Le5Anaconda

November 27, 2019

1 Esercitazione 5

1.0.1 Formattazione delle Stringhe

Questo metodo permette di formattare stringhe, ovvero inserire alcuni valori variabili all'interno di una stringa predefinita:

[ ]:
```
ore = 12
minuti = 30
secondi = 45
orario = "Sono le ore {}, {} minuti e {} secondi".format(ore,minuti,secondi)
print(orario)
```

Le {} sono dei "placeholder", cioè dei segnaposto (Ref: https://docs.python.org/3.7/library/string.html#format-examples)

[ ]:
```
ore = 12.0
minuti = 30.0
secondi = 45.0
orario = "Sono le ore {}, {} minuti e {} secondi".format(ore,minuti,secondi)
print(orario)
```

1.1 Strutture Dati

1.1.1 TUPLE

[ ]:
```
t = 'Francesco', 'Gallo', '177', '1977'
```

[ ]:
```
t
```

L'operatore che definisce le tuple è la virgola (,), anche se per evitare ambiguità la sequenze di elementi vengono spesso racchiuse tra parentesi tonde.

[ ]:
```
t = ('Francesco', 'Gallo', '177', '1977')
t
```

[ ]:
```
type(t)
```

[ ]:
```
len(t)
```

[ ]:
```
len(('Francesco', 'Gallo', '177', '1977'))
```

[ ]:
```
tVuota = () # Creazione di una tupla vuota
tVuota
type(tVuota)
```

[ ]:
```
len(tVuota)
```

1.1.2 Operazioni sulle TUPLE

Supportano le operazioni comuni a tutte le sequenze, come indexing, slicing, contenimento, concatenazione e ripetizione:

[ ]:
```
t = ('Francesco', 'Gallo', 177, '1977')
print(t[0])
```

[ ]:
```
print(t[:2])
```

[ ]:
```
'Gallo' in t
```

[ ]:
```
177 in t
```

Le tuple sono immutabili, quindi una volta create non è possibile aggiungere, rimuovere, o modificare gli elementi:

[ ]:
```
print(t)
t[0] = 'Pippo'
```

E' possibile usare funzioni e metodi comuni a tutte le sequenze: len() per contare gli elementi, min() e max() per trovare l'elemento più piccolo/grande (a patto che i tipi degli elementi siano comparabili), .index() per trovare l'indice di un elemento, e .count() per contare quante volte un elemento è presente nella tupla:

[ ]:
```
min(t)
```

[ ]:
```
print(t.index(177))
```

1.1.3 LISTE

[ ]:
```
vocali = ['a','e','i','o','u']
print(vocali)
```

[ ]:
```
type(vocali)
```

1.1.4 Operazioni sulle LISTE

E' possibile usare funzioni e metodi comuni a tutte le sequenze: len() per contare gli elementi, min() e max() per trovare l'elemento più piccolo/grande (a patto che i tipi degli elementi siano comparabili), .index() per trovare l'indice di un elemento, e .count() per contare quante volte un elemento è presente nella lista.

A differenza di tuple e stringhe che sono immutabili, le liste possono essere mutate. È quindi possibile assegnare un nuovo valore agli elementi, rimuovere elementi usando la keyword del, o cambiare gli elementi usando uno dei metodi aggiuntivi delle liste:

```
[ ]: alcuneLettere = ['b','c','d']
     vocali.append(alcuneLettere)
     print(vocali)
```

```
[ ]: vocali.extend(alcuneLettere)
     print(vocali)
```

```
[ ]: vocali.insert( 0, alcuneLettere)
     print(vocali)
```

```
[ ]: print(vocali.pop())
```

```
[ ]: print(vocali)
```

```
[ ]: alcuneLettere.reverse()
     print(alcuneLettere)
```

```
[ ]: copia = alcuneLettere.copy()
     print(copia)
```

```
[ ]: copia.clear()
```

```
[ ]: print(copia)
```

1.1.5 SET

I set (insiemi) vengono usati per rappresentare un insieme (mutabile) non ordinato di oggetti unici.

```
[ ]: numeri = set()
     print(numeri)
```

```
[ ]: numeri = {1,2,3,4,5,5,'Pippo'}
     print(numeri)
```

```
[ ]: nums = [1,2,3,4,4,4,5,6]
     numsSet = set(nums)
     print(numsSet)
```

```
[ ]: print(len(numsSet))
```

```
[ ]: print(min(numsSet))
```

```
[ ]: print(max(numsSet))
```

```
[ ]: 6 in numsSet
```

```
[ ]: 6 not in numsSet
```

```
[ ]: numsSet.add(8)
     print(numsSet)
```

```
[ ]: numsSet.remove(8)
     print(numsSet)
```

```
[ ]: numsSet.remove(8)
```

[ ]:
```
numsSet.discard(8)
```

[ ]:
```
numsSet.pop()
print(numsSet)
```

[ ]:
```
numsSet.pop()
print(numsSet)
```

[ ]:
```
numsSet.pop()
print(numsSet)
```

I set supportano anche una serie di operazioni tipiche degli insiemi. (Ref: https://docs.python.org/3/library/stdtypes.html#frozenset.union)

[ ]:
```

```