Activité NSI : Programmation de base en Python (préparation du projet "codage de *Huffman*")

On étudie des tableaux indicés (`list`) de longueur 2, qu'on appellera de façon schématique *couple* :
- le premier élément de ces tableaux est une chaine de caractères (`str`)
- le deuxième élément de ces tableaux est un nombre entier positif (`int`) appelé "*poids*"

Quelques exemples de *couples* :
```
["r" , 1]
["a" , 5]
["l" , 4]
```

On étudie également des tableaux indicés (`list`), qu'on appellera de façon schématique *conteneurs,* contenant uniquement les *couples* précédemment décrits, classés par ordre de *poids* décroissant.

Exemple d'un *conteneur* : `[["a" , 5],["l" , 4],["r" , 1]]`

Défi n°1 :

Ecrire une fonction `inserer_au_bon_endroit(couple, conteneur)` :
- prenant en paramètre :
    - un *couple* : tableau indicé (`list`) de longueur 2, précédemment décrit.
    - un *conteneur* : tableau indicé (`list`) précédemment décrit
- qui insère le tableau *couple* dans le tableau *conteneur* de manière à ce que les *couples* restent classés par ordre de *poids* décroissant.
- effet de bord : le tableau *conteneur* est modifié
- cette fonction ne renvoie rien

Outil :
La *méthode* `nom_de_la_liste.insert(position,element)` : permet d'insérer un élément `element` dans la liste `nom_de_la_liste` à la position `position`

Exemple :
```
>>> L = ["A","B","C","D"]
>>> L.insert(2,"Z")
>>> L
['A', 'B', 'Z', 'C', 'D']
```

Exemple de fonctionnement :
```
>>> conteneur1 = [["a" , 5],["l" , 4],["r" , 1]]
>>> couple1 = ["z" , 3]
>>> inserer_au_bon_endroit(couple1, conteneur1)
>>> conteneur1
[["a" , 5],["l" , 4],["z" , 3],["r" , 1]]
>>> couple2 = ["p" , 8]
```

```
>>> inserer_au_bon_endroit(couple2, conteneur1)
>>> conteneur1
[["p" , 8],["a" , 5],["l" , 4],["z" , 3],["r" , 1]]
>>> couple3 = ["k" , 0]
>>> inserer_au_bon_endroit(couple3, conteneur1)
>>> conteneur1
[["p" , 8],["a" , 5],["l" , 4],["z" , 3],["r" , 1],["k" , 0]]
```

Remarque n°1:
Le tableau *conteneur* dans lequel est inséré le tableau *couple* peut être vide.

```
>>> conteneur2 = []
>>> couple1 = ["z" , 3]
>>> inserer_au_bon_endroit(couple1, conteneur2)
>>> conteneur1
[["z" , 3]]
```

Remarque n°2:
Le tableau *couple* est inséré à la première position dans les couples du tableau *conteneur* sont correctement classés.

```
>>> conteneur1 = [["a" , 5],["l" , 4],["r" , 1]]
>>> couple4 = ["t" , 1]
>>> inserer_au_bon_endroit(couple4, conteneur1)
>>> conteneur1
[["a" , 5],["l" , 4],["t" , 1],["r" , 1]]
```

Défi n°2 :

On dispose d'un dictionnaire dont chaque *clé* (`str`) est associée à la *valeur correspondante* (`int`).

On souhaite obtenir un *conteneur* (qui n'a pas été précédemment déclaré) contenant des *couples* [*clé, valeur correspondante*] , issus du dictionnaire et classés par *valeur correspondante* décroissante.
Par soucis de cohérence entre équipes, il est important de travailler avec les clés du dictionnaire qui auront été classées par `sorted(dico_occurrence.keys())`

Exemple de fonctionnement :

```
>>> dico1 = {"t":1,"r":1,"a":5,"l":4}
>>> liste_de_couples(dico1)
[['a', 5], ['l', 4], ['t', 1], ['r', 1]]
```

Compléter alors la fonction `liste_de_couples(dico)` qui répond à cette description.

```
def liste_de_couples(dico):
    conteneur = ...
    liste_des_cles = sorted(dico_occurrence.keys())
    for cle in ... :
        ...([cle,...],conteneur)#on exploite la fonction du défi n°1
    return conteneur
```