LABORATORIO DI PYTHON

DEFINIZIONE ED USO DELLE LISTE IN PYTHON

5 Aprile 2019



LISTE: DEFINIZIONE E OPERATORI

Sequenze ordinate e **mutabili** di elementi.

- · Elementi all'interno di parentesi quadre [1,2,3]
- · Lista vuota: []
- · Lista con un solo elemento: [42]

Operazioni comuni a tutte le sequenze (già visti):

- Operatore + per concatenare liste
- Operatore * per ripetere elementi (es. [1]*5 = ?)
- Operatore [] di selezione di singoli elementi (es. [1,2,3][0] = ?)
- · Operatore di slicing [:] e [::]
- · Operatore in e not in
- Funzione len()
- · Ciclo for per iterare sugli elementi della lista

LISTE: SEQUENZE MUTABILI

Essendo sequenze mutabili ammettono altre operazioni.

- Assegnamento: l[i] = 3
- · Assegnamento ad una sottolista: l[1:3] = [28, 30]
- · Inserimento di una sottolista:

```
>>> a_list = ["a", "d", "f"]
>>> a_list[1:1] = ["b", "c"]
>>> a_list
['a', 'b', 'c', 'd', 'f']
```

- Cancellazione di un elemento: del l[i] o di una sottolista: del a_list [1:5] .
- Inserire un elemento in fondo a una lista: a_list.append(x)
- · ... ma anche a_list + [x] (che però crea una copia...)
- Inserire un elemento in una posizione specifica i: a_list.insert(i, x)

LISTE: RIFERIMENTI E COPIE

· Riferimenti:

```
>>> x = [1, 2, 3]
>>> y = [1, 2, 3]
>>> x == y
True
>>> x is y
False
```

· Creare una copia di una lista. Differenza tra:

```
1 a = [1, 2, 3]
b = a
```

е

```
1 a = [1, 2, 3]
b = a.copy() #o equiv: b = a[:]
```

· Cosa restiuisce a is b nel primo caso? E nel secondo?

Si scriva una funzione Python sdoppia(lista) che, presa una lista (se non è una lista ritornare None), restituisce una nuova lista composta dagli stessi elementi di lista e nello stesso ordine, ma nella quale eventuali elementi contigui ripetuti compaiono una sola volta.

```
Esempio. Se A = [1,2,2,-1,0,0,2,2,2,2,3], sdoppia(A) deve restituire [1,2,-1,0,2,3].
```

Verificare che la lista A non sia stata modificata dopo l'esecuzione della funzione, per esempio con:

```
A = [1,2,2,-1,0,0,2,2,2,2,3]
print(A)
B = sdoppia(A)
print(A)
print(B)
```

Si scriva una funzione Python **sdoppiaSulPosto(lista)** che, presa una **lista**, la **modifica** così che eventuali elementi contigui ripetuti siano eliminati.

```
Esempio. Se A = [1,2,2,-1,0,0,2,2,2,2,3],
sdoppiaSulPosto(A) deve modificare A così che A ==
[1,2,-1,0,2,3]. La funzione restituisce None.
```

Verificare che la lista A **sia stata modificata** dopo l'esecuzione della funzione, per esempio con:

```
A = [1,2,2,-1,0,0,2,2,2,2,3]
print(A)
sdoppiaSulPosto(A)
print(A)
```

ESERCIZIO D'ESAME

Cosa stampa il seguente frammento di codice Python? Rispondere senza eseguirlo, usando eventualmente carta e penna.

```
1  T=[1,2]
2  L=T
3  print(T is L)
4  T.append(3)
5  print(T)
6  print(L)
7  L = L+[4]
8  print(T is L)
9  print(T)
10  print(L)
```

ESERCIZIO

Scrivere una funzione che presa una lista e un valore n appartenente a tale lista (verificarlo, se falso restituire due copie di lista) restituisca una coppia di liste, la prima contenente i valori minori o uguali a n e la seconda quelli maggiori di n.

N.B. NON utilizzare la funzione sort.

Scrivere una funzione che presa una lista controlla se al suo interno sono presenti valori uguali in posizioni successive, e in tal caso crea una nuova lista, in cui una serie di valori uguali contigui è sostituita da una sola occorrenza di quel valore seguita dal numero di occorrenze (solo nel caso in cui le occorrenze siano maggiori di una).

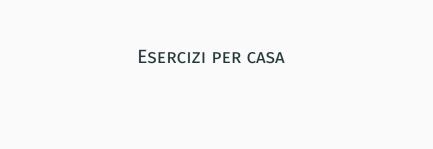
Esempio.

```
A = ['a', 'b', 'b', 'z', 'o', 'o', 'b', 'b', 'b', 'b', 'k']
print(valoriUguali(A))
['a', 'b', 2, 'z', 'o', 2, 'b', 4, 'k']
```

Scrivere una funzione che presa una lista controlli se al suo interno sono presenti valori uguali in posizioni successive, e in tal caso modifica la lista in questo modo: una serie di valori uguali contigui è sostituita da una sola occorrenza di quei valori seguita dal numero di occorrenze.

Esempio.

La funzione restituisce None.



- Scrivere una funzione che verifica se una lista è ordinata in modo crescente (supponiamo che la lista contenga oggetti ordinabili)
- Scrivere una funzione che presi come parametri due liste ordinate in modo crescente (verificarlo usando la funzione dell'esercizio precedente) ne crei una terza, anch'essa ordinata, dall'unione delle due. NON usare la funzione sort.
- 3. Scrivere una funzione ripetis(s) che, data una lista di numeri naturali s, restituisce una nuova lista in cui compaiono gli stessi elementi di s e nello stesso ordine, ma ciascun s[i] è ripetuto s[i] volte. Esempio. da [3,0,2] si deve ottenere [3,3,3,2,2].
- 4. Scrivere una funzione analoga alla precedente, ma che **modifica** la lista **s**.