Chapitre 14

Les techniques de Recherche et de Tri

Recherche linéaire (séquentielle)

On examine successivement tous les éléments de la table et on regarde sin on trouve l'élément recherché dans la table.

Nombre de tests de comparaison afin déterminer la complexité de ce type de recherche. Dans ce cas, nous effectuons au pire des cas n opérations, n .étant la taille de la table. On dit que la complexité d'une telle recherche est de l'ordre de n ou O (n).

Fichier : Search.java ; Méthode linearsearch

Recherche dichotomique

Si vous avez un tableau déjà trié de taille n, on peut écrire une fonction qui cherche si un élément donné se trouve dans la table. Comme le tableau est déjà trié, on peut procéder par dichotomie :
Cherchant à savoir si x est dans t[deb, fin[, on calcule milieu = (deb+fin)/2 et on compare x à t[milieu]. Si x=t[milieu], on a gagné, sinon on réessaie avec t[deb, milieu[si t[milieu]>x et dans t[milieu+1,fin[sinon.

Fichier : Search.java ; Méthode dichoSearch

Recherche récursive

On fait appel au même programme au cours de son déroulement.

Fichier : Search.java ; Méthode dichoRecurSearch

Fichier : Factoriel.java

Tri

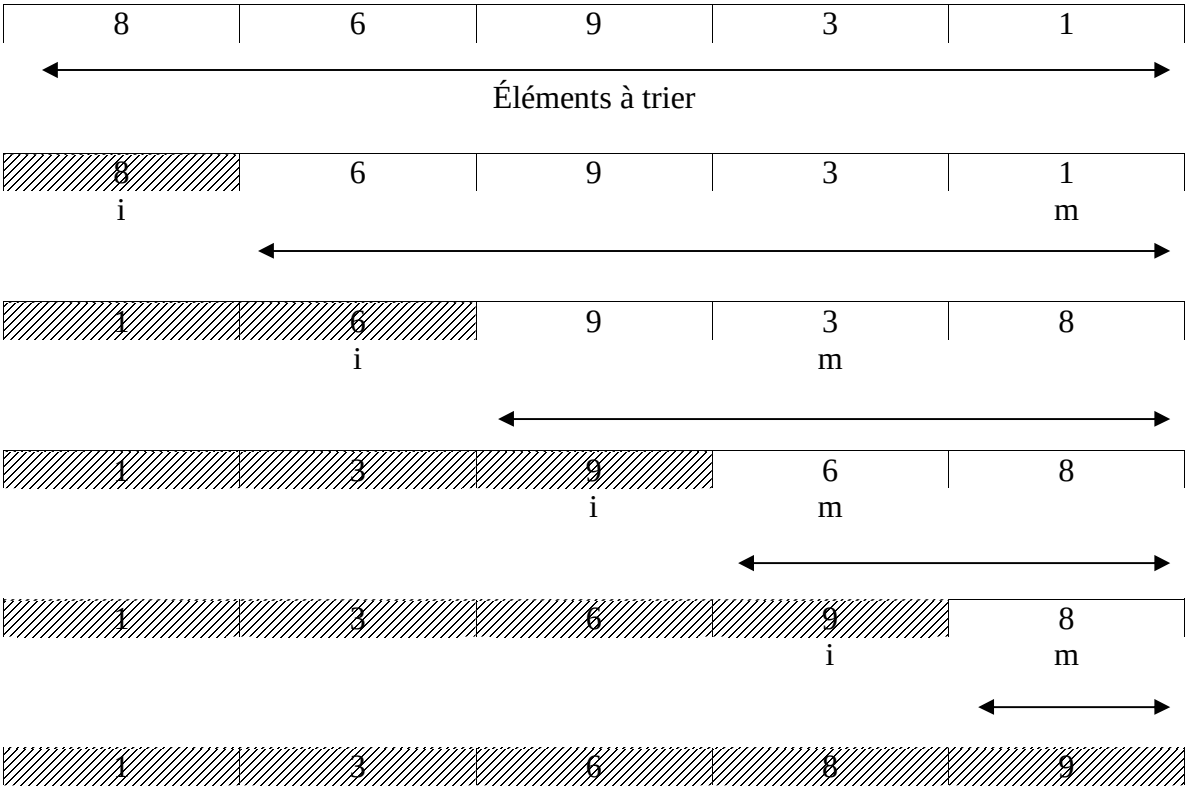
Le tri est rendu nécessaire par exemple s'il faut établir le classement des élèves, mettre en ordre certaines informations d'une application quelconque, comme par exemple le fichier membres du tp#3 ; tout cela afin de permettre de trouver l'information recherchée de manière rapide.
Nous allons étudier trois sortes de tris : par sélection, à bulles et par insertion.
On suppose que nous avons un tableau d'entiers de taille N. On suppose aussi que le tableau est indexé de 0 à N-1.

