ISM MPSI/PCSI, Informatique
Année 2023/2024

Corrigé du TP Informatique 3

Exercice 1

1, 2. On saisit une liste entre crochets.
3, 4, 5. L'opération `+` réalise la concaténation de listes et l'opération `*` réalise la duplication de listes.

Exercice 2

1. La variable `a` est de type `list`, de taille `len(a)`.
2. Le slicing permet d'extraire des sous-listes de `a` et aussi de la renverser.
3. L'instruction `in` teste la présence d'une élément dans une liste.

Exercice 3

1. L'instruction `list` convertit en liste.
2, 3. La conversion d'un `range` en liste permet d'en voir le contenu.

Exercice 4

On saisit :

```
n=10
b=list(range(n))
a=b+[n]+b[::-1]
print(a)
```

Exercice 5

Les listes sont *mutables* : on peut ajouter un élément en fin de liste avec `append`, supprimer le dernier élément en le renvoyant avec `pop`, affecter/concaténer avec `+=` et aussi modifier une composante existante ou même ordonner une liste avec la méthode `sort`.

Exercice 6

1,2. On saisit :

```
a=1023456789
b=list(str(a))
b.sort()
c=list("0123456789")
print(b==c)
```

Exercice 7

1. La duplication `b=a` n'est qu'une duplication d'étiquettes sur une même objet.
2. Avec l'indexation `[:]` ou l'instruction `list`, on crée des copies indépendantes.

Exercice 8

1. Pour une liste contenant une sous-liste, l'indexation `[:]` ou l'instruction `list` ne suffisent pas pour que les sous-listes soient des copies indépendantes.
2. Il faut effectuer une *copie en profondeur* pour éviter de simples copies d'étiquettes.

Exercice 9

1, 3. La duplication de sous-liste peut n'être qu'une duplication d'étiquettes sur un même objet `list`.
2, 4. On force la construction de sous-listes indépendantes.

Exercice 10

1. On saisit :

```
>>> [0]*5
[0, 0, 0, 0, 0]
>>> [0,1]*4
[0, 1, 0, 1, 0, 1, 0, 1]
>>> [1,0]*4+[1]
[1, 0, 1, 0, 1, 0, 1, 0, 1]
```

2. On saisit :

```
>>> list(range(6))
[0, 1, 2, 3, 4, 5]
>>> list(range(1,12,2))
[1, 3, 5, 7, 9, 11]
>>> list(range(10,-1,-2))
[10, 8, 6, 4, 2, 0]
```

3. On saisit :

```
>>> a=[[1]*10,[1]*10]
>>> a
[[1, 1, 1, 1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
```

On peut vérifier l'indépendance des sous-listes en modifiant la première par exemple :

```
>>> a[0].append(0)
>>> a
[[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0], [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
```

4. On saisit :

```
>>> from copy import deepcopy
>>> a=[[0],1]
>>> a+=deepcopy(a)
>>> a+=deepcopy(a)
>>> a
[[0], 1, [0], 1, [0], 1, [0], 1]
```

On peut vérifier l'indépendance des sous-listes en modifiant la première par exemple :

```
>>> a[0].append(2)
>>> a
[[0, 2], 1, [0], 1, [0], 1, [0], 1]
```

Exercice 11

On saisit :

```
a=[1,2,3]
b=[0]*2*len(a)
b[0::2]=a
b[1::2]=a
```