(1+2+3+4+5+6+real)
21.5
```

Entrada/Salida de Datos

Lectura de datos de teclado :

```
>>> cantidad = int(input("Ingrese un cantidad:"))
Ingrese un cantidad:12
>>> print(cantidad,"unidades")
12 unidades
>>> nota = float(input("Ingrese Calificación:"))
Ingrese Calificación:10.5
>>> print("La calificación es :",nota)
La calificación es : 10.5

>>> texto=input("Ingrese un texto : ")
Ingrese un texto : Un texto de ejemplo
>>> print("El texto ingresado : ", texto)
El texto ingresado :  Un texto de ejemplo
```

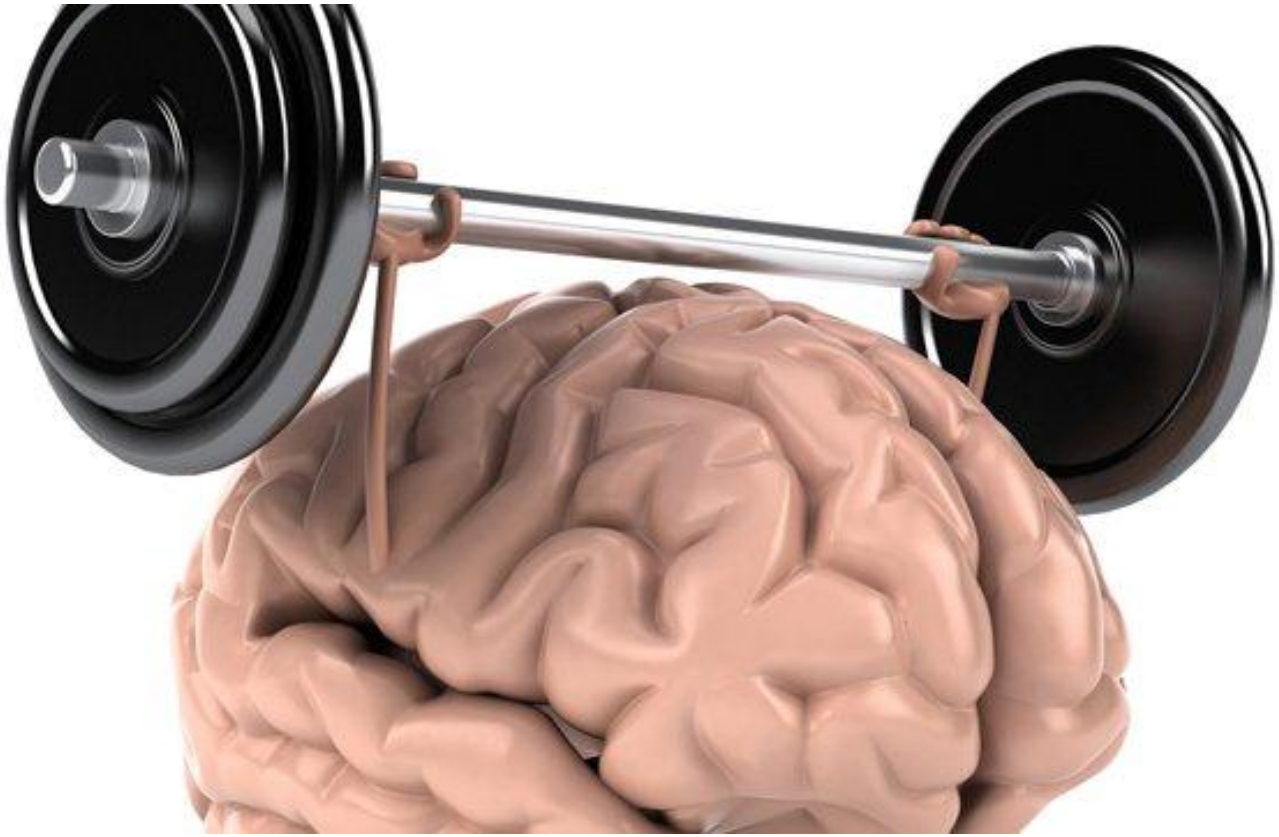
Operadores aritméticos

Operador	Descripción	Ejemplo
+	Suma	r = 3 + 2 # r es 5
-	Resta	r = 4 – 7 # r es -3
-	Negación	r = -7 # r es -7
*	Multiplicación	r = 2 * 6 # r es 12
**	Exponente	r = 2 ** 6 # r es 64
/	División	r = 3.5 / 2 # r es 1.75
//	División entera	r = 3.5 // 2 # r es 1.0
%	Módulo	r = 7 % 2 # r es 1

Operadores relacionales

Operador	Descripción	Ejemplo
==	¿son iguales a y b?	r = 5 == 3 # r es False
!=	¿son distintos a y b?	r = 5 != 3 # r es True
<	¿es a menor que b?	r = 5 < 3 # r es False
>	¿es a mayor que b?	r = 5 > 3 # r es True
<=	¿es a menor o igual que b?	r = 5 <= 5 # r es True
>=	¿es a mayor o igual que b?	r = 5 >= 3 # r es True

Llego La Hora … A practicar



Tipo de booleanos

- Un dato tipo lógico puede tener dos valores : True (cierto) y False (falso).
- Tienes 3 operadores lógicos : **and** (conjunción), **or** (disyunción), **not** (negación).

and		
operandos		resultado
izquierdo	derecho	
True	True	True
True	False	False
False	True	False
False	False	False

or		
operandos		resultado
izquierdo	derecho	
True	True	True
True	False	True
False	True	True
False	False	False

not	
operando	resultado
True	False
False	True

Tipo cadenas

- Es un texto encerrado entre comillas simples o dobles.
- Dentro de las comillas se pueden añadir caracteres especiales como ‘\ln’, el carácter de nueva línea o el de tabulación ‘\t’ .

