Python

Tipo de secuencia:

TUPLAS

T U P L A S

- Al igual que las listas son un conjunto de valores separados por comas
  - Generalmente se encierran entre paréntesis

    ("Juan", 201555555, 20, "IC")

  - O pueden estar sin paréntesis

    "Juan", 201555555, 20, "IC"

- Tupla vacía: ()

- Operaciones comunes de secuencias:
  - Indexamiento
  - Membresía
  - Trozos
  - max, min, len
- Diferencia con las listas: son inmutables, es decir, no se pueden cambiar elementos particulares
- El uso de tuplas es más eficiente que las listas cuando se trata de algoritmos de búsquedas de datos

CREAR TUPLAS

```
>>> tupla="a","b","c","d","e"
>>> tupla=("a","b","c","d","e")
('a', 'b', 'c', 'd', 'e')
```

Para crear una tupla con un solo elemento, al final del elemento hay que poner “,” de lo contrario se tratará como un elemento simple (por ejemplo: un string).

```
>>> tup=("a",)
>>> type(tup)
<class 'tuple'>

>>> tup=("a")
>>> type(tup)
<class 'str'>
```

- Accediendo elementos de una tupla: por medio del índice

```
>>> tupla=("a","b","c","d","e")
>>> tupla[0]
'a'
>>> tupla[1:4]
('b', 'c', 'd')
```

TUPLAS SON INMUTABLES

- Los elementos de una tupla no se pueden modificar

- Error si se trata de modificar algun elemento

```
>>> tupla[0]="A"
Traceback (most recent call last):
  File "<pyshell#136>", line 1, in <module>
    tupla[0]="A"
TypeError: 'tuple' object does not support item assignment
```

- Para cambiar la tupla
  - Reemplazar todo el contenido de la tupla

```
>>> t = (2, 6, 7)
>>> t = (20, 35)
>>> print(t)
(20, 35)
```

  - Agregar elementos a la tupla: concatenar tuplas

```
>>> t = t + (10, 2, -57)
>>> print(t)
(20, 35, 10, 2, 57)
```

Recorrido de tuplas: usando for

```
>>> for x in (2,8,9):
        print(x)
2
8
9
```

Recorrido de tuplas: usando índices

```
>>> t = (2,8,9)
>>> i = 0
>>> largo = len(t)
>>> while i < largo:
        print(t[i]*2)
        i = i + 1
4
16
18
```

Membresía: operador in

```
>>> 5 in (8,5,9)
True

>>> 5 in (8,9)
False
```

PRÁCTICA

- Ejercicio: haga la función negativos_positivos que reciba una lista de números y retorne una tupla con los negativos y otra con los positivos. Ejemplo del funcionamiento:

```
>>> negativos_positivos([8, -10, 0, 50, 100, -1])
( (-10, -1), (8, 0, 50, 100) )
```

Haga 2 versiones:

1- usando for (negativos_positivos_for)

2- usando for (negativos_positivos_while)