Exercice 1 :

On veut trier par ordre croissant les notes d'une évaluation qui sont des nombres entiers compris entre 0 et 10 (inclus).

Ces notes sont contenues dans un tableau `notes_eval` (type `list`).

Écrire une fonction `effectif_notes` prenant en paramètre le tableau `notes_eval` et renvoyant un tableau de longueur 11 tel que la valeur d'indice `i` soit le nombre de notes valant `i` dans le tableau `notes_eval`.

Écrire ensuite une fonction `notes_triees` prenant en paramètre le tableau des effectifs des notes et renvoyant un tableau contenant les mêmes valeurs que `notes_eval` mais triées dans l'ordre croissant.

Exemple :

```
>>> notes_eval = [2, 0, 5, 9, 6, 9, 10, 5, 7,
                  9, 9, 5, 0, 9, 6, 5, 4]

>>> eff = effectif_notes(notes_eval)
>>> eff
[2, 0, 1, 0, 1, 4, 2, 1, 0, 5, 1]

>>> notes_triees(eff)
[0, 0, 2, 4, 5, 5, 5, 5, 6, 6, 7, 9, 9, 9, 9, 9, 10]
```

Exercice 2 :

On considère la fonction `insere` ci-dessous qui prend en argument un tableau `tab` d'entiers triés par ordre croissant et un entier `a`. Cette fonction crée et renvoie un nouveau tableau `tab` d'entiers triés par ordre croissant.

Cette fonction crée et renvoie un nouveau tableau à partir de celui fourni en paramètre en y insérant la valeur a de sorte que le tableau renvoyé soit encore trié par ordre croissant. Les tableaux seront représentés sous la forme de listes Python.

```
def insere(tab, a):
    """
    Insère l'élément a (int) dans le tableau tab (list)
    trié par ordre croissant à sa place et renvoie le
    nouveau tableau.
    """
```

Compléter la fonction `insere` ci-dessus.

Exemples :

```
>>> insere([1, 2, 4, 5], 3)
[1, 2, 3, 4, 5]
>>> insere([1, 2, 7, 12, 14, 25], 30)
[1, 2, 7, 12, 14, 25, 30]
>>> insere([2, 3, 4], 1)
[1, 2, 3, 4]
>>> insere([], 1)
[1]
```

Exercice 3 :

Sur le réseau social TipTop, on s'intéresse au nombre de « like » des abonnés. Les données sont stockées dans des dictionnaires où les clés sont les pseudos et les valeurs correspondantes sont les nombres de « like » comme ci-dessous :

```
{ 'Bob': 102, 'Ada': 201, 'Alice': 103, 'Tim': 50 }
```

Écrire une fonction `max_dico` qui :

- prend en paramètre un dictionnaire `dico` non vide dont les clés sont des chaînes de caractères et les valeurs associées sont des entiers ;
- et qui renvoie un tuple dont :
  - la première valeur est la clé du dictionnaire associée à la valeur maximale ;
  - la seconde valeur est la première valeur maximale présente dans le dictionnaire.

Exemples :

```
>>> max_dico({ 'Bob': 102, 'Ada': 201, 'Alice': 103, 'Tim': 50 })
('Ada', 201)
>>> max_dico({ 'Alan': 222, 'Ada': 201, 'Eve': 222, 'Tim': 50 })
('Alan', 222)
```