```
>>> nombre = 'Pepe'
>>> nombre
'Pepe'
```

```
>>> 'a' + 'b'
'ab'
>>> nombre = 'Pepe'
>>> nombre + 'Cano'
'PepeCano'
```

```
>>> 'Hola' * 5
'HolaHolaHolaHolaHola'
>>> '-' * 60
'------------------------------------------------------------'
>>> 60 * '-'
'------------------------------------------------------------'
```

```
>>> cadenab = "Texto entre comillas dobles"
>>> print(cadenab)
```

Tipo de conjuntos

- Es una bolsa sin ordenar de valores únicos.
- Puede contener simultáneamente valores de cualquier tipo de datos.
- Con dos conjuntos se puede efectuar las típicas operaciones de unión, intersección y diferencia de conjuntos .

```
>>> un_conjunto={1,2,3,4,5}
>>> un conjunto
```

```
>>> un_conjunto.add(6)
>>> un_conjunto
```

```
>>> un_conjunto.update({7,8})
>>> un_conjunto
```

```
>>> un_conjunto.discard(8)
>>> un_conjunto
```

```
>>> un_conjunto.remove(7)
>>> un_conjunto
```

Operaciones típica de conjuntos :

```
>>> un_conjunto = {2, 4, 5, 9, 12, 21, 30, 51, 76, 127, 195}
>>> 30 in un_conjunto
True
>>> 31 in un_conjunto
False
>>> otro_conjunto = {1, 2, 3, 5, 6, 8, 9, 12, 15, 17, 18, 21}
>>> un_conjunto.union(otro_conjunto)
{1, 2, 195, 4, 5, 6, 8, 12, 76, 15, 17, 18, 3, 21, 30, 51, 9, 127}
>>> un_conjunto.intersection(otro_conjunto)
{9, 2, 12, 5, 21}
>>> un_conjunto.difference(otro_conjunto)
{195, 4, 76, 51, 30, 127}
```

Tipo de tuplas

- Es una variable que permite almacenar varios datos inmutables de tipos diferentes .

