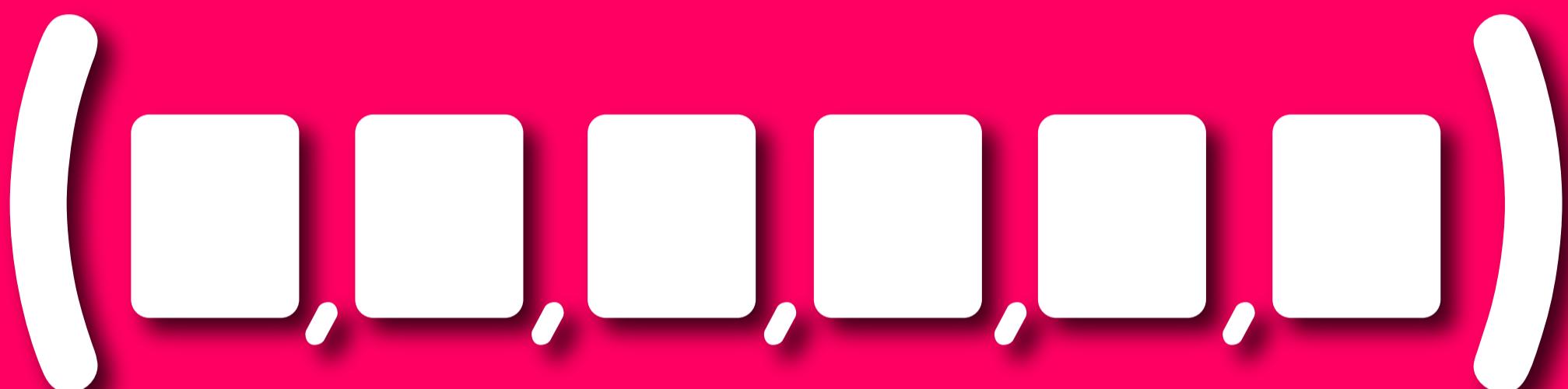




El futuro digital  
es de todos

MinTIC



# Tuplas (Práctico)



Universidad de Caldas

# Hola:

Vamos a ver de manera práctica las funciones relacionadas con **Tuplas**. Recordemos que las Tuplas son una estructura de datos compuestos, es decir, que no representa solamente un dato simple, sino que puede representar varios datos bajo un mismo identificador. Estos son inmutables, lo que significa que una vez se han asignado valores, no pueden ser cambiados.

Vamos a ver las funciones básicas con Tuplas en este laboratorio para tener claro su funcionamiento, luego veremos otro laboratorio, en el que haremos un ejercicio más completo y más contextualizado.

## Tuplas Vacias

```
tupla1=()
print(type(tupla1))

tupla2=tuple()
print(type(tupla2))
I

I
D <class 'tuple'>
<class 'tuple'>
```

Lo primero, con cualquier estructura es: cómo se crea. Aquí vemos que, para crear una Tupla, simplemente lo podemos asignar con paréntesis vacíos, es decir, una tupla vacía, y si imprimimos el tipo de la Tupla, nos arroja que es de tipo Tupla, o también puedo crearla con el segundo método. Tupla2 es un método que lo vamos a llamar método constructor, simplemente porque está construyendo una Tupla, así que cualquiera de los dos métodos funciona. No tiene mucho sentido crear una Tupla vacía, pero va a ser de mucha utilidad, sobre todo cuando necesitamos hacer comparaciones.

Hay otro aspecto importantísimo con respecto a la creación de elementos

## Creación con elementos

```
▶ circulo1=3,-4,3,"Blue" # Empaquetamiento
print(type(circulo1))

circulo2=(10,5,6,"Yellow")
print(type(circulo2))

circulo3=tuple((2,4,5,"Green"))
print(type(circulo3))
print(circulo1)
print(circulo2)
print(circulo3)

↳ <class 'tuple'>
<class 'tuple'>
<class 'tuple'>
(3, -4, 3, 'Blue')
(10, 5, 6, 'Yellow')
(2, 4, 5, 'Green')
```

Aquí estoy creando una Tupla, que estoy llamando círculo1, en la que tenemos unos valores: 3, -4, 3 y blue, esto se llama empaquetamiento. Estamos empaquetando esos cuatro (4) valores en uno solo que es círculo1. Cualquiera de estos tres (3) métodos hacen exactamente lo mismo.

