

COMMANDES À NE PAS UTILISER

Dans ce TP, nous n'utiliserons aucune des commandes suivantes :

- `in` (mais un `for ... in ...` pourra être utilisé)
- `count`
- `max` et `min`

OBJECTIFS

Après avoir, dans le cours sur les listes, donné les méthodes de base sur les listes et en avoir vu certaines applications (en particulier, un algorithme de tri de liste), l'objectif est maintenant un travail de recherche dans des listes.

1. Écrire une fonction prenant en arguments d'entrée une liste `L` et un élément `x`, puis renvoyant en sortie le nombre d'occurrences de `x` dans `L`.

```
1 def occurrence(L, x):
2     c=0
3     for e in L:
4         if e==x:
5             c=c+1
6     return c
```

2. Écrire une fonction prenant en arguments d'entrée une liste `L` et un élément `x`, puis renvoyant en sortie le rang de la première occurrence de `x` dans `L` si la liste `L` contient l'élément `x`, et `False` sinon.

```
1 def occ(L, x):
2     for k in range(0, len(L)):
3         if L[k]==x:
4             return k
5     return False
```

3. Écrire une fonction prenant en argument d'entrée une liste `L` puis renvoyant en sortie le maximum de `L` ainsi que son nombre d'occurrences.

```
1 def monmax(L):
2     maxi=L[0]
3     c=0
4     for x in L:
5         if x>maxi:
6             maxi=x
7             c=1
8         elif x==maxi:
9             c=c+1
10    return maxi, c
```

4. Écrire une fonction prenant en argument d'entrée une liste `L` puis renvoyant en sortie le maximum de `L` ainsi que le rang de sa première occurrence.

```
1 def monmax(L):
2     maxi=L[0]
3     for k in range(0, len(L)):
4         if L[k]>maxi:
5             maxi=L[k]
6             rang=k
7     return maxi, rang
```

5. Écrire une fonction prenant en argument d'entrée une liste `L` (contenant au moins deux éléments distincts) puis renvoyant en sortie le second maximum de cette liste.

```
1 def mes2max(L):
2     if L[0]<L[1]:
3         maxi1, maxi2=L[1], L[0]
4     else:
5         maxi1, maxi2=L[0], L[1]
6     for k in L[2:len(L)]:
7         if k>maxi1 and k>maxi2:
```

PETITE REMARQUE

Pour l'ensemble de ces questions, l'algorithme proposé ne devra parcourir la liste qu'une seule fois.

```
8     maxi1, maxi2=k, maxi1
9     elif maxi2<k<maxi1: #on met maxi2<k<=maxi1 si on compte les répétitions du max
10        maxi2=k
11    return maxi2
```

6. Écrire une fonction prenant en argument d'entrée une liste L (contenant au moins deux éléments) puis renvoyant en sortie les deux éléments les plus proches.

```
1 def proches(L):
2     ecart=abs(L[1]-L[0])
3     a,b=L[0],L[1]
4     for i in range(0, len(L)):
5         for j in range(i+1, len(L)):
6             if abs(L[i]-L[j])<ecart:
7                 ecart=abs(L[i]-L[j])
8                 a,b=L[i],L[j]
9     return a,b
```