Université de Bordeaux
Département de Licence
4TPM205U : Algorithmique des tableaux

Deuxième TP noté - 2 mai 2017
groupes A3/A4/A5/B5
Durée 40 minutes

Tout document personnel autorisé. Votre voisin n'est pas un document.

Nom : **Prénom :**

Signature :

Cette feuille contient 4 exercices.

Consignes à lire attentivement :

- Placez vous dans un répertoire contenant le fichier `bibTableau.py`.
- Avec `idle3` créer un fichier nomPrenom.py où vous remplacerez nomPrenom par les vôtres. À la première ligne de ce fichier écrivez votre nom et votre prénom (en commentaire). Dans ce fichier vous écrirez les fonctions démandées ci-dessous ainsi que les appels permettant de les tester.
- Les tests font partie du travail qui est évalué, vos instructions de test doivent donc impérativement figurer dans votre fichier python (tester une fonction signifie ici vérifier son bon fonctionnement sur au moins un exemple). Il faut donc appeler la fonction et afficher son résultat.
- Si une fonction que vous avez écrite ne fonctionne pas, mettez-la en commentaire mais ne l'effacez pas (on pourrait tout de même vous accorder des points si l'idée est pertinente).

En fin de séance :

1. Envoyer le fichier par e-mail à votre enseignant(e) de TD, dont l'adresse sera écrite au tableau.
2. Avant de quitter la salle :
   - Se présenter à l'enseignant(e) pour vérifier que l'e-mail est arrivé.
   - Remettre à l'enseignant(e) cette feuille signée (elle attestera de votre présence à l'épreuve).
   - Ne sortez pas de la salle sans l'accord de l'enseignant(e).
3. Garder le fichier contenant votre travail **sans le modifier**. Il pourrait vous être redemandé en cas de problème.

Exercice 1

On considère la suite définie par récurrence de la façon suivante :

$$\begin{cases} u_0 & = a \\ u_{n+1} & = \begin{cases} u_n - 1 & \text{si } n \text{ est pair,} \\ 3u_n & \text{sinon.} \end{cases} \end{cases}$$

où a est un entier naturel strictement positif.
Écrire une fonction **récursive** `suiteRec(a,n)` qui calcule le terme d'indice $n \geq 0$ de cette suite lorsque le premier terme u_0 est égal à a. Appeler la fonction pour $a = 2$ et $n = 10$ (on doit obtenir 123).

Exercice 2

Dans un tableau de n entiers, on souhaite trier uniquement les éléments compris entre deux indices (non négatifs) `borneInf` et `borneSup` inclus. Ces deux indices seront passés comme arguments à une fonction `triIntervalle(t, n, borneInf, borneSup)` qui mettra en oeuvre l'algorithme du **tri insertion**.
Si `borneSup`$\geq n$, la fonction doit trier les éléments d'indice supérieur ou égal à `borneInf`.
Tester votre fonction sur un tableau généré aléatoirement.

Exercice 3

Étant donné deux tableaux d'entiers triés dans l'ordre décroissant et **sans doublons**, on souhaite obtenir une suite triée décroissante **sans doublons**. Adapter la fonction `fusion(t1, n1, t2, n2, t3)` étudiée au cours du semestre pour obtenir ce résultat dans un troisième tableau passé aussi en paramètre (on supposera que le troisième tableau a une taille suffisante). La fonction retournera le nombre d'éléments de la suite ainsi obtenue.
Exemple :
la fusion sans doublons de [9, 7, 4, 3, 2] et [10, 7, 3, 1] donnera [10, 9, 7, 4, 3, 2, 1]. Pour ces données la fonction renverra 7.

Exercice 4

Étant donné deux tableaux `t1` et `t2` contenant respectivement `n1` et `n2` nombres entiers, on veut écrire une fonction qui renvoie le nombre d'élements distincts qui sont à la fois dans `t1` et dans `t2`. Il s'agit donc de vérifier si les éléments de `t1` sont (ou pas) dans `t2`, et les compter.
Pour l'instant on suppose que les tableaux sont sans doublons.

1. Écrire une fonction `CptInterV1(t1, n1, t2, n2)` de complexité $\mathcal{O}(n1 * n2)$.
2. Il est important de remarquer que vous avez déjà programmé des algorithmes de tri et la vérification (qu'un élément appartient à un tableau trié) au cours du semestre. En utilisant ces connaissances, écrire une fonction `CptInterV2(t1, n1, t2, n2)`. Remarque : on peut améliorer les performances de l'algorithme en ne triant que le plus petit des deux tableaux.

Question bonus : On suppose maintenant que les tableaux peuvent contenir des doublons.

3. Écrire une fonction `CptInterV3(t1, n1, t2, n2)` qui en tient compte. Ainsi si `t1 =[7,3,7]` et `t2=[6,-1,6,7]`, l'appel `CptInterVersion3(t1,3,t2,4)` devra renvoyer 1 (et pas 2) car il y a un seul élément en commun : 7.