Algorithmique td 4 - tris et un problème récursif

Exercice 1. Écrire des procédures `maxi` qui prend comme argument une liste d'entiers naturels L et en retourne le maximum, ainsi que l'indice correspondant à ce maximum (i.e. i tel que $L[i] = \max(L)$).
Un outil que l'on peut utiliser : `nops(L)` *qui donne le nombre de termes de la liste* L*.*

Exercice 2. Utiliser les procédures précédentes pour écrire une procédure `tri1` de tri en ordre croissant d'une liste d'entiers.
Remarque : on créera une nouvelle liste, triée au fur et à mesure. Pour manipuler ces listes, on peut remplacer le maximum par -1 pour chercher le maximum suivant. Quand le maximum vaut -1, on a fini le tri.

Exercice 3. On se propose de répondre à la question suivante :

« de combien de façons peut-on choisir quatre maisons dans une rangée de douze, si l'on s'interdit de prendre deux maisons contiguës (c'est à dire à côté l'une de l'autre) ? » .

Soient n et k deux naturels non nuls. On note $F(n,k)$ le nombre de façons de choisir k maisons dans une rangée de n, de telle manière que deux maisons ne soient pas contiguës. Commençons par étudier la situation.

1. Justifier que $F(n,1) = n$, puis que $F(n,k) = 0$ si $n < 2k-1$.
2. Justifier que $F(2k-1,k) = 1$.
3. Démontrer que pour tout $(n,k) \in \mathbb{N} \times \mathbb{N}^*$, $F(n+2,k+1) = F(n,k) + F(n+1,k+1)$.
4. En déduire une procédure `F`, de variables n et k, récursive, qui calcule $F(n,k)$.
5. Vérifier sur des exemples la conjecture : $F(n,k) = \binom{n-k+1}{k}$.

Exercice 4 (tri à bulles)**.** L'algorithme parcourt le tableau, et compare les couples d'éléments successifs. Lorsque deux éléments successifs ne sont pas dans l'ordre croissant, ils sont échangés. Après chaque parcours complet du tableau, l'algorithme recommence l'opération. Lorsqu'aucun échange n'a lieu pendant un parcours, cela signifie que le tableau est trié. On arrête alors l'algorithme.

Exercice 5 (tri par selection)**.** Sur un tableau de n éléments (numérotés de 1 à n), le principe du tri par sélection est le suivant : rechercher le plus petit élément du tableau, et l'échanger avec l'élément d'indice 1 ; rechercher le second plus petit élément du tableau, et l'échanger avec l'élément d'indice 2 ; continuer de cette façon jusqu'à ce que le tableau soit entièrement trié.

Exercice 6 (tri par insertion)**.** Sur un tableau de n éléments (numérotés de 1 à n), le principe du tri par insertion est le suivant : on parcourt le tableau à trier du début à la fin. Au moment où on considère le i-ème élément, les éléments qui le précèdent sont déjà triés. L'objectif d'une étape est d'insérer le i-ème élément à sa place parmi ceux qui précèdent. Il faut pour cela trouver où l'élément doit être inséré en le comparant aux autres, puis décaler les éléments afin de pouvoir effectuer l'insertion. En pratique, ces deux actions sont fréquemment effectuées en une passe, qui consiste à faire « remonter » l'élément au fur et à mesure jusqu'à rencontrer à un élément plus petit.

En pseudo-code, ces trois algorithmes de tris donnent :

1. tri à bulles :
```
procédure tribulle(tableau T)
n=nb de termes de T
répéter
échangeeffectué = faux
pour j de 1 à n - 1
si T[j] > T[j + 1], alors
échanger T[j] et T[j + 1]
échangeeffectué = vrai
tant que échangeeffectué
```
2. tri par sélection :
```
procédure triselection(tableau T)
n=nb de termes de T
pour i de 1 à n - 1
min? i
pour j de i + 1 à n
si T[j] < T[min], alors min? j
si min ≠ i, alors échanger T[i] et
T[min]
```
3. tri par insertion :
```
procédure triinsertion(tableau T)
n=nb de termes de T
pour i de 1 à n - 1
x := T[i]
j = i
tant que j > 0 et T[j - 1] > x
T[j] = T[j - 1]
j = j - 1;
T[j] = x
```