Travaux Pratiques d'algorithmique n°7

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

Listes chaînées triées

Le but de ce TP est de manipuler les listes chaînées triées. Nous considérons des listes chaînées d'entiers.

Dans tout ce qui suit, nous appelons simplement liste une liste chaînées d'entiers implémentée par :

```
typedef struct _cellule
        { int Val;                    /* donnee stockee : un entier.       */
          struct _cellule * PtrSuiv; /* pointeur sur la cellule suivante. */
        } Cellule;                    /* definition d'un nouveau type.     */

typedef Cellule * Liste;              /* definition d'un nouveau type.     */
```

▸ **Exercice 1.** *Écrire les fonctions suivantes :*

- `Liste Insere( Liste, int );` *qui insère un élément dans une liste triée ;*
- `int Recherche( Liste, int );` *qui recherche un élément dans une liste triée.*

▸ **Exercice 2.** *Écrire les fonctions suivantes :*

- `Creation( Liste * );` *qui crée une liste triée à partir d'entiers lus sur l'entrée standard ;*
- `void AfficheOrdre( Liste );` *qui affiche dans l'ordre de chaînage une liste triée ;*
- `void AfficheInverse( Liste );` *qui affiche dans l'ordre inverse de chaînage une liste triée ;*

▸ **Exercice 3.** *Écrire une fonction* `void Libere(Liste * );` *qui libère la place mémoire prise par une liste.*

▸ **Exercice 4.** *Écrire une fonction qui crée une liste à partir d'entiers lus dans un fichier, et une fonction qui sauvegarde une liste dans un fichier. Ces fonctions appelleront les fonctions d'affichage et de saisie écrites précédemment, qu'on modifiera si besoin.*

▸ **Exercice 5.** *Écrire une fonction* `int Suppression( Liste*, int );` *qui supprime la première occurrence de l'entier dans la liste.*

Écrire une fonction qui supprime tous les éléments égaux dans une liste triée. Par exemple, la liste `(1 1 1 3 4 4 5 5 5)` *sera transformée en* `(1 3 4 5)`.

▸ **Exercice 6.**

- *Écrire une fonction* `Construct` *prenant en argument un entier* `N` *et qui retourne la liste des entiers compris entre 2 et* `N` *(dans l'ordre croissant) ;*
- *écrire une fonction* `Elimine` *prenant en argument une liste d'entiers* `l` *et un entier* `n` *et qui supprime dans* `l` *tous les multiples de* `n` *;*

On veut maintenant implémenter sur des listes chaînées l'algorithme du crible d'Eratosthène. Cet algorithme permet de calculer les nombres premiers inférieurs à une borne fixée `N`*. On part de l'ensemble de tous les entiers inférieurs à* `N`*. On supprime tous les multiples de* 2 *(sauf* 2 *qui est premier). Le plus petit nombre* > 2 *qui reste est alors* 3 *qui est premier. On supprime alors tous ses multiples (sauf lui même). On recommence la même opération avec le plus petit nombre* > 3 *restant et ceci jusqu'à ce que la liste ne contienne que des nombres premiers.*

- *Déduire des questions précédentes une fonction* `Crible` *prenant en argument un entier* `N` *et qui retourne la liste des entiers premiers inférieurs à* `N`*.*

▸ **Exercice 7.**

Écrire une fonction qui, étant données deux listes triées, renvoie 0 si elles contiennent les mêmes éléments aux mêmes positions, 1 sinon. Par exemple, la fonction appliquée aux listes `(1,2,2,3,3,3)` *et* `(1,2,2,3,3,3)` *retournera 1, et la fonction appliquée aux listes* `(1,2,2,3,3,3)` *et* `(1,2,3)` *retournera 0.*

Écrire une fonction qui, étant données deux listes triées, renvoie 0 si elles contiennent les mêmes éléments, 1 sinon. Par exemple, la fonction appliquée aux listes `(1,2,2,3,3,3)` *et* `(1,2,2,3,3,3)` *retournera 1, et la fonction appliquée aux listes* `(1,2,2,3,3,3)` *et* `(1,2,3)` *retournera 1.*

Calculer la complexité de chacune de ces fonctions.