

# COL·LECCIONS

LLISTES, TUPLES, DICCCIONARIS



M<sup>a</sup> Belén Tortosa Pedrón



# TUPLES

- S'utilitzen per descriure un objecte com un agrupament de dades de diferents tipus.
- Serveix per agrupar, com si fossen un únic valor, diversos valors.
- Exemples:
  - Una data la podem representar com la terna dia (un número enter), més (una cadena de caràcters), i any (un número enter), i tindrem per exemple: (25, 'Maig', 1810).
  - Dades dels alumnes: NIF, nom i cognoms, per exemple: (123456789-D, 'Alicia', 'Hacker').
  - Punts en el pla: x, y, per exemple: (1, 2)
  - Es possible unir tuples: com dades dels alumnes volem guardar NIF, nom, cognom i data de naixement, per exemple : (123456789-D, 'Alicia', 'Hacker', (9, 'Juliol', 1988))



# TUPLES

- Una tupla és una llista **inmutable**. Una tupla no pot modificar-se de cap manera després de la seva creació i té la mida fixa. Són més lleugeres que una llista.
- Tot el que hem explicat sobre les llistes s'aplica també a les tuples, a excepció de la forma de definir-la ja que s'utilitzen parèntesis en lloc de claudàtors. El constructor de la tupla és la coma.


- Exemple: **variable ( 1, 2, 3 )**

- Si la tupla té només un element: **variable (1,)**

- Una Tupla buida: **variable=()**

- Exemple:

```
14_exemple_tuples.py
1  tupla= (1,2,3)
2  print(tupla)
3
4  t=(1, )
5  print(t)
6  print(type(t))
7
```



```
C:\WINDOWS\system32\cmd.exe
(1, 2, 3)
(1,)
<class 'tuple'>
```

```
15_exemple_tuples.py x
1  # Exemple simple de tupla
2  tupla = 12345, 54321, 'hola!'
3  print(tupla)
4  # Exemple de tuplas anidades
5  otra = tupla, (1, 2, 3, 4, 5)
6  print(otra)
7  # operació assignació de valors d'una tupla en variables
8  x, y, z = tupla
9  print('x= ', x, '\ny=', y, '\nz=', z)
10 print('tipus x= ', type(x))
11 print('tipus y= ', type(y))
12 print('tipus z= ', type(z))
```



```
C:\WINDOWS\system32\cmd.exe
(12345, 54321, 'hola!')
((12345, 54321, 'hola!'), (1, 2, 3, 4, 5))
x= 12345
y= 54321
z= hola!
tipus x= <class 'int'>
tipus y= <class 'int'>
tipus z= <class 'str'>
```



# ACCÉS A LES TUPLES

- Per accedir als elements d'una tupla, s'utilitzen els claudàtors `[]` per a tallar en porcions:
- Exemples:

```
tup1 = ('physics', 'chemistry', 1997, 2000);  
tup2 = (1, 2, 3, 4, 5 );  
  
print(tup1[1:], tup2[1:5])
```

```
C:\> Seleccionar C:\WINDOWS\system32\cmd.exe
```

```
('chemistry', 1997, 2000) (2, 3, 4, 5)
```



# DESEMPAQUETANT TUPLES

- Els valors individuals d'una tupla poden ser recuperats assignant la tupla a les variables respectives:

- Exemple:

```
15_exemple_tuples_a.py x
1 # Exemple simple de tupla
2 persona = 'Amadeus', 'Mozart'
3 print(persona)
4 # Desempaquetant:
5 nom,cognom=persona
6 print('\nNOM:',nom)
7 print('\nCOGNOM:',cognom)
8
C:\WINDOWS\system32\cmd.exe
('Amadeus', 'Mozart')
NOM: Amadeus
COGNOM: Mozart
```

