TP : File Chainée générique

1.1 Travail demandé :

- Ecrire le code de la classe FileChainee, à partir de l'interface IFileNonBornee, en stockant les valeurs grâce à des cellules chainées entre-elles.
- Tester cette classe.

1.2 Classe Cellule

Une file chainée déclare et utilise une classe interne (inner class) pour créer une succession de cellules chainées entre-elles.
Une cellule est une instance de la classe **Cellule** qui possède deux attributs :

- Un élément de type **E**
- La référence de la cellule suivante

Elle possède également un constructeur permettant de créer une cellule à partir d'un élément de type **E** passé en paramètre.

1.3 FileChainee

La file est définie par deux références de **Cellule**, premier qui donne accès à la cellule de tête et dernier qui donne accès à la cellule de queue.
Quand la file est vide, premier == dernier == null.

FileChainee<Elem> f = new FileChainnee<Elem>() ;

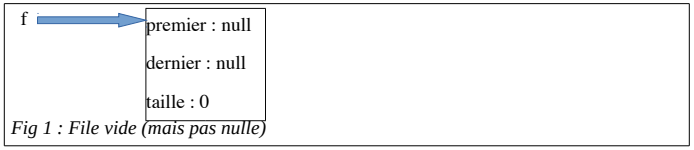

Fig 1 : File vide (mais pas nulle)

Quand la file ne contient qu'un élément, premier == dernier == référence de la cellule qui contient cet élément.

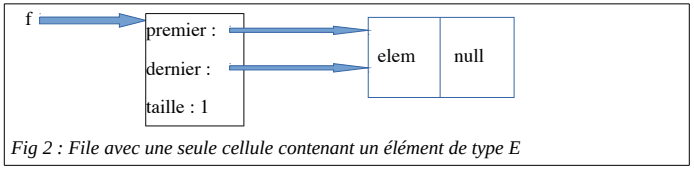

Fig 2 : File avec une seule cellule contenant un élément de type E

Quand la file contient plusieurs éléments, premier est la référence de la cellule de tête et dernier la référence de la cellule de queue.

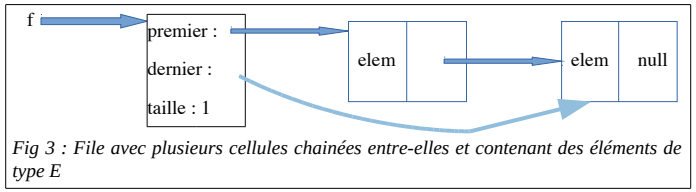

Fig 3 : File avec plusieurs cellules chainées entre-elles et contenant des éléments de type E

L'ajout d'un élément se fait différemment si la file est vide ou non.
La suppression d'un élément doit faire attention à conserver une file cohérente, car la suppression peut réduire la taille de la file à 1 ou 0 éléments.

La file chainée est déclarée comme suit :

```
public class FileChainee<E>  implements IFileNonBornee<E>  {
    // Inner classe, dont les attributs sont accessibles au this
    private class Cellule {
      E valeur;
      Cellule suivant;
      Cellule(E e) {
        valeur = e;
        suivant = null;
      }
    }
// Attributs
/** Référence de la cellule contenant le premier élément de la file */
    private Cellule premier;
/** Référence de la cellule contenant le dernier élément de la file */
    private Cellule dernier;
/** Taille de la file */
    private int taille;
// Constructeur
// Méthodes de l'interface à mettre en oeuvre
    …
}
```