

Lenguaje de Programación Python

Tuplas

Dr. Mario Marcelo Berón

Universidad Nacional de San Luis
Departamento de Informática
Área de Programación y Metodologías de Desarrollo de Software



Tuplas



Es una secuencia ordenada de cero o más referencias a objetos. Sintácticamente, una tupla es una lista de valores separados por comas. Aunque no es necesario, la convención dice que las tuplas se encierran entre paréntesis.

Ejemplo

```
'a', 'b', 'c', 'd', 'e'  
( 'a', 'b', 'c', 'd', 'e' )
```



Tuplas: Características



- Las tuplas son inmutables.
- Tienen que ser definidas antes de que la misma sea usada.
- El primer elemento de una lista está ubicado en la posición 0.
- Los elementos de una tupla aparecen, en general, encerrados con paréntesis.
 - ❶ (1, 3, 4, 5)
 - ❷ ('hola', 3, 4, 5, [100, 200])
- Los elementos se pueden acceder de acuerdo a una posición.
Si `t=(20,30,40,50)` entonces
 - ❶ `t[0]` es 20
 - ❷ `t[3]` es 50
- No son homogéneas.
 - ❶ ("hola", 1, 2, 3, [1, 'a'])



Tuplas: Características



- Una tupla es idéntica que una lista excepto por estas diferencias:
 - Se escriben con paréntesis en lugar de corchetes.
 - Son inmutables
- Si se tiene una tupla con un único elemento para se identifique que efectivamente es una tupla se tiene que poner entre paréntesis y con una coma al final.
- Se pueden convertir tuplas en listas y listas en tuplas usando los constructores *list()* y *tuple()* respectivamente.
- Python almacena referencias a los objetos. Cuándo el objeto es inmutable operaciones como la asignación se crea un nuevo objeto y se cambia la referencia pero cuando el objeto es mutable se crean alias.

Operaciones

Operación

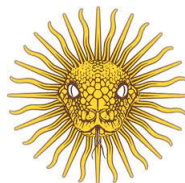
tupla[expresión]

Tarea

Accede al elemento de la lista indicado por expresión. Expresión debe retornar como valor un entero.

Ejemplo

```
t=(10,20,30)
t[0]
10
i=1
t[i+1]
30
```



Operaciones

Operación

$$var_0, var_1, var_2 = val_0, val_1, val_2$$

Tarea

El lado izquierdo es una tupla de variables, el lado derecho es una tupla de valores. Cada valor se asigna a su respectiva variable. Todas las expresiones del lado derecho se evalúan antes de que se realicen las asignaciones.

Ejemplo

a,b,c=1,2,3



Observación

Naturalmente, el número de variables de la izquierda y el número de valores a la derecha deben ser iguales.

Empaquetado de Tuplas

$$t = val_0, val_1, val_2, ..$$

Tarea

Los valores $val_0, val_1, val_2, ..$ se empaquetan en t .

Ejemplo

```
t=1,2,3
```

```
t
```

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

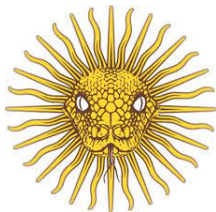


Desempaquetado de Tuplas

$$var_0, var_1, var_2, \dots = t$$

Tarea

Funciona para cualquier secuencia en el lado derecho de la asignación. Requiere que la cantidad de variables a la izquierda del signo igual sea el tamaño de la secuencia.



Ejemplo

```
t=(19,20)
```

```
a,b=t
```

```
a
```

```
19
```

```
b
```

```
20
```

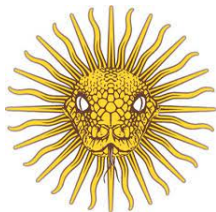


count

t.count(x)

Tarea

Retorna como resultado el número de veces que el objeto *x* aparece en la tupla *t*.



Ejemplo

```
t=(10,20,10)
t.count(10)
2
```



index

`t.index(x)`

Tarea

Retorna la posición de la ocurrencia de más a la izquierda del objeto `x` en la tupla `t`. Esta operación produce una excepción `ValueError` si `x` no está en la tupla.



Ejemplo

```
t=(20,30,40,50,60)
t.index(60)
4
```





Observaciones

- Las tuplas se pueden usar con los operadores `+` (concatenación), `*` (repliación) y `[]` (slice) e `in` y `not in` para probar membresía. También con los operadores `+=` y `*=`.
- Las tuplas se pueden comparar usando los operadores de comparación estándar. Las comparaciones se aplican ítem por ítem (y recursivamente para ítems anidados tales como tuplas dentro de tuplas).
- Los operadores `<` y `>` no significan estrictamente menor o mayor sino ocurre antes o ocurre después.

Tuplas Nombradas

Concepto

Se comporta como una tupla y tiene el mismo desempeño y características. Lo que este tipo de dato agrega es la posibilidad de referirse a los ítems en la tupla por nombre como así también por un índice.

Observaciones

```
Venta=collections.namedtuple("Venta","producto cliente fecha cantidad  
precio")
```

Observaciones

El módulo *collections* provee este tipo.



Ejemplo

```
import collections
Person = collections.namedtuple('Person', 'name age gender')
print ('Type of Person:', type(Person))
bob = Person(name='Bob', age=30, gender='male')
print ('Representation:', bob)
jane = Person(name='Jane', age=29, gender='female')
print ('Field by name:', jane.name)
print ('Fields by index:')
for p in [ bob, jane ]:
    print (' %s is a %d year old %s' % p)
```





Observaciones

- El primer argumento de `collections.namedtuple()` es el tipo de dato, el segundo es un string de nombres separados que se corresponden con los valores que la tupla tomará.
- La función retorna una clase personalizada (tipo de dato) que puede ser usada para crear tuplas nombradas.

