Du tri pour terminer

1. Écrire un algorithme qui, étant 2 tableaux non triés A et B, vérifie si tous les éléments de A appartiennent à B. Quelle est sa complexité en nombre de comparaisons entre éléments de tableaux.
   Améliore-t-on cette complexité si l'un des tableaux est trié et si oui, précisez lequel.
   Améliore-t-on également cette complexité si les 2 tableaux sont triés ?
   Est-il intéressant du point de vue complexité de trier les tableaux pour résoudre ce propblème ?
2. A et B étant 2 tableaux, il s'agit de vérifier si l'ensemble des valeurs des éléments de A est inclus dans l'ensemble des valeurs des éléments de B mais en comptant aussi le nombre d'occurrences. Lorsqu'une valeur apparaît plusieurs fois dans A, son nombre d'occurrences dans B doit être supérieur à son nombre d'occurrences dans A. Ainsi on dira que A est inclus dans B si et seulement si
   $\forall i \in [0..taille(A)[$, `nbOccurrences(A[i],A)`$\leq$ `nbOccurrences(A[i],B)` où `nbOccurrences(x,T)` désigne le nombre d'éléments du tableau T dont la valeur est x.
   Écrivez un algorithme testant l'inclusion de A dans B. Quelle est sa complexité ?
3. Le *Tri par fusion* est un autre algorithme de tri appliquant le principe *Diviser pour résoudre*. Son principe pour trier un tableau à n éléments est :
   trier le sous-tableau constitué des $n/2$ premiers éléments
   trier le sous-tableau constitué des $n/2$ derniers éléments
   fusionner les éléments des 2 sous-tableaux triés.
   Ecrivez et prouvez cet algorithme. Donnez sa complexité dans le pire et dans le meilleur des cas.
4. Le problème du *Drapeau Hollandais* consiste à trier un tableau dont les éléments ne peuvent prendre que 3 valeurs que l'on notera R, V, B avec l'ordre $R < V < B$. Donnez un algorithme linéaire pour ce problème.
5. On s'intéresse au problème de tri **particulier** suivant : les éléments à trier prennent leur valeur dans un intervalle fini et *petit* connu a priori. On prendra comme exemple le tri d'un tableau T de notes à valeurs entières comprises entre 0 et 20. Pour ce problème on peut utiliser un tableau supplémentaire $NbNotes$ indicé de 0 à 20, tel que pour tout indice i $NbNotes[i]$ est le nombre d'éléments de T égaux à i.
   Ecrivez l'algorithme et donnez sa complexité.