- Si intentem desempaquetar un quantitat incorrecta de valors, obtindrem un error:

```
15_exemple_tuples_a.py x
1 # Exemple simple de tupla
2 persona = 'Amadeus', 'Mozart'
3 print(persona)
4 # Desempaquetant:
5 nom,cognom, edat=persona
6 print('\nNOM:',nom)
7 print('\nCOGNOM:',cognom)
8
C:\WINDOWS\system32\cmd.exe
('Amadeus', 'Mozart')
Traceback (most recent call last):
  File "M:\Curs2016-2017\codi\15_exemple_tuples_a.py", line 5, in <module>
    nom,cognom, edat=persona
ValueError: not enough values to unpack (expected 3, got 2)
```

- És possible extreure el seu valor utilitzant l'índex:

```
15_exemple_tuples_a.py x
1 # Exemple simple de tupla
2 persona = 'Amadeus', 'Mozart'
3
4 print(persona[1])
C:\WINDOWS\system32\cmd.exe
Mozart
```



# ELIMINAR ELEMENTS D'UNA TUPLA

- La eliminació d'elements individuals no és possible. Per a eliminar una tupla completa, només cal utilitzar la declaració **del**.
- Exemple:

```
tup = ('physics', 'chemistry', 1997, 2000)

print (tup)
del tup;
print ("After deleting tup : ")
print (tup)
```

```
C:\WINDOWS\system32\cmd.exe
('physics', 'chemistry', 1997, 2000)
After deleting tup :
Traceback (most recent call last):
  File "I:\Curs2016-2017\M3\AWS1\UF1\ejemplos\provatuples", line 12, in <module>
    print (tup)
NameError: name 'tup' is not defined
```



# DESEMPAQUETANT TUPLES

- Els elements no es poden modificar:

15\_exemple\_tuples\_a.py x

```
1 # Exemple simple de tupla
2 persona = 'Amadeus', 'Mozart'
3 print(persona[1])
4 persona[1]= 'Beethoven'
5 # Desempaquetant:
6 nom,cognom=persona
7 print('\nNOM:',nom)
8 print('COGNOM:',cognom)
```

C:\WINDOWS\system32\cmd.exe

Mozart

Traceback (most recent call last):

File "M:\Curs2016-2017\codi\15\_exemple\_tuples\_a.py", line 4, in <module>  
 persona[1]= 'Beethoven'

TypeError: 'tuple' object does not support item assignment



# COMPARACIÓ DE TUPLES

- Dues tuples són iguals quan tenen el mateix nombre d'elements i cadascun dels seus elements tenen el mateix valor:

```
15_exemple_tuples_b.py x
1 # Exemple comparació de tuples
2 print((1,2) == (3//2,1+1)) #True
3
4 print((6,1)==(6,2)) #False
5
6 print((6,1)==(6,1,0)) #False
```

- Per determinar si una tupla és menor que altra, s'utilitza l'ordre lexicogràfic. Si els elements en la primera posició d'ambdós tuples són diferents, ells determinen l'ordenació de les tuples:

```
15_exemple_tuples_b.py
1 # Exemple comparació de tuples
2
3 print((1,4,7)<(2,0,0,1)) #True
4
5 print((1,9,10)<(0,5)) #False
6
7
```

- Si els elements de la primera posició són iguals, llavors s'utilitza el valors següents per fer la comparació:

**La primera comparació és True ja que  $1 < 2$ .**  
**La segona comparació és False ja que  $1 > 0$ .**

- Si a una tupla se li acaben els elements per comparar abans que a altra, llavors es considera menor.

```
# Exemple comparació de tuples
print((6,1,8)<(6,2,8)) #True
print((6,1,8)<(6,0)) #False
```





# ITERACIÓ SOBRE TUPLES

- Com les llistes, les tuples són iterables:

```
15_exemple_tuples_c_iteracions.py x
# Exemple comparació de tuples

for valor in (6,1,8,6,2,8):
    print (valor**2)
```

```
36
1
64
36
4
64
```

- Es pot convertir una tupla en una llista utilitzant la funció `list`, i una llista en una tupla utilitzant la funció `tuple`:

```
a= (1,2,3)
b= [4,5,6]

c=list(a)
print(c, type(c))

d=tuple(b)
print(d, type(d))
```

```
Selecció C:\WINDOWS\system32\cmd.exe
[1, 2, 3] <class 'list'>
(4, 5, 6) <class 'tuple'>
```



