

A.1 Tris simples

1. Soient les structures de données suivantes. Ecrire une fonction booléenne qui, étant donné une variable de type `liste` répond vrai si le tableau est (déjà) trié.

```
const M = 100 ;
type article = record
  cle : integer ;
  nom : string[4] ;
  prix : real ;
end ;
liste = array[1..M] of article ;
var T : liste ;
```

2. Ecrire une fonction booléenne de comparaison de tableaux de caractères : étant donnés deux tableaux de caractères terminés par `'\0'` (méthode de codage du C), soient u et v , renvoyer -1 si $u < v$, 0 si $u = v$, et 1 si $u > v$ (ordre lexicographique).
3. Proposer une version sans sentinelle de l'algorithme de tri par insertion, et déterminer le coût supplémentaire.
4. Proposer une version itérative de l'algorithme de tri par insertion.
5. **Champion borné** Ecrire un programme qui recherche dans un tableau la plus petite valeur comprises entre deux bornes a et b données par l'utilisateur.
6. Ecrire deux versions, l'une récursive, l'autre itérative, d'une fonction booléenne nommée `appartient` qui reçoit comme argument un tableau d'entiers et des indices de début et de fin, ainsi qu'une valeur (entière), et répond vrai si la valeur figure dans le tableau. Pour la version récursive, on pourra raisonner de la manière suivante : *un élément appartient au tableau soit s'il se trouve dans la première case, soit s'il appartient au tableau formé en enlevant la première case.*
7. On suppose que l'on dispose d'un tableau de chaînes de caractères (de longueur maximale 20 caractères : `string[20]`), de dimension N , correspondant à un ensemble de mots.

```
var Tm = array[1..N] of string[20]
```

Ce tableau n'est pas trié. On veut produire un tableau trié sur l'envers des mots (on considère chaque mot « à l'envers », de la dernière lettre à la première)¹.

- (a) Ecrire une fonction `avant` qui prend en argument deux chaînes de caractère, et répond vrai si la première est avant la seconde, dans le sens défini précédemment (c'est-à-dire en prenant les mots à l'envers).

```
function avant(x, y : string[20]) : boolean ;
```

- (b) Proposer un algorithme qui réalise le tri du tableau initial, en utilisant la fonction `avant`.
- (c) Discuter et justifier l'algorithme que vous avez choisi.

¹On utilise fréquemment ce genre de technique pour travailler sur les suffixes.