Tri par sélection

L'algorithme de tri associé au tri par sélection consiste à trouver l'emplacement du plus petit élément dans un tableau. Dès que cet élément est trouvé, nous l'inter changeons avec le premier élément du tableau (i=0). Nous recommençons l'opération pour le reste du tableau (i.e. i = [1, N [; N étant la taille du tableau).

Tableau à trier : [8, 6, 9, 3, 1].

L'index m pointe le plus petit élément dans un tableau contenant les éléments restants à trier.



FINI

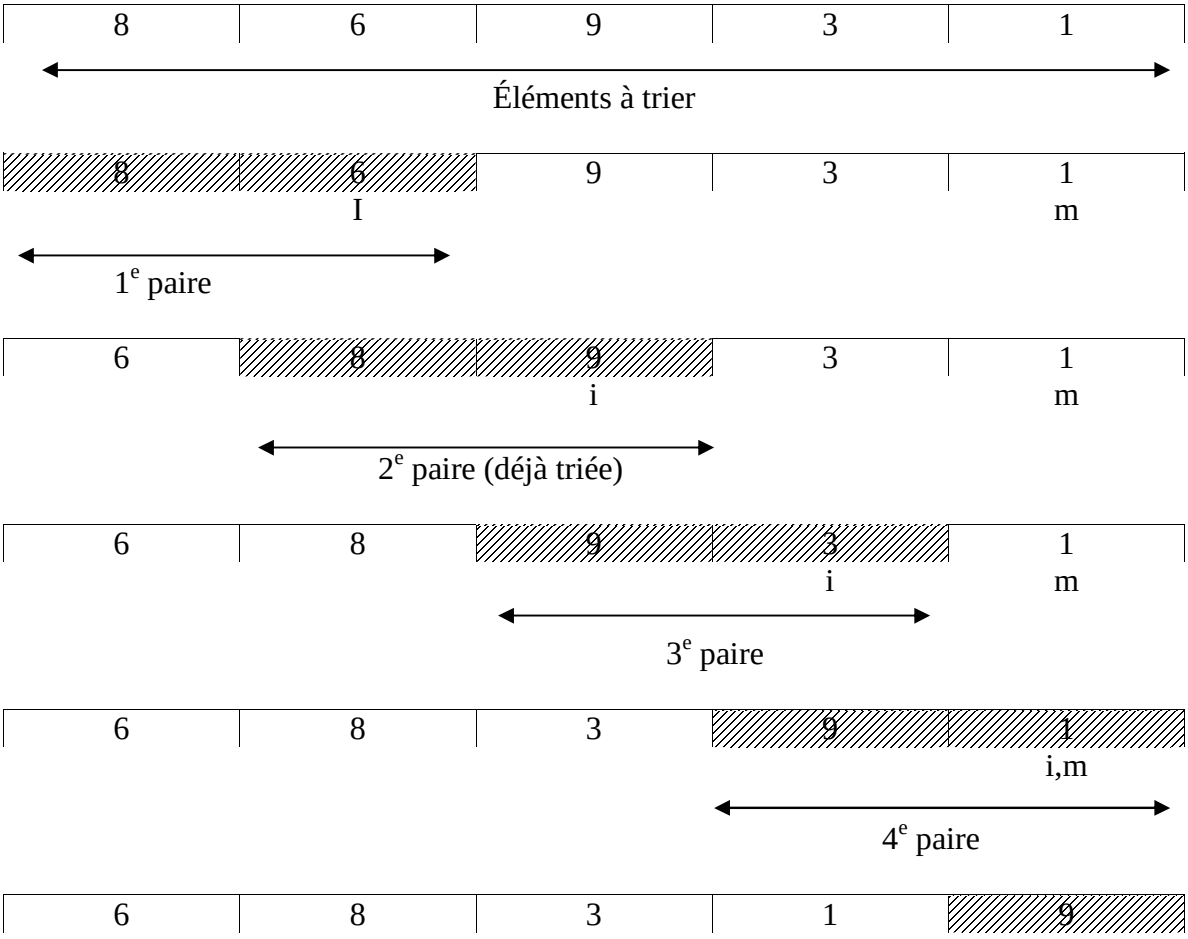
Fichier : Tris.java ; Méthode triParSelection

Tri à bulles

Le tri à bulles est une variante du tri par sélection. Son principe consiste à échanger deux éléments consécutifs qui ne sont pas ordonnés d'un tableau donné. Après ce parcours l'élément le plus grand va se retrouver en dernier. Nous recommençons l'opération avec les N-1 éléments du tableau [0, N-1[.

