

Corrigé sujet 02 - Année : 2023

[Sujet 02 - 2022 ↴](#)

1. Exercice 1



```

1 def indices_maxi(tab):
2     maxi, indices_maxi = tab[0], [0]
3     for i in range(1, len(tab)):
4         if tab[i] > maxi:
5             maxi = tab[i]
6             indices_maxi = [i]
7         elif tab[i] == maxi:
8             indices_maxi.append(i)
9     return maxi, indices_maxi

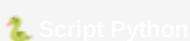
```



On parcourt la liste par indice:

- lignes 4 à 6 : si on trouve un élément plus grand (strictement) que le maximum alors cet élément est le nouveau maximum (et la liste des indices du maximum ne contient que l'indice de cet élément)
- lignes 7 et 8 : si on trouve un élément égal au maximum alors on rajoute son indice à la liste des indices du maximum

2. Exercice 2



```

1 def positif(pile):
2     pile_1 = list(pile) #1
3     pile_2 = [] #2
4     while pile_1 != []:
5         x = pile_1.pop() #3
6         if x >= 0:
7             pile_2.append(x) #4
8     while pile_2 != []:
9         x = pile_2.pop()
10        pile_1.append(x) #5
11    return pile_1

```

1. On réalise dans `pile_1` une copie indépendante de la pile
2. `pile_2` est initialement vide, on y empilera les éléments positifs de `pile_1`
3. On retire successivement les éléments `x` de `pile_1`
4. S'ils sont positifs, on les empile dans `pile_2`
5. On dépile tous les éléments de `pile_2` dans `pile_1` afin qu'ils soient dans l'ordre initial

Attention

1. Bien comprendre que le sujet se limite à l'interface habituelle d'une pile (`empile` avec `append`, `depile` avec `pop` et `est_vide` avec `==[]`). On pourrait trier les éléments positifs d'une liste bien plus simplement (par exemple par compréhension).
2. On rappelle que `x = pile_1.pop()` possède deux effets : supprimer le dernier élément de `pile_1()` et affecte sa valeur à `x`.