POLYTECH PARIS-SUD

Et3 - Tronc commun

Programmation C

TP7 - Structures de données - Deuxième Partie

Objectifs :

— Savoir définir une structure depuis une spécification
— Savoir manipuler des structures depuis des fonctions
— Savoir implanter quelques algorithmes sur des structures de données

Comme dans le TP précédent, n'oubliez pas de répartir votre code dans plusieurs fichiers, en regroupant les fonctions de manière logique, de faire les `include` dans l'ordre nécessaire et de correctement renseigner les fichiers *.h !

Notion de liste

Une liste chaînée est une structure de données représentant une collection ordonnée de taille arbitraire d'éléments de même type, dont la représentation en mémoire est une succession de cellules faites d'un contenu (la valeur de la cellule) et d'un pointeur vers une autre cellule. De façon imagée, l'ensemble des cellules ressemble à une chaîne dont les maillons seraient les cellules. L'accès aux éléments d'une liste se fait de manière séquentielle : chaque élément permet l'accès au suivant (contrairement au tableau dans lequel l'accès se fait de manière directe, par adressage de chaque cellule dudit tableau).

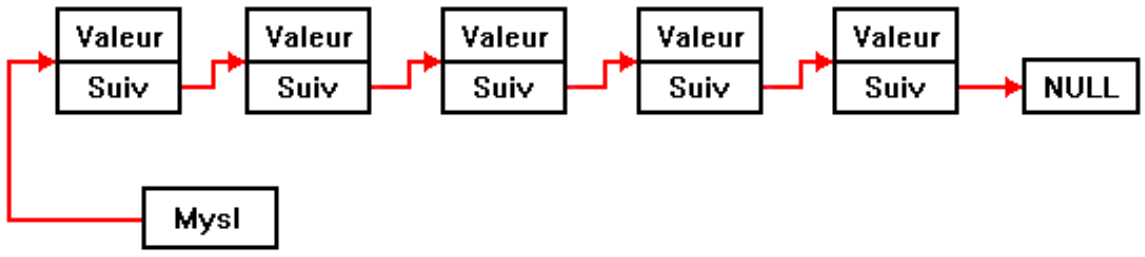

FIGURE 1 – Liste simplement chaînée

Exercice : 1 Manipulation de cellule

1. Définissez une structure `cell_t` qui représente une cellule de liste contenant comme valeur un nombre réel double précision et un pointeur vers une autre instance de `cell_t`.
2. Définissez une fonction `cell_create` qui alloue et initialise une cellule à partir d'une valeur réelle double précision. On considérera que cette cellule n'a pas de successeur.
3. Définissez une fonction `cell_delete` qui désalloue une cellule.

4. Une fois créées, les cellules doivent êtres liées les unes aux autres via leur membre `next`. Implémenter la fonction `cell_link` qui permet d'effectuer cette opération. Proposez une fonction `cell_unlink` qui effectue l'opération inverse en remettant à zéro le champ `next` de la cellule.
5. Modifiez la fonction `cell_delete` pour prendre en compte le chaînage et désalloue une cellule ainsi que toutes ses suivantes. Note : Faîtes en récursif.

Exercice : 2 Manipulation de liste

Une fois la structure `cell_t` définie, il va nous être possible de construire une liste simplement chaînée complète.

1. Définissez une structure `list_t` qui représente une liste chaînée contenant un pointeurs vers une instance de `cell_t` : `head`, qui pointe vers la première cellule de la liste.
2. Pour créer une liste, définissez une fonction `list_create`. Cette fonction alloue une instance de `list_t` et affecte à son membre `head` une valeur adéquate pour signifier que cette liste est vide. Quelle valeur semble appropriée ? Définissez ensuite la fonction `list_delete` qui détruit une instance de `list_t` sans se préoccuper de son contenu.
3. Implémentez une fonction `list_empty` qui renvoie 1 si la liste ne contient aucune cellule et 0 sinon. Attention, cette fonction ne nécessite aucun parcours de la liste.
4. Implémentez une fonction `list_push_front` qui insère une cellule au début de la liste chaînée.
5. Implémentez une fonction `list_push_back` qui insère une cellule à la fin de la liste chaînée. Comparez le nombre d'opérations nécessaires entre cette fonction et la fonction précédente. Quelle limitation des listes chaînées venez-vous de mettre en avant ?
6. Implémentez une fonction `list_find` qui parcourt les éléments d'une liste à la recherche d'une cellule contenant une valeur donnée. Un fois trouvée, la fonction renvoie un pointeur vers la cellule qui contient la valeur recherchée. Testez votre fonction sur une liste contenant des nombres aléatoires.

