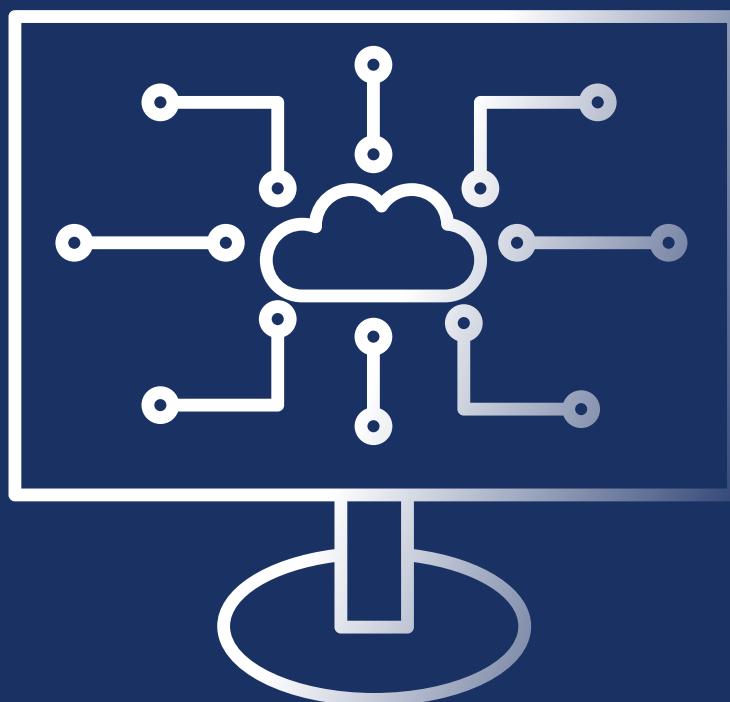


Programación I



Lic. Julia Monasterio
marmonasterio@uade.com.ar



Unidades

-  **Unidad 1:** Repaso de matrices, estructura, operaciones y uso en funciones
-  **Unidad 2:** Implementación y gestión en proyectos en Git
-  **Unidad 3:** Listas avanzadas, cadena de caracteres y expresiones regulares.
-  **Unidad 4:** Manipulación avanzada de diccionarios, tuplas y conjuntos.
-  **Unidad 5:** Excepciones y pruebas unitarias
-  **Unidad 6:** Archivos
-  **Unidad 7:** Recursividad

Clase N°7

TEMAS

- Tuplas

Tuplas

Que son las tuplas?

- Son una secuencia de valores agrupados. Una tupla sirve para agrupar, como si fueran un único valor, varios valores que, por su naturaleza deben ir juntos.

Tuplas

- Son similares a las listas, ya que se accede a ellas a través de subíndices
- Para crear una tupla se encierran sus elementos **entre paréntesis** en lugar de corchetes

```
gaseosas=("Schewepes","Coca Cola","Sprite")  
print(gaseosas)
```

```
('Schewepes', 'Coca Cola', 'Sprite')
```

Tuplas

- Si se escribe una secuencia de elementos sin paréntesis pero separados por comas, también se crea una tupla. **Siempre es recomendable por claridad usar parentesis.**
- Por ejemplo:

```
elementosPc="Monitor", "Teclado", "Mouse"
```

```
print(elementosPc)
```

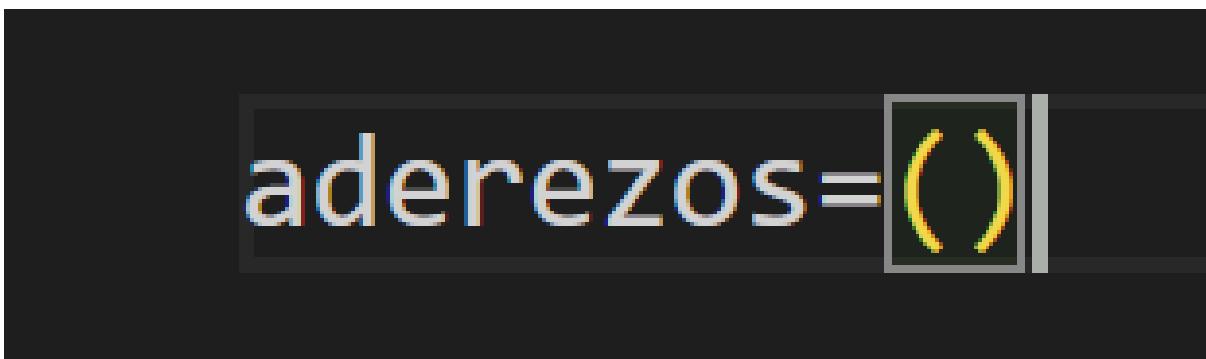
```
('Monitor', 'Teclado', 'Mouse')
```

Tuplas

- Cuando en una función se devuelve mas de un valor lo que se esta devolviendo es una **tupla**

