CHAPITRE 3 : LES FILES

Une file est une structure de données utilisée pour le stockage de données (similaire aux listes chaînées et aux piles). Dans une file d'attente, l'ordre dans lequel arrivent les données est important.

Un exemple est une file de personnes qui attendent d'être servies dans un guichet de banque dans un ordre séquentiel commençant au début de la ligne.

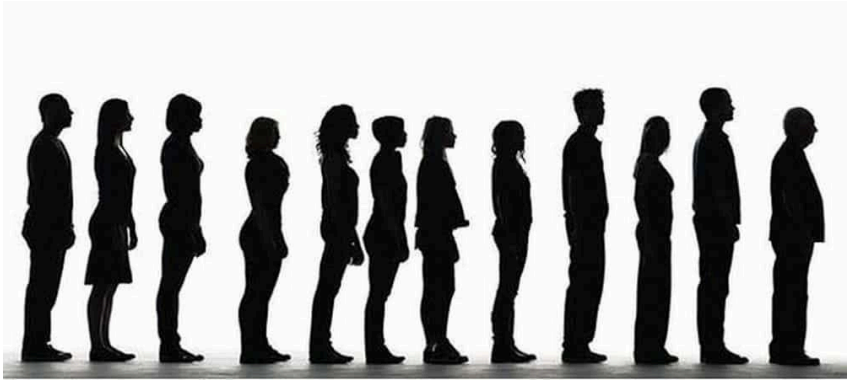

Figure 1 : File de personnes

DEFINITION : **Une file** est une liste ordonnée dans laquelle les insertions sont faites à une extrémité (par l'arrière) et les suppressions se font à l'autre bout (par le devant). Le premier élément inséré est le premier supprimé. C'est pourquoi on parle de **First in First out (FIFO)** ou **Last in Last out (LILO).**

Comme pour les piles, des noms spéciaux sont donnés aux deux modifications qui peuvent être apportées à une file. Lorsqu'un élément est inséré dans une file, le concept est appelé **enfiler**, et lorsqu'un élément est retiré de la file d'attente, le concept est appelé **défiler**.

Le concept de file peut s'expliquer par l'observation d'une file d'attente à un guichet de réservation. Lorsque vous entrez dans la queue, vous vous trouvez à la fin de celle-ci et la personne qui se trouve au début de la ligne est celle qui sera servie en premier. Dans ce cas, la personne suivante sera en tête de file, sortira de la file d'attente et sera servie. Comme chaque personne en tête de file continue à sortir de la file, vous vous dirigez vers la tête de la queue. Enfin, vous atteindrez la tête de la queue et vous en sortirez pour être servis.

Ce comportement est très utile dans les cas où il est nécessaire de maintenir l'ordre d'arrivée et de sortir.

OPERATIONS PRINCIPALES SUR LA FILE :

- **Enfiler(int donnee)** : Insère **donnee** en fin de file.
- **int Defiler()** : Supprime et renvoie le dernier élément inséré dans la file.

OPERATIONS AUXILIAIRES SUR LA FILE :

- **int Devant()** : Retourne le premier élément de la file.
- **int Size()** : Retourne le nombre d'éléments stockés dans la file (c'est la taille de la file).

- **int IsEmptyQueue()** : Indique si des éléments sont stockés dans la file ou non (autrement dit, si la file est vide, on renvoie 0 -False-, sinon, on renvoie 1 -True-).

EXCEPTIONS :

L'exécution de **Defiler** sur une file d'attente vide entraîne une exception de type « file d'attente vide » et l'exécution d'**Enfiler** sur une file d'attente pleine lève une exception « file d'attente pleine ».

APPLICATIONS :

Voici quelques-unes des applications dans lesquelles les files jouent un rôle important.

APPLICATIONS DIRECTES :

- Les systèmes d'exploitation programment les exécutions (avec une priorité égale) dans leur ordre d'arrivée ;
- Les serveurs d'impression, qui doivent traiter les requêtes dans l'ordre dans lequel elles arrivent, et les insèrent dans une file d'attente.
- En général, on utilise des files pour mémoriser temporairement des transactions qui doivent attendre pour être traitées ;
- Un algorithme de parcours en largeur d'un graphe utilise une file pour mémoriser les nœuds visités.

APPLICATIONS INDIRECTES :

- Structure de données auxiliaires pour les algorithmes ;
- Composante d'autres structures de données ;
- Pour créer toutes sortes de mémoires tampons (en anglais buffers) ;
- Une application possible d'une file est l'inversion de l'ordre des éléments d'une pile. En effet, il suffit de dépiler tous les éléments de la pile, puis de les ré-empiler dans l'ordre où ils sont sortis. On utilisera donc une file comme stockage intermédiaire.

IMPLEMENTATION :

Il existe de nombreuses façons de mettre en œuvre la File ; voici les méthodes les plus couramment utilisées :

- Mise en œuvre des listes simplement chaînées ;
- Mise en œuvre des tableaux ;
- Mise en œuvre des listes doublement chaînées.