

# Corrigé sujet 10 - Année : 2023

[Sujet 10 - 2022 ↴](#)

## 1. Exercice 1



```

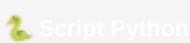
1 def maxliste(tab):
2     maxi = tab[0]
3     for elt in tab:
4         if elt > maxi:
5             maxi=elt
6     return maxi

```

### Commentaires

L'énoncé précise que la liste est non vide, on peut donc se permettre d'initialiser le maximum courant avec le premier élément de la liste.

## 2. Exercice 2



```

1 class Pile:
2     """ Classe définissant une pile """
3     def __init__(self):
4         self.valeurs = []
5
6     def est_vide(self):
7         """Renvoie True si la pile est vide, False sinon"""
8         return self.valeurs == []
9
10    def empiler(self, c):
11        """Place l'élément c au sommet de la pile"""
12        self.valeurs.append(c)
13
14    def depiler(self):
15        """Supprime l'élément placé au sommet de la pile, à condition qu'elle soit non vide"""
16        if self.est_vide() == False:
17            self.valeurs.pop()
18
19
20    def parenthesage (ch):
21        """Renvoie True si la chaîne ch est bien parenthésée et False sinon"""

```

```
22     p = Pile()
23     for c in ch:
24         if c == "(": #(1)
25             p.empiler(c)
26         elif c == ")": #(2)
27             if p.est_vide():
28                 return False
29             else:
30                 p.depiler()
31     return p.est_vide() #(3)
```

1. On suit l'algorithme donné dans l'énoncé : si on rencontre une parenthèse ouvrante alors on l'empile
2. Si c'est une parenthèse fermante, on dépile dans le cas où la pile est vide, l'expression est mal parenthésée.
3. Si à la fin du parcours la pile n'est pas vide, l'expression est mal parenthésée.

### Remarque

L'utilisation d'une pile pour vérifier le bon parenthésage est surtout utile lorsqu'il y a plusieurs types de parenthèses ouvrantes et fermantes : () mais aussi {}, [] ...