Tupla vacía

- Una tupla vacía se define con un par de paréntesis, sin nada en su interior.
- Por ejemplo:



A screenshot of a dark-themed code editor showing the line of code `aderezos=()`. The cursor is positioned at the end of the tuple definition, indicated by two vertical bars. The code is written in a light-colored font against a black background.

Tupla de un elemento

- Para crear una tupla con un solo elemento es necesario escribir coma luego de este. Por ejemplo:

galletitas= “criollitas”,

- Los paréntesis también son opcionales en este caso

Listas vs Tuplas

- Las listas son mutables pero las tuplas **SON INMUTABLES**, es decir no pueden ser modificadas
- Debido a que son inmutables, las tuplas carecen del método **append()**
- La única forma de agregar elementos es a través del operador de concatenación creando una nueva tupla:
 - **gaseosas = gasesosas + (“fanta”,)**

Error al modificar elemento

- Tratar de modificar un elemento de una tupla mediante su subíndice de forma directa provocara una excepción de tipo **TypeError**

```
alumno=(1345,"Pedro Hernandez","Moreno 411",(15,"Junio",1998))

alumno[1]="Juan Perez"
```

```
alumno[1]="Juan Perez"
~~~~~^~~
TypeError: 'tuple' object does not support item assignment
```

Operador multiplicador

- El operador de repetición * también puede ser usado con tuplas

```
binario=(0,1)*3
```

```
print(binario)
```

```
(0, 1, 0, 1, 0, 1)
```

Tuplas

- En una misma tupla se pueden combinar distintos tipos de datos.

```
primavera=(21, "Septiembre")
```

- Si bien las listas también soporta tipos de datos heterogéneos, por convención generalmente las listas tienen un único tipo de dato y las tuplas diferentes tipos de datos

Tuplas

- También pueden ser anidadas

```
alumno=(1345,"Pedro Hernandez","Moreno 411",(15,"Junio",1998))

print(alumno)
```

- Es importante no usar tuplas para las matrices, sino seguir utilizando listas. **Debido a que las tuplas no se pueden modificar, eso estaría obligando a Python a copiar la tupla de matrices todo el tiempo en nueva tupla.**

Acceder a elementos de tuplas

- Al igual que en las listas, en las tuplas se puede acceder a sus elementos mediante un subíndice (base 0), o mas de uno si las tuplas están anidadas.

```
alumno=(1345,"Pedro Hernandez","Moreno 411", (15,"Junio",1998))

print(alumno[1])
print(alumno[3][1])
```

Pedro Hernandez
Junio

Recorrer Tuplas

- Las tuplas son secuencias, iterables por lo tanto pueden ser recorridas mediante un ciclo **for**

```
canalesTV = ("Canal 13", "Telefe", "America Noticias")  
  
for canal in canalesTV:  
    print(canal)
```

```
Canal 13  
Telefe  
America Noticias
```

Recorrer Tuplas

- Tambien pueden ser manipuladas mediante rebanada o slice

```
canalesTV = ("Canal 13", "Telefe", "America Noticias")
```

```
canal2=canalesTV[2:]
```

```
print(canal2)
```

```
('America Noticias',)
```

Desempaquetado de tuplas

- Los valores individuales de una tupla pueden ser recuperados asignando la tupla a las variables respectivas. Esto se llama **desempaquetar** la tupla.
- La cantidad de variables que se va a asignar el contenido de la tupla, debe ser coincidente; de lo contrario ocurrirá un error.

Comparación de tuplas

- Dos tuplas son **iguales cuando tienen el mismo tamaño y cada uno de sus elementos correspondientes tiene el mismo valor**
- Esta comparación se realiza elemento por elemento, lo que significa que, si las tuplas tienen el mismo numero de elementos y cada uno de ellos es igual en posición y valor, entonces se consideran iguales en términos de contenido.

Comparación de tuplas

```
print((3,5)==(33/11,2+3))
print((1,3)==(3,1))
print((3,3)==(3,3))
print((3,"3")==("3",3))
print((1,2)==(0,1,2))
```

```
True
False
True
False
False
```

Comparación de tuplas

- Para saber si una tupla es mayor o menor a otra, se realiza lo siguiente:
 - Si una tupla tiene mas elementos que otra, esta se considera mayor
 - Python compara el primer elemento, si son iguales continua al siguiente.