```
>>> mi_tupla=('cadena de texto',15,2.8,'otro dato',25)
>>> print(mi_tupla[1])
15
>>> print(mi_tupla[1:4])
(15, 2.8, 'otro dato')
>>> print(mi_tupla[3:])
('otro dato', 25)
>>> print(mi_tupla[:2])
('cadena de texto', 15)
>>> print(mi_tupla[-1])
25
>>> print(mi_tupla[-2])
otro dato
```

Tipo de listas

- Es similar a una tupla con la diferencia que permite modificar los datos una vez creado.

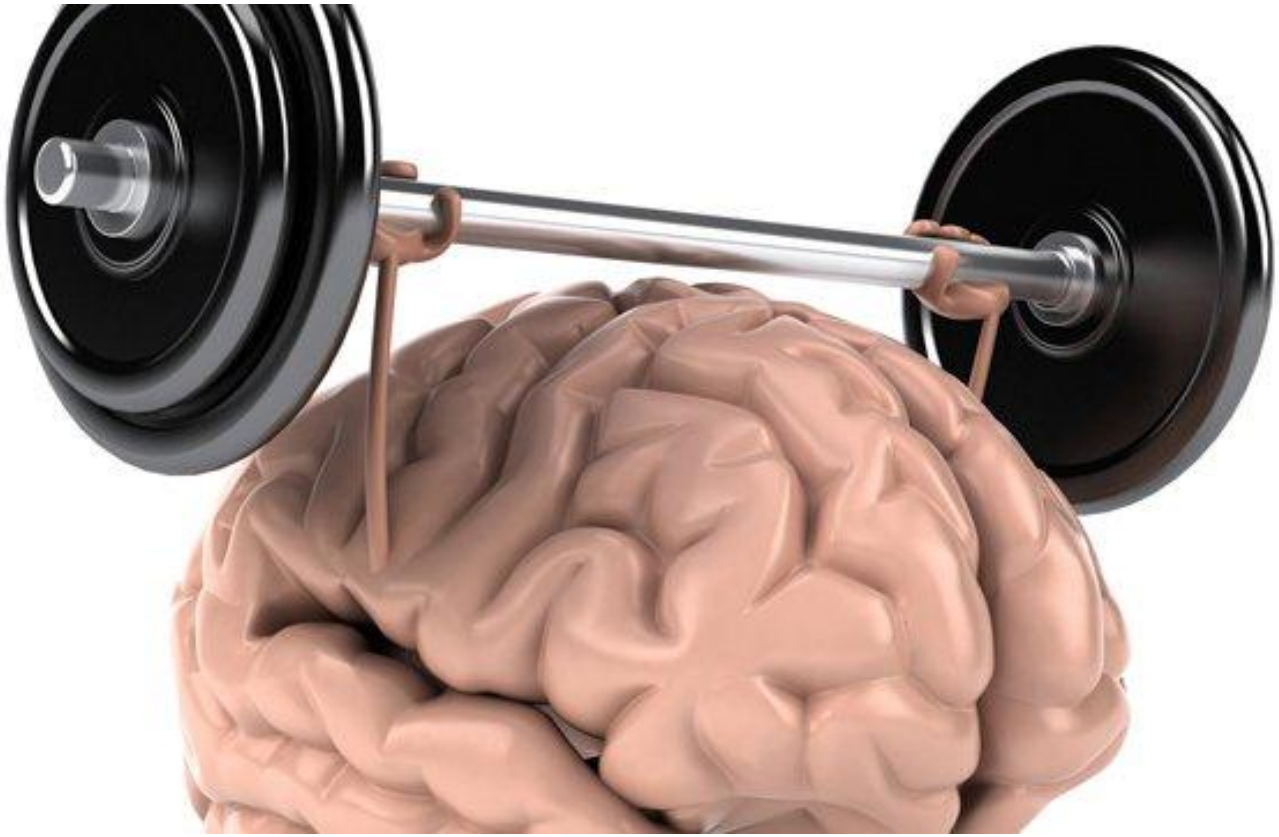
```
>>> mi_lista=['cadena de texto',15,2.8,'otro dato',25]
>>> print(mi_lista)
['cadena de texto', 15, 2.8, 'otro dato', 25]
>>> print(mi_lista[1])
15
>>> print(mi_lista[1:4])
[15, 2.8, 'otro dato']
>>> print(mi_lista[-2])
otro dato
>>> mi_lista[2]=3.8
>>> print(mi_lista)
['cadena de texto', 15, 3.8, 'otro dato', 25]
>>> mi_lista.append('Nuevo Dato')
>>> print(mi_lista)
['cadena de texto', 15, 3.8, 'otro dato', 25, 'Nuevo Dato']
>>> mi_lista.extend([1,"cadena",5.6])
>>> print(mi_lista)
['cadena de texto', 15, 3.8, 'otro dato', 25, 'Nuevo Dato', 1, 'cadena', 5.6]
>>> del mi_lista[7]
>>> print(mi_lista)
['cadena de texto', 15, 3.8, 'otro dato', 25, 'Nuevo Dato', 1, 5.6]
```

Tipo de diccionarios

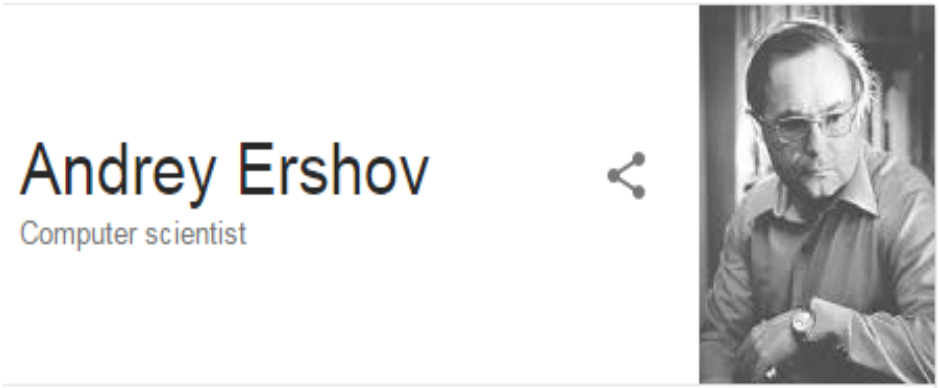
- Es un conjunto desordenado de parejas clave-valor.
- Cuando se añade una clave a un diccionario, se tiene que añadir también un valor para esa clave.

