

Cristian Ruz Ruz Profesor Asistente Adjunto, Departamento de Ciencia de la Computación Facultad de Ingeniería – Pontificia Universidad Católica de Chile

Índice



Recomendaciones y cuidados

Introducción

- Para qué sirven las tuplas.
- Qué cuidados debemos tener al usar tuplas.

Ejemplos de uso de tuplas



1 Elementos ordenados



Inmutables

1

Elementos heterogéneos, ordenados, e inmutables

```
estudiantes = ['Aurora', 'Sebastián', 'Rafaela', 'Dario',
'Lisa', 'Almendra', 'Camilo']
puntos = [67, 56, 48, 21, 60, 38, 89]
temperaturas = [16.8, 35.2, 27.9, 19.2, 19.8, 24.8, 34.3]
dias = ['Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes',
'Sábado', 'Domingo']
```

- ¿Cuál es el puntaje de Rafaela?
- ¿Quién obtuvo el puntaje más alto?
- ¿Cuántos grados hubo el día viernes?
- ¿Qué días ha habido temperaturas mayores a 30 grados?

Buscar elemento(s) que cumple(n) una condición 2

Asociar elementos de listas diferentes

```
estudiantes = ['Aurora', 'Sebastián', 'Rafaela', 'Dario',
'Lisa', 'Almendra', 'Camilo']
puntos = [67, 56, 48, 21, 60, 38, 89]
```

¿Qué puntaje obtuvo Rafaela?

```
busqueda = "Rafaela"
for i in range(len(estudiantes)):
    if estudiantes[i] == busqueda:
        pos = i
print(f"El puntaje de {busqueda} es {puntos[pos]}")

El puntaje de Rafaela es 48
```

Buscar elemento(s) que cumple(n) una condición 3

Usar listas de tuplas para mantener información agrupada

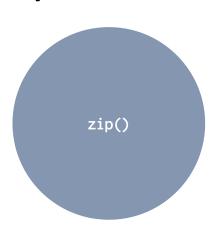
```
estud_puntos =[('Aurora', 67), ('Sebastián', 56), ('Rafaela',
48), ('Dario', 21), ('Lisa', 60), ('Almendra', 38), ('Camilo',
89)]
```

¿Qué puntaje obtuvo Rafaela?

```
busqueda = "Rafaela"
for elemento in estud_puntos:
    if elemento[0] == busqueda:
        tupla = elemento
print(f"El puntaje de {tupla[0]} es {tupla[1]}")

El puntaje de Rafaela es 48
```

Construyendo tuplas:



1

Construyendo listas de tuplas

```
estudiantes = ['Aurora', 'Sebastián', 'Rafaela', 'Dario',
'Lisa', 'Almendra', 'Camilo']
puntos = [67, 56, 48, 21, 60, 38, 89]
respuestas = ['a', 'c', 'c', 'b', 'd', 'd', 'a', 'c', 'e', 'b']
```

```
for elemento in zip(estudiantes, puntos):
    if elemento[0] == busqueda:
        tupla = elemento
print(f"El puntaje de {tupla[0]} es {tupla[1]}")
```

El puntaje de Rafaela es 48

2

Construyendo listas de tuplas

```
estudiantes = ['Aurora', 'Sebastián', 'Rafaela', 'Dario', 'Lisa',
   'Almendra', 'Camilo']
puntos = [67, 56, 48, 21, 60, 38, 89]
respuestas = ['a', 'c', 'c', 'b', 'd', 'd', 'a', 'c', 'e', 'b']
```

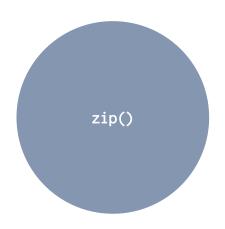
```
for elemento in zip(estudiantes, puntos, respuestas):
    print(elemento)
```

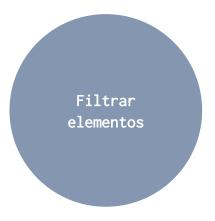
```
('Aurora', 67, 'a')
('Sebastián', 56, 'c')
('Rafaela', 48, 'c')
('Dario', 21, 'b')
('Lisa', 60, 'd')
('Almendra', 38, 'd')
('Camilo', 89, 'a')
```

```
print(list(zip(estudiantes, puntos, respuestas)))

[('Aurora', 67, 'a'), ('Sebastián', 56, 'c'), ('Rafaela', 48, 'c'),
  ('Dario', 21, 'b'), ('Lisa', 60, 'd'), ('Almendra', 38, 'd'),
  ('Camilo', 89, 'a')]
```