```
print((1,2,3)>(1,2,3,4))
#True ya que el tercer elemento 3 es menor a 4
print((1,2,3)<(1,2,4))

#Falso porque al comparar el primer elemento ya arroja falso
print([(8,13)<(0,21)])
```

Funciones y métodos

- Las funciones **len()**, **max()**, **min()** y **sum()** operan con tuplas igual que lo hacen con listas. Es importante destacar que para que max y min funciones la tupla tiene que tener elementos **homogeneos**
- También actual del mismo modo el operador in, y los métodos **index** y **count**

Funciones y métodos

```
canalesTV = ("Canal 13", "Telefe", "America Noticias")

canal2=canalesTV[2:]

print(len(canalesTV))
print(max(canalesTV))
print(min(canalesTV))
print(canalesTV.index("Telefe"))
print(canalesTV.count("America Noticias"))

if "Canal 13" in canalesTV:
    print("Se encuentra")
```

```
3
Telefe
America Noticias
1
1
Se encuentra
```

Funciones y métodos

- En las tuplas **no vamos a poder usar** los siguientes métodos de listas que trabajan sobre la lista.
 - **remove**
 - **pop**
 - **append**
 - **sort**

Conversión de tuplas

- Una lista puede ser convertida en una tupla mediante la función **tuple()**
- Una tupla puede ser convertida en lista con la función **list()**
- **No es recomendable estar continuamente convertir tuplas a listas o viceversa, para evitar sobrecargar el sistema.**

Conversión de tuplas - Ejemplo

```
lista=[1,2,3,4]  
tupla=tuple(lista)  
  
print(tupla)
```

```
(1, 2, 3, 4)
```

Aplicaciones

- Mediante el uso de tuplas es posible intercambiar valores entre variables sin necesidad de usar una variable auxiliar

a,b = b,a

- Python realiza lo siguiente:
 - Empaque los valores de la derecha en una tupla
 - Luego desempaque esa tupla y asigna esos valores a las variables de la izquierda

Aplicaciones

- Por lo general se prefiere usar tuplas cuando los elementos son heterogéneos, y listas cuando estos son homogéneos. **No solo en cuanto a tipo de dato sino tambien en significado.**
- Por ejemplo:
 - fecha=(dia,mes,anio)
- Los tres datos son enteros, pero significan diferentes cosas.

Práctica en clase

Realizar una función que reciba dos puntos en un plano de coordenadas, y calcule mediante el teorema de Pitágoras la distancia entre ambos



Resolución

```
def calcularDistancia(punto1,punto2):  
  
    x1,y1 = punto1  
    x2,y2=punto2  
  
    diferenciaX=x2-x1  
    diferenciaY=y2-y1  
  
    distancia= (diferenciaX**2 + diferenciaY**2)**0.5  
  
    return distancia  
  
#Programa Principal  
  
puntoA=(-34.559296008841464, -58.450331615342910)  
puntoB=(-34.60331083120916, -58.3815230070455)  
  
print(f"La distancia entre el punto A {puntoA} y el punto B {puntoB} es : {calcularDistancia(puntoA,puntoB)}")
```

Práctica en clase

Realizar programa que reciba dos fechas de nacimiento en formato YYYY-MM-DD pertenecientes a dos personas e indique quien es mayor que quien.



Resolución

```
nacimientoJuan=(1992,4,6)
nacimientoPedro=(1992,4,6)

if nacimientoJuan>nacimientoPedro:
    print("Pedro es mayor a Juan")
elif nacimientoPedro>nacimientoJuan:
    print("Juan es mayor que Pedro")
else:
    print("Tienen la misma edad")
```

Ventajas sobre el uso de listas

- **Seguridad:** las tuplas son inmutables, esto puede ser beneficioso para asegurarnos de que los datos no cambien accidentalmente.
- **Eficiencia:** las tuplas son mas eficientes que las listas en términos de uso de memoria y rendimiento, especialmente para operaciones simples como acceso y recuperación de elementos. Esto se debe a que las tuplas son estructuras de datos mas simples que no admiten operaciones de modificación, lo que las hace mas ligeras y rápidas en algunas situaciones

Resumen de la clase

- Tuplas
- Creación de tuplas
- Comparación de tuplas
- Ejercicios

Guía de tuplas

- Realizar la guía de tuplas

Muchas gracias!

Consultas?