L'algorithmique est comme suit :

```
affecter true à flag
tantQue flag=true faire
  affecter false à flag
  pourChaque (éléments de tableau)-1
    si tableau[courant]>tableau[suivant] alors
       permuter les 2 valeurs
       affecter true à flag
    fin si
  fin pourChaque
fin tantQue
```



FINI itération -1-

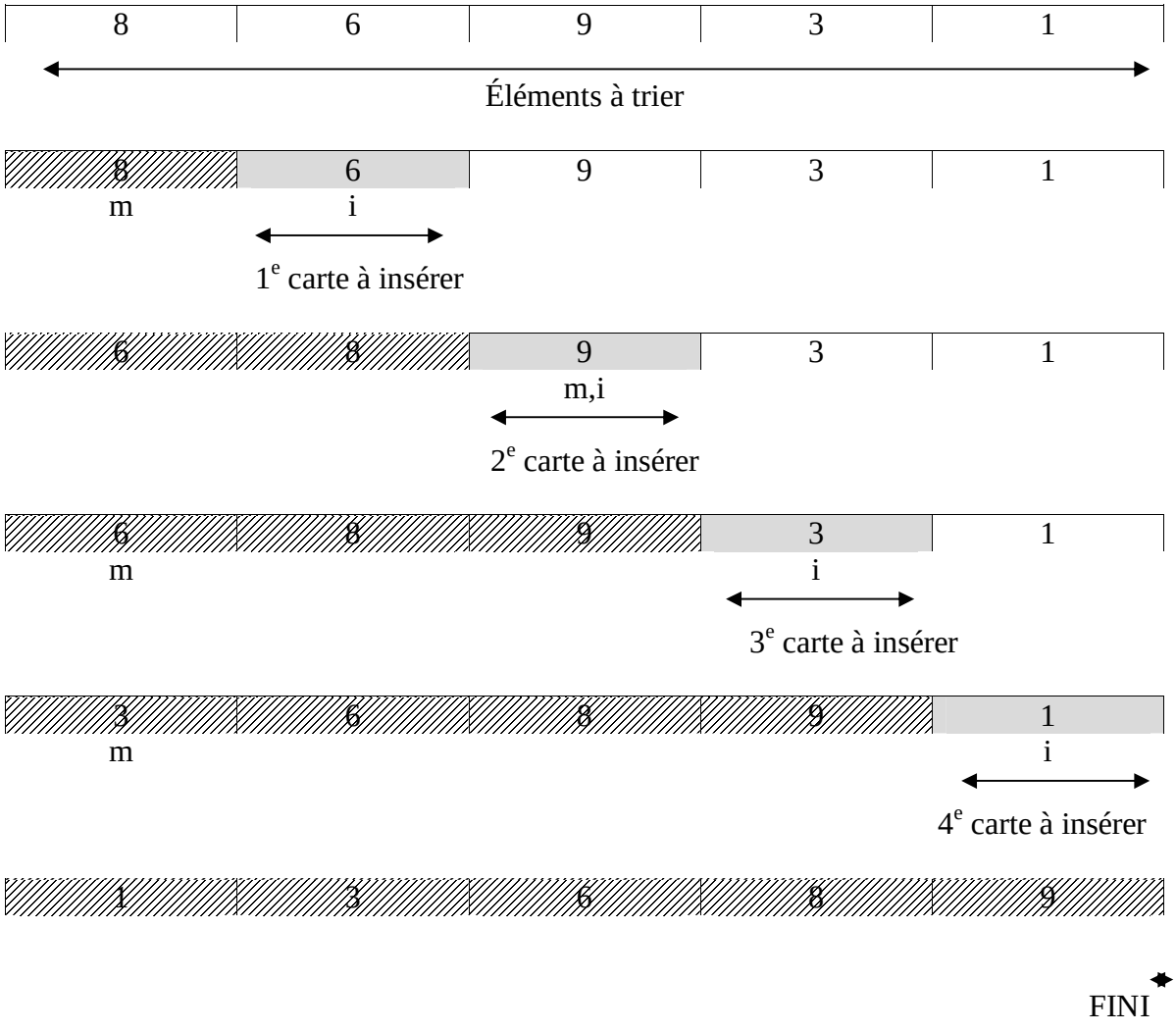
Ainsi prend fin la première itération avec l'élément le plus élevé qui se retrouve à la fin du tableau. Notons que la valeur de la variable est à true car nous avons permuté au moins une fois deux éléments consécutifs. Nous recommençons l'opération mais qu'avec les éléments non encore triés du tableau i.e : [6, 8, 3, 1].
Nous obtenons ainsi les résultats suivants pour les autres itérations :

Itération	Combinaison	flag
2	63189	true
3	31689	true
4	13689	true
5	13689	true

Fichier : Tris.java ; Méthode triABulle

Tri par insertion

L'algorithme du tri par insertion repose sur le même principe que la technique utilisée pour trier un paquet de cartes. Ayant i-1 cartes déjà triées, on essaye de mettre la i^{e} carte à sa place dans le paquet déjà trié. Ainsi de suite jusqu'à i=N, le nombre de cartes.
La variable m représente l'index de la case où l'élément sera inséré.
La variable i représente l'index de l'élément en cours de traitement.



Fichier : Tris.java ; Méthode triParInsertion