Mini-projet C ANSI

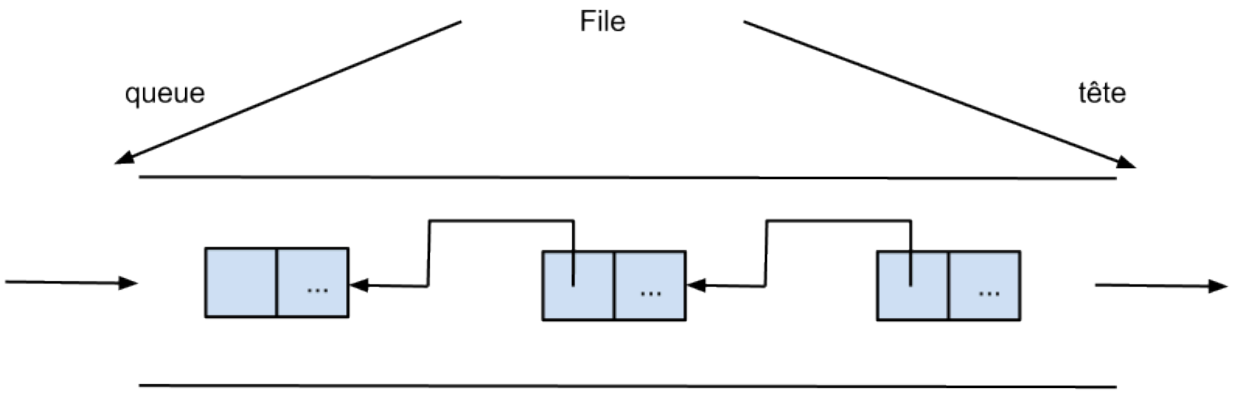
Trier

Vincent FROCHOT - Simon JORNET

Travail effectué

Implantation des files

Notre système de files suit le principe FIFO (First In First Out), à savoir que le premier élément ajouté sera le premier à sortir d'une file. Chaque *file* contient un maillon de tête, un maillon de queue, et des fonctions de copie et de libération de mémoire, qui permettent de stocker n'importe quel type de donnée souhaité.
Les files contiennent des éléments de types *maillon*, qui sont des structures contenant une valeur et un pointeur vers le maillon suivant.



Pour créer une file, on a donc besoin de passer à ***file_creer()*** :

- un pointeur d'emplacement mémoire
- un pointeur vers une fonction copier
- un pointeur sur une fonction libérer

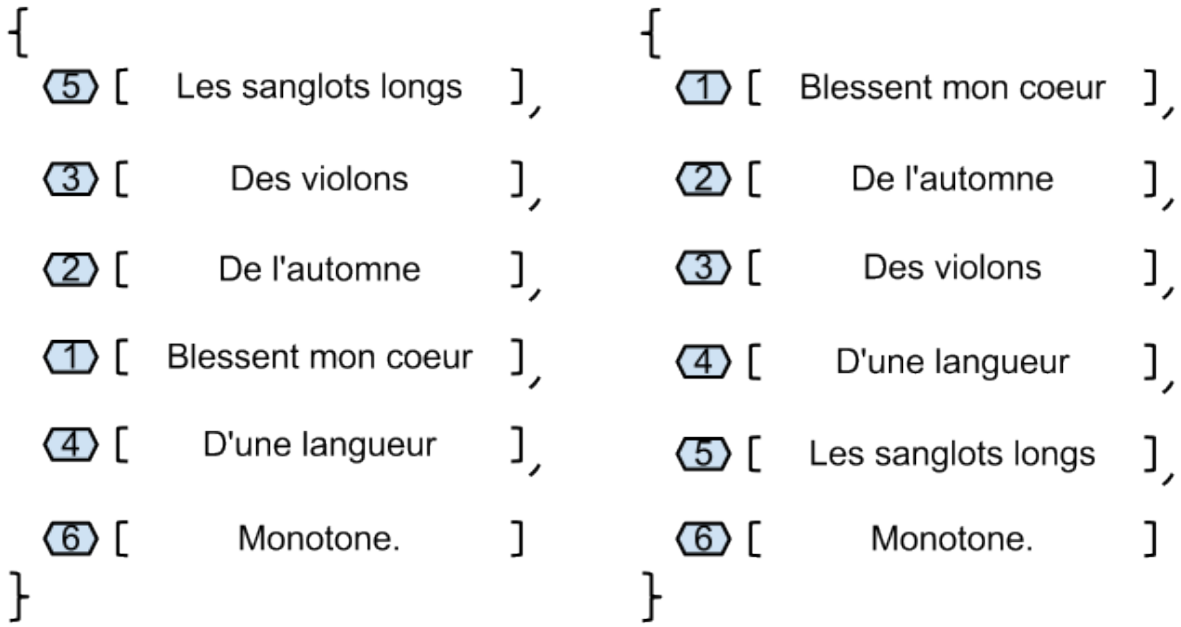
La fonction va alors s'occuper d'allouer la mémoire nécessaire à la file, en vérifiant que tout se passe correctement, et initialiser les éléments de la file.

