

# Corrigé sujet 01 - Année : 2023

[Sujet 01 - 2022 ↴](#)

## 1. Exercice 1



```

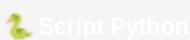
1 def verifie(tab):
2     for i in range(len(tab)-1):
3         if tab[i] > tab[i+1]:
4             return False
5     return True

```



- On parcours le tableau par indice, si un élément est supérieur à son successeur alors on renvoie `False`.
- Si on atteint la fin du tableau, on renvoie `True` (tous les éléments sont bien inférieurs à leur successeur)
- On parcours jusqu'à l'*avant-dernier* (le dernier n'a pas de successeur)

## 2. Exercice 2



```

1 urne = ['A', 'A', 'A', 'B', 'C', 'B', 'C', 'B', 'C', 'B']
2
3 def depouille(urne):
4     resultat = {} #(1)
5     for bulletin in urne:
6         if bulletin in resultat: #(2)
7             resultat[bulletin] = resultat[bulletin] + 1
8         else:
9             resultat[bulletin] = 1
10    return resultat
11
12 def vainqueur(election):
13     vainqueur = ''
14     nmax = 0
15     for candidat in election:
16         if election[candidat] > nmax : #(3)

```

```
17     nmax = election[candidat]
18     vainqueur = candidat
19     liste_finale = [nom for nom in election if election[nom] == nmax] #(4)
20     return liste_finale
```

1. Initialisation à un dictionnaire vide.
2. Si la clé existe dans le dictionnaire on incrémente sa valeur de 1, sinon on ajoute cette clé avec la valeur 1 (c'est le premier vote pour ce groupe)
3. Algorithme classique de recherche du maximum en parcourant toutes les valeurs
4. On construit donc ici par compréhension la liste des candidats (car il peut y en avoir plusieurs) ayant le nombre de votes maximales

### Attention

La variable `vainqueur` définie à la ligne 18 (et qui porte le même nom que la fonction) peut laisser penser qu'il y en a un seul ! Alors qu'on construit justement une liste pour gérer les cas d'ex-aequo, cette variable n'a en fait aucune utilité.