Exercice : 3 Les listes doublement chaînées

A partir de la notion de liste, on dérive à la liste doublement chaînée. Cette liste contient toujours des cellules mais ces cellules ont deux pointeurs : un vers la cellule suivante et un vers la cellule précédente. La structure de liste elle-même contient un pointeur vers la première cellule et une vers la dernière.

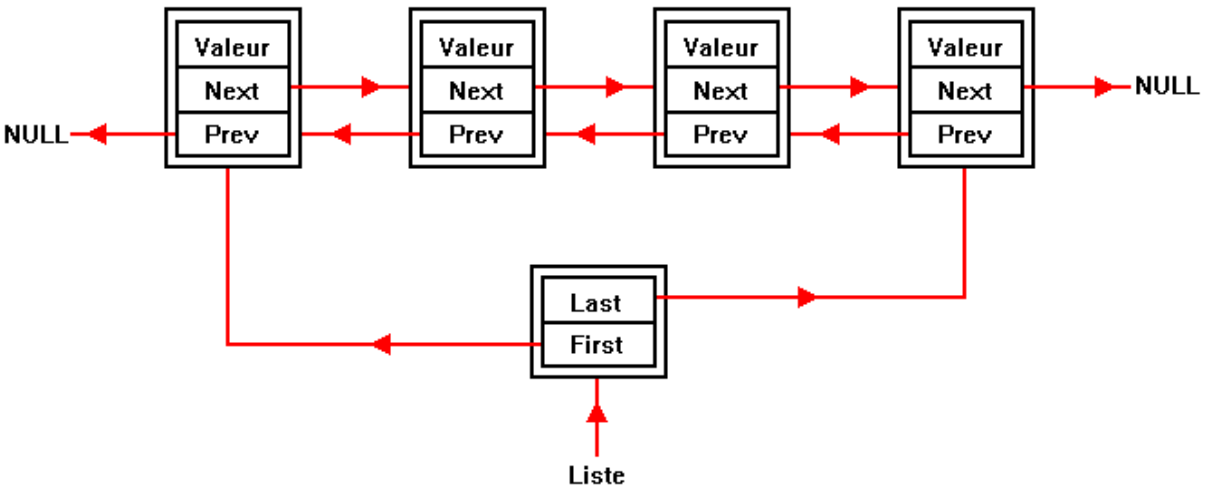

FIGURE 2 – Liste doublement chaînée

Ré-implémentez les structures et les fonctions de la liste chaînée pour supporter la liste doublement chaînée. Quelles fonctions sur les listes voient leur implémentation se simplifier ?

Exercice : 4 Algorithmes

Pour les deux types de liste implémentés, implémentez les fonctions et procédures suivantes :

1. `zip` qui prend deux listes du même type et renvoie une unique liste contenant de manière alternée les cellules de ses deux arguments. Ainsi les listes `[1 2 3]` et `[0.1 0.01 0.001]` créent la liste `[1 0.1 2 0.01 3 0.001]`
2. `find_last` qui prend en paramètre une liste et une valeur réelle double précision. Elle retourne le pointeur vers la dernière cellule contenant cette valeur ou `NULL` si la valeur n'est pas présente dans la liste.
3. `reverse` qui prend une liste en paramètre et inverse l'ordre des cellules.
4. `filter` qui prend en paramètre une liste et un pointeur vers une fonction `f` elle-même prenant un réel double précision en paramètre et retournant un entier 0 ou 1. Cette fonction renvoie une liste contenant les cellules pour lesquelles la fonction `f` appliquée à sa valeur retourne 1.

Pour certaines de ces fonctions, il faudra prendre soin de copier les cellules si besoin. Implantez donc une fonction `cell_copy` qui effectue cette tâche.