```
>>> un_dic={"Star Wars":"George Lucas","Kill Bill":"Tarantino","A.I.":"Steven Spielberg"}
>>> print(un_dic["Kill Bill"])
Tarantino
>>> un_dic["Star Wars"]="Gareth Edwards"
>>> print(un_dic)
{'Star Wars': 'Gareth Edwards', 'Kill Bill': 'Tarantino', 'A.I.': 'Steven Spielberg'}
>>> del(un_dic['Star Wars'])
>>> print(un_dic)
{'Kill Bill': 'Tarantino', 'A.I.': 'Steven Spielberg'}
```

Llego La Hora … A practicar

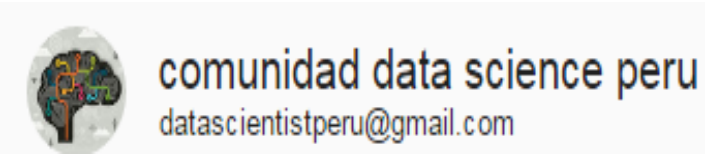


“Mi sentimiento es que cuando preparo un programa, es como componer poesía o música, como dijo una vez Andrei Ershov, programar nos puede dar ambas una satisfacción intelectual y emocional, porque es un verdadero logro dominar la complejidad y establecer un sistema de reglas consistentes”.




fanpage
facebook


Comunidad
Data Science
Perú
@comunidaddatascience


comunidad data science peru
datascientistperu@gmail.com