QUIZ 6 - LISTES (3 points possibles)

Listes (1)

Rechercher un cours

On a

```
liste = [0, 1, 2, 3]
```

On veut **modifier** l'objet `liste` pour que sa valeur devienne `[0, 1, 4, 2, 3]`

Que faut-il faire ? (plusieurs réponses possibles)

- [x] `liste[2] = 4`
- [ ] `liste[2] = [4]`
- [ ] `liste.insert(2,4)` ✔
- [ ] `liste[2:3] = [4]`
- [x] `liste[2:2] = [4]` ✔

✖

EXPLANATION

La première formule remplace `2` par `4`, `liste` conserve 4 éléments et devient `[0, 1, 4, 3]`

La seconde, idem mais `liste` devient `[0, 1, [4], 3]`

La troisième est correcte

La quatrième est équivalente à la première, elle remplace `2` par `4`

La cinquième est correcte également

À nouveau on a

```
liste = [0, 1, 2, 3]
```

On souhaite *extraire* et retourner le premier élément `0`, tout en la retirant de la liste.

Plus précisément on veut affecter à la variable `suivant` la valeur `0` de telle sorte qu'après l'exécution, `liste` ne contienne plus que `[1, 2, 3]`

Que faut-il faire ? (plusieurs réponses possibles)

- [ ] `suivant = liste[0]`
- [ ] `suivant = liste.pop(0)` ✔
- [ ] `del liste[0]`
- [x] `suivant = liste[0]; del liste[0]` ✔
- [ ] `suivant = del liste[0]`

✖

EXPLANATION

La première formule retourne bien `0` mais ne modifie pas `liste`

La seconde est correcte

La troisième enlève bien `0` de `liste` mais ne retourne rien

La quatrième version fonctionne - quoi qu'assez peu pythonique

La dernière formule est syntaxiquement incorrecte; `del` est une instruction et ne peut pas être utilisée comme résultat d'une affectation.

On a cette fois

```
liste = [1, 0, 3, 2]
```

On veut trier la liste en ordre décroissant et **en place**, c'est-à-dire **sans dupliquer** la liste ni ses éléments.

Faut-il faire : (plusieurs réponses possibles)

- [ ] `liste.sort(reverse=True)` ✔
- [ ] `liste.sort()`
- [ ] `sorted(liste, reverse=True)`
- [x] `liste.sort(); liste.reverse()` ✔

✖

EXPLANATION

Il faut bien préciser `reverse=True` pour obtenir un tri *décroissant*

On ne veut pas dupliquer la liste initiale, c'est pourquoi ce n'est pas une bonne idée d'utiliser `sorted` .

La dernière formule fonctionne très bien également.

Vous avez utilisé 3 essais sur 3