## Tupla unitaria

```
▶ tupla=(8,)
print(type(tupla))
I
I
↳ <class 'tuple'>
```

Podemos crear también Tuplas con un solo elemento, que es como lo vemos aquí.

# Acceso a los elementos de la tupla

```
▶ circulo3=tuple((2,4,5,"Green"))
print(circulo3[2])
print(circulo3[0:2]) ↴
print(circulo3[-1])
```

```
⇨ 5
(2, 4)
Green
```

El acceso siempre es similar. Creamos un Tupla y después accedemos a la posición dos (2) de esa Tupla, que sería el número cinco (5). Aquí lo podemos hacer mediante slicing, y podemos hacer accesos negativos -1, este es devolverse desde el final, una posición.

# Tamaño de las tuplas

```
▶ len(circulo3)
```

```
4
```

Otra función es la longitud, la cual me retorna el número de elementos que tiene la Tupla.

# Desempaquetado de tuplas

```
I  
x,y,radio,color=circulo3 #Desempaquetamiento  
print(x,y,radio,color)  
D 2 4 5 Green
```

Esta operación se llama desempaquetamiento. Lo que tenemos es la Tupla al contrario, estamos “desgranando” una estructura de datos compuestas, por estructura de datos simple, es decir, cuatro (4) variables simples.

# Retorno de valores

```
I  
def operaciones_aritmeticas(num1,num2):  
    s=num1+num2  
    r=num1-num2  
    m=num1*num2  
    d=num1/num2  
  
    return s,r,m,d  
I  
r1,r2,r3,r4 = operaciones_aritmeticas(10,20)  
print(r1,r2,r3,r4)  
D 30 -10 200 0.5
```

Otro elemento muy importante es el manejo de funciones, por ejemplo: en esta vamos a retomar las operaciones aritméticas (suma, resta, multiplicación y división).

# Tuplas con parámetros

```
▶ import math

def area_circulo(circulo:tuple):
    radio=circulo[2]
    area=math.pi*radio
    return area

area=area_circulo((2,2,2,"Blue"))
print(area)

▷ 12.566370614359172
```

También es muy útil cuando necesitamos pasar parámetros. Por ejemplo: nosotros podemos pasar un círculo, y en lugar de poner los cuatro (4) valores, podemos pasar un círculo que es de tipo Tupla. Es muy útil cuando necesitamos pasar parámetro, para no pasar uno a uno los elementos que tengo allí, en la definición de la función.

## Named tuples

```
▶ from collections import namedtuple
Circulo=namedtuple('Circulo', ['x', 'y','radio','color']) #Nuevo tipo de datos

print(type(Circulo))
print(type(int))

c1= Circulo(2,3,5,"Blue") #Definiendo una variable
print(type(c1))

print(c1.color)
```

```
▷ <class 'type'>
<class 'type'>
<class '__main__.Circulo'>
Blue
```

Por último, están las Named Tuples o Tuplas Nombradas que son un tipo de colección donde estamos creando un círculo que va a tener X, Y, radio y color. Lo más interesante aquí es que si imprimimos el tipo de círculo no retorna a Tupla específicamente, sino a un tipo de Tupla, esto quiere decir que círculo es un tipo de dato creado por nosotros. Recuerden que las variables pueden tener un tipo de datos y nosotros aquí acabamos de definir nuestro propio tipo de dato.

Hemos terminado este video que corresponde a las funciones básicas como Tuplas, que nos quedan claras para poder entender el laboratorio y el taller que están relacionados con el tema.



Universidad de Caldas