Construyendo tuplas:





1

Filtrar elementos usando zip y tuplas

```
temperaturas = [16.8, 35.2, 27.9, 19.2, 19.8, 24.8, 34.3]
dias = ['Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes',
'Sábado', 'Domingo']
```

¿Qué días hubo más de 30 grados?

```
calurosos = []
for dia in zip(temperaturas,dias):
    if dia[0] >= 30.0:
        calurosos.append(dia)
print(f"Días calurosos: {calurosos}")

Días calurosos: [(35.2, 'Martes'), (34.3, 'Domingo')]
```

1

Empaquetar valores en tuplas

```
fecha = ("Sábado", 15, "Febrero", 2020)
print(fecha)

('Sábado', 15, 'Febrero', 2020)
```

Packing y unpacking con tuplas

Desempaquetar valores de tuplas a variables individuales

```
(dia_sem, dia, mes, año) = fecha
print(dia_sem)
print(dia)
print(mes)
print(año)
Sábado
15
Febrero
2020
```

Empaquetar valores al terminar una función

```
def seleccionar(nombre, todos):
   for elemento in todos:
       if elemento[0] == nombre:
           return elemento ## retornamos una tupla
```

Desempaquetar valores al recibir el valor de retorno

valores

```
Retornar múltiples estudiantes = ['Aurora', 'Sebastián', 'Rafaela', 'Dario', 'Lisa',
                             'Almendra', 'Camilo']
                             puntos = [67, 56, 48, 21, 60, 38, 89]
                             respuestas = ['a', 'c', 'c', 'b', 'd', 'd', 'a', 'c', 'e', 'b']
                             res = seleccionar("Dario", zip(estudiantes, puntos, respuestas))
                             print(f"Resultado es {res}")
                             (e, p, r) = seleccionar("Dario", zip(estudiantes, puntos, respuestas))
                             print(f"El estudiante {e} tuvo {p} puntos y respondió {r}")
                             Resultado es ('Dario', 21, 'b')
                             El estudiante Dario tuvo 21 puntos y respondió b
```

Recomendaciones al usar tuplas

Crear tupla de tamaño uno

Notación con paréntesis no_tupla_de_uno = ("Solitario") print(type(no_tupla_de_uno)) print(no_tupla_de_uno) <class 'str'> Solitario

```
X Usando constructor
    de tuplas X

tupla_string = tuple("Solitario")
print(type(tupla_string))
print(tupla_string)

<class 'tuple'>
('S', 'o', 'l', 'i', 't', 'a', 'r', 'i', 'o')
```

```
tupla_de_uno = ("Solitario",)
print(type(tupla_de_uno))
print(tupla_de_uno)

<class 'tuple'>
('Solitario',)
```

1

Las tuplas son inmutables, pero podemos modificar listas

```
estudiante = ('Sebastián', [56])
estudiante[1].append(90)
print(estudiante)

('Sebastián', [56, 90])
```

Tuplas con elementos mutables

```
estudiantes = [('Aurora', [67, 70, 29]), ('Sebastián', [56]),
    ('Rafaela', [48, 50, 10, 68]), ('Dario', [21, 30])]

estudiantes[0][1].sort()
    estudiantes[1][1].append(90)
    estudiantes[2][1].pop()
    estudiantes[3][1].remove(21)
    print(estudiantes)

[('Aurora', [29, 67, 70]), ('Sebastián', [56, 90]), ('Rafaela', [48, 50, 10]), ('Dario', [30])]
```

¿Listas o tuplas?

list

- Mutable.
- Acceso eficiente a cada posición.
- Cuando vamos a coleccionar o modificar datos del mismo tipo.

tuple

- Inmutable.
- Acceso más eficiente a cada posición.
- Agrupaciones de datos semánticamente relacionados.

Síntesis

- Uso de tuplas.
- Datos heterogéneos y relacionados.
- Búsqueda, construcción con zip, empaquetamiento.
- Inmutabilidad.
- Recomendaciones.

Referencias bibliográficas

 Data Structures. The Python Tutorial, v3.8.2rc1. Recuperado de: https://docs.python.org/3/tutorial/datastructures.html

