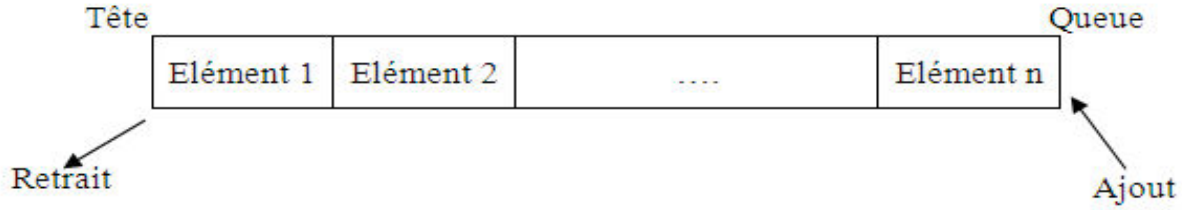
Chapitre 5
Structure de données : Files

1. Définition et types

La file d'attente est une structure qui permet de stocker des éléments dans un ordre donné et de les retirer dans le même ordre, c'est à dire selon le protocole FIFO *'first in first out'*.

« **On ajoute toujours un élément en queue de liste et on retire celui qui est en tête** ».



Selon le type d'implémentation, on distingue deux types de files:

- **File statique (contiguë):** Implémentation par un tableau.
- **File dynamique:** Implémentation par une liste chainée

2. Files statiques

2.1. Définition de type :

La définition du type **File** (statique) est comme suit :

Type Structure Pile

début

Tab: Tableau[MAX] d'Éléments ;

Tête : entier ; // indice de premier de la File.

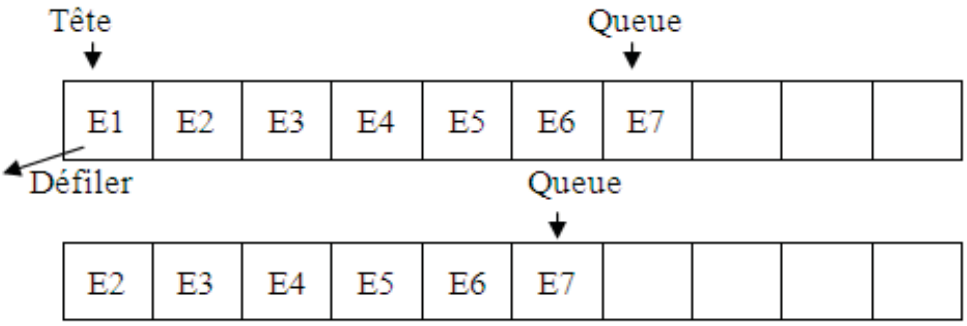
Queue: entier ; // indice de dernier élément inséré dans la File.

Fin

L'implémentation de files statiques peut être réalisée par décalage en utilisant un tableau avec une tête fixe, toujours à 1, et une queue variable. Elle peut être aussi réalisée par flot en utilisant un tableau circulaire où la tête et la queue sont toutes les deux variables

2.2. Implémentation par tableau (Par décalage)

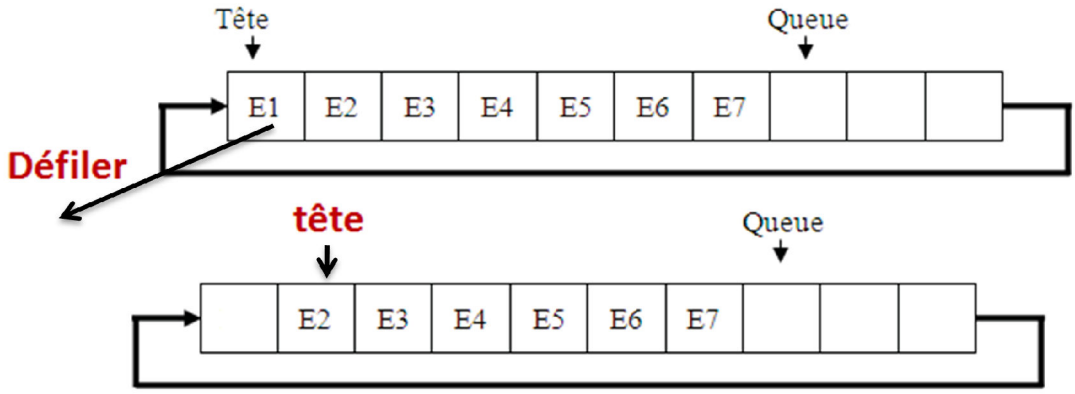
Tête fixe et queue variable**.** Elle soufre du problème de décalage à chaque défilement.



