NSI - Première

Algorithmique - Tris, bilan

1. Reconnaitre un algorithme de tri

```
def tri_1(tableau: list) -> None:
    for i in range(len(tableau)):
        j = i
        while j > 0 and tableau[j] < tableau[j - 1]:
            tableau[j], tableau[j - 1] = tableau[j - 1], tableau[j]
            j -= 1
```

```
def tri_2(tableau: list) -> None:
    for i in range(len(tableau)):
        mini = i
        for j in range(i, len(tableau)):
            if tableau[j] < tableau[mini]:
                mini = j
        tableau[i], tableau[mini] = tableau[mini], tableau[i]
```

1. Identifier parmi les fonctions python précédentes le tri par insertion et le tri par sélection
2. Ces fonctions renvoient-elles quelque chose ? Quel est l'objet qui est "trié" après leur exécution ? Comment qualifie-t-on de telles fonctions ?

2. Faire tourner un algorithme de tri à la main

1. Identifier les lignes correspondant à la boucle principale et celles correspondant à la boucle interne des fonctions précédentes.
2. Indiquer dans un tableau l'état des variables (`tableau, i, j` et éventuellement `mini`) à chaque fois *qu'on sort* de la boucle interne lorsqu'on exécute ces fonctions sur le tableau `[3, 1, 7, 5]`.

3. La fonction `est_triee`

Proposer une fonction Python prend un tableau non vide d'entiers et renvoie un booléen vrai si le tableau reçu est trié par ordre croissant et faux sinon.

```
>>> est_trie([1, 3, 5, 7])
True
>>> est_trie([3, 1, 5, 7])
False
```

4. Amélioration de `tri_1`.

Vous avez bien sûr reconnu en `tri_1` le tri par insertion. La version proposée plus haut n'est pas optimale, ainsi qu'on s'en rend compte lors d'un déroulé détaille (cf exercice précédent).

Proposer une version améliorée qui evite une grande partie des échanges de la boucle interne.