Afin d'ajouter un élément à une file, on passe par la fonction ***enfiler()***, qui attend un pointeur sur une file et un autre sur une valeur. Elle va alors, si la file a été créé et celle-ci allouée, reserver la place nécessaire à un nouveau maillon dans la mémoire, et faire pointer l'ancien maillon de queue vers le nouveau maillon, qui devient alors la queue de la file.

Pour enlever un élément d'une liste, c'est la fonction ***defiler()*** à laquelle on fait appel. Elle ne prend en paramètre que le pointeur vers la file souhaitée, puisque l'on va défiler

l'élément de tête (pour suivre le principe FIFO). Elle met le maillon de tête dans une variable temporaire, fait pointer la tête de liste vers le nouvel élément en première position, avant le libérer la mémoire de celui défilé.

L'algorithme de tri



Entrée **Sortie**

L'algorithme de tri est un **tri par monotonie**. On transforme une file de valeurs en deux files de monotonies. On fusionne alors ces deux files par couple de monotonies, ce qui nous donne une nouvelle file unique, avec un nombre plus petit de monotonies. On procède ainsi jusqu'à ce que l'on n'ait plus qu'une seule monotonie sur la file.
Dans cette fonction, nous avons du faire appel à *strcmp* par le biais de notre propre fonction *string_file_comparer_elements()*, qui s'occupe de comparer deux éléments de deux files.

Extensions

En plus des fonctionnalités de bases décrites ci-dessus, nous avons également géré la **généricité** de nos files. On peut ainsi utiliser notre programme avec n'importe quel type de donnée dans celles-ci. En revanche, la fonction de tri n'a elle pas été codée de manière à tirer parti de cet avantage. On ne peut donc trier que des chaînes de caractères.

La gestion des **options en ligne de commande** a aussi été codée. En effet, il est possible à l'utilisateur de spécifier à l'exécution du programme s'il souhaite que les lignes soient triée par ordre inverse, s'il souhaite enlever les répétitions à la sortie, ainsi que les fichiers d'entrée et de sortie.

Conclusion

Nous avons rencontré quelques problème au niveau de la compréhension du sujet lors des premiers jours/semaines. En effet, le tri par monotonie nous était assez obscur, et nous avons passé un certain temps sur l'algorithmie du tri fusion, avant de devoir changer de cap. Nous avons donc perdu un certain temps à cela.
Un autre problème aura été clairement la gestion du temps, puisque nous avons dû au final rendu notre projet avec plusieurs heures de retard, afin d'éviter un certain nombre de concession notamment au niveau des finitions du code et du contenu du compte rendu, et de rendre un travail fini et fonctionnel.

Il a parfois été assez compliqué de travaillé en binôme, à cause des contraintes physiques et horaires, ainsi que pour la collaboration sur le code. Et ce, malgré la tentative d'utilisation de plateformes type SVN ou GitHub.

En résumé, ce mini projet aura tout de même été un bon moyen de se replonger en profondeur dans le C, et de manipuler de manière relativement avancée des notions comme les pointeurs. Il y aura aussi eu un intérêt au niveau de la manipulation de listes chaînées et de la gestion de la mémoire. Ce projet aura donc été l'occasion de se remettre dans le bain avec le C avant d'aborder de nouvelles notions comme l'objet avec le C++.