- La file est vide si Queue = 0
- La file est pleine si Queue = Max

2.3. Implémentation par tableau circulaire

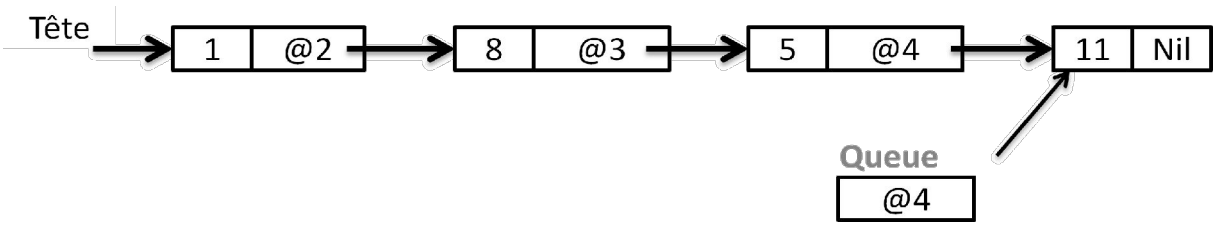
La file est représentée par un tableau circulaire (tête est variable et queue variable).



La file est vide si Tête = Queue
La file est pleine si (Queue + 1) mod Max = Tête

3. Files dynamiques

C'est une Liste chainée où le défilement se fait seulement à la tête et l'enfilement se fait seulement à la queue de la liste.



3.1. Définition de type File dynamique

La définition du type **File** (dynamique) estcomme suit :

Type

Structure Maillion

Ele : typeq;

suivant: * Maillion;

fin

Type Structure File

Début

Tête : *Maillon ; // Gade l'adresse de la tête de la File.

Queue: *Maillon; // Gade l'adresse ddernier élément inséré dans la File**.**

Fin

Dans le langage C++, on définira le type **File** (dynamique) comme suit :

```
struct maillon {
   type ele; // typeqq
   maillon *suivant;
}
struct file {
  maillon * tete;
  maillon * queue;
}
```

3.2. Opérations primitives

- **Est_vide**: retourne vrai si la file est vide sinon retourne faux.

```
Fonction est_vide (F: File): booléen
Début
    Retourner (F.tete ==NIL);
Fin
```

- **Tête** (F: File) : retourne l'élément existe dans l'entête de la File (Le premier élément enfilé).

```
Fonction  tete (F: File): entier
début
    Retourne (F.tete -> ele);
Fin
```

- **Enfiler :** ajoute un élément en queue de la file.

```
Procedure enfiler (var F: File,  x: typeq)
    P: *maillon
Début
      P ← créer_maillon (x);
    Si (est_vide (F)) alors
        F.tete ← P;
        F.Queue ← P;
    Sinon
        F. Queue -> suivant = P;
        F.Queue ← P;
    Finsi
Fin
```

- **Défiler** : supprime le premier élément (la tète) de la file.

```
Procedure défiler (var F: File)
  P: *maillon
Début
  Si( ! est_vide (f))
            P ← F.tête;
            F.tete ← f.tete ->suivant;
            libérer (p);
           Si (F.tete =Nil) alors
                F.Queue ←Nil;
           Finsi
    Finsi
Fin
```

- **La fonction taille**: retourne le nombre d'éléments déjà enfilés dans la pile.

```
fonction Taille (File : Pile) :entier
    Courant : *Maillon
    n : entier ;
Début
   Courant ← P ;  n=0 ;
    Tantque  (courant != NIL)

              n++;
              courant = courant->suivant ;
   fin tantque
    Retourner n ;
fin
```