# OPERACIONS BÀSIQUES AMB TUPLES

Expressió Python	Resultats	Descripció
<code>len ((1, 2, 3))</code>	3	Longitud
<code>(1, 2, 3) + (4, 5, 6)</code>	(1, 2, 3, 4, 5, 6)	Concatenació
<code>('¡Hola!') * 4</code>	('¡Hola!', '¡Hola!', '¡Hola!', '¡Hola!')	Repetició
<code>3 in (1, 2, 3)</code>	Cert	Afiliació
<code>for x in (1, 2, 3):     print (x)</code>	1 2 3	Iteració



# FUNCIONS INTERNES DE TUPLES

## Función con Descripción

### len (tupla)

Dona la longitud total de la tupla.

### max (tupla)

Retorna l'element de la tupla amb valor màxim.

### min (tupla)

Retorna l'element de la tupla amb valor mínim.



# UTILITZACIÓ DE TUPLES

- S'utilitzen per agrupar valors, generalment, conceptes del món real.
- Exemple:
  - Per representar punts en el pla:

```
15_exemple_tuples_d_exemple_Punts.py •
1 # Exemple punt en el pla
2
3 punt1= (1,2)
4 punt2=(8,9)
5 x1,y1=punt1
6 x2,y2=punt2
7 dx=x2-x1
8 dy=y2-y1
9 distancia=(dx**2+dy**2)**0.5
10
11 print('La distancia entre: ',punt1, ' i', punt2, 'es: ',distancia)
^^
```

C:\WINDOWS\system32\cmd.exe  
La distancia entre: (1, 2) i (8, 9) es: 9.899494936611665

- Per representar dates: S'agrupen amb l'any, el mes i el dia. L'avantatge de fer-ho en aquest ordre, permet poder fer correctament comparacions amb els operadors, <, >, ==, !=.

```
15_exemple_tuples_d_exemple_Dates.py •
1
2
3 avui=(2016,4,12)
4 ahir=(2016,4,11)
5 nadal=(2016,12,25)
6 any_nou=(2016,12,31)
7
8 if(avui>ahir):
9     print('It's True')
10
11 if (avui<nadal<any_nou):
12     print('I\'m feeling lucky')
13
```

Selecció C:\WINDOWS\system32\cmd.exe  
It's True  
I'm feeling lucky



# UTILITZACIÓ DE TUPLES

- Una tupla pot contenir altres tuples:

- Exemple: Una persona pot ser descrita pel seu nom, el seu NIF i la seva data de naixement.

- `persona = ( 'Wolfgang Amadeus', '123456789-D', ( 1756, 1, 27 ) )`

- Per recuperar només un dels valors de la tupla:

- `_,_,(,mes,_) = persona` # mes tindrà el valor 1

```
15_exemple_tuples_d_exemple_Persona.py
1
2
3 persona = ( 'Wolfgang Amadeus', '123456789-D', ( 1756, 1, 27 ) )
4 _,_,(,mes,_) = persona
5 print(' Mes: ',mes)
6
7
8
```

C:\WINDOWS\system32\cmd.exe  
Mes: 1

- Una taula de dades, generalment es representa com una tupla:

```
15_exemple_tuples_d_exemple_Alumnos.py
1
2
3 alumnos = [( 'Wolfgang','Amadeus', '125','INF'),
4             ( 'Ana','Aparicio', '345', 'ADM'),
5             ( 'Andreu','Miró','268','MEC')]
6 for nom, cognom, num, estudis in alumnos:
7     print ( ' ',nom,cognom,'estudia:',estudis)
8
```

C:\WINDOWS\system32\cmd.exe  
Wolfgang Amadeus estudia: INF  
Ana Aparicio estudia: ADM  
Andreu Miró estudia: MEC

Si el **cognom** i el **num** no fossin necessaris:

```
for nom,_,_,estudis in alumnos:
    print (nom,'estudia:',estudis)
```