TD 3 - I41

1 Liste doublement chaînée

Ce TD propose de gérer des listes doublement chainées. Dans les listes doublements chainées, le chainage est réalisé par 2 pointeurs : le premier pointeur `pred` pointe sur l'élément situé avant l'élément courant, le second pointeur `succ` pointe sur le successeur de l'élément courant. La liste $1, 2, 3, 4$ sera donc représentée par la structure définie dans la figure 1.

Pour ce TD, vous ferez individuellement les exercices d'abord sur feuille avant de les implémenter.

1. Déclarez le type `list2_t` correspondant à une liste doublement chainée.
2. Ecrire la fonction `list2_t * new(int val)` qui crée un nœud.
3. Faite une fonction `list2_t * nth(list2_t *h, int n)` qui retourne l'adresse du $n^{ième}$ élément si celui-ci existe. Sinon cette fonction donne l'adresse du dernier élément de la liste qui sera `NULL` si aucun élément n'est présent.
4. Faites une fonction `insert(list2_t **h, list2_t *t, int n)` qui insère un objet `p` après le $n^{ième}$ élément **si cela est possible**. Pour l'insertion en tête, `n = 0`.
5. Faites une fonction `list2_t * retrait(list2_t **h, int n)` qui retire le $n^{ième}$ élément **si cela est possible** et le retourne. Cet élément ne doit plus avoir de références sur la liste.
6. Faites une fonction `print(list2_t *h, sens_t dir)` qui imprime la liste selon deux directions : si `dir == endroit`, l'impression sera dans le sens tête vers queue, si `dir == envers`, l'impression s'effectuera dans le sens queue vers tête. Vous définirez le type `sens_t` qui décrit le sens de parcours de la liste.

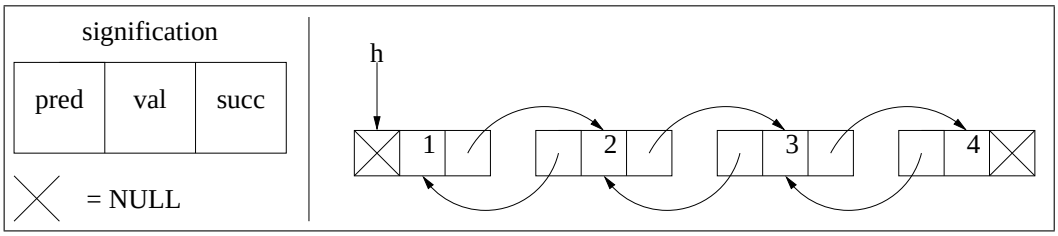


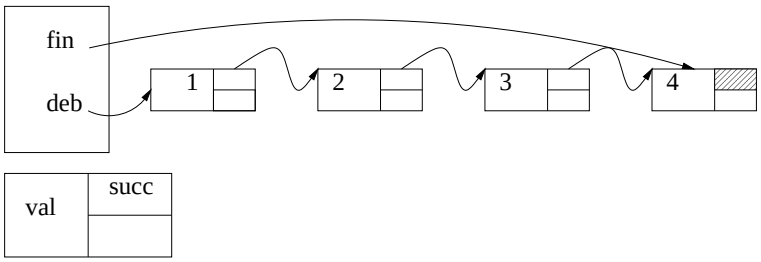
FIG. 1 – Représentation d'une liste doublement chainée

2 Gestion de file

On se propose de définir plusieurs stratégies pour gérer une file d'attente en utilisant une liste chaînée.

- la structure de listes est la suivante :
  `typedef struct LIST_T { int val;  struct LIST_T * succ ;} liste_t`
- la structure de file est la suivante :
  `typedef struct FILE_T { liste_t * deb ; liste_t *fin} file_t`
  cette structure contient 2 pointeurs : `deb` pointe sur le début de la liste. `fin` pointe sur la fin de la liste.

Par exemple, une file contenant les valeurs $\{1, 2, 3, 4\}$ est implantée de la manière suivante :



Par implantation, vous ferez les exercices d'abord sur feuille avant de les implanter.

Implantation préliminaire Ecrivez la fonction `liste_t * new(int val)` qui crée un noeud de la liste possédant la valeur `val`.

Première implantation Implantation sans optimisation.

1. Ecrivez la fonction `file_t * newfile(void)` qui crée une file vide.
2. Faites une fonction `void add(file_t * f, int val)` qui insère un élément en fin de la file. Envisagez les différents cas.
3. Faites une fonction `int del(file_t * f)` qui retire l'élément en tête de la file, qui libère le nœud de la liste. Cette fonction retourne le numéro retiré de la file. Si la file est vide, cet élément est 0.
4. Faites une fonction qui imprime le contenu de la file.

Seconde implantation Dans cette implantation, on ajoute dès la création, un élément «dummy» dont la valeur (`val`) est 0, qui doit être toujours en tête de la file et ne jamais être retiré. Cet élément permet de simplifier la gestion des files lors du retrait ou de l'insertion car la file n'est jamais vide.

Modifiez les fonctions précédentes en tenant compte de cet élément.

dummy : factice. provient de la culture informatique anglo-saxonne. Il désigne un élément qui sert uniquement à simplifier le traitement mais qui n'apparaît pas dans le résultat.

Troisième implantation Dans cette dernière implantation, on considérera une liste allouée dès la création de la file qui contient `N` noeuds et qui est rebouclée sur elle-même (le suivant du dernier nœud correspond au premier élément) Ni on n'alloue, ni on ne libère de nœuds lors de l'insertion ou du retrait. L'effet de l'insertion ou du retrait est de faire évoluer les pointeurs de tête et de fin selon les règles suivantes :
- `deb` pointe toujours sur le dernier nœud à traiter.
- `fin` pointe toujours sur le première cellule de libre.

1. quelle condition identifie une file vide ?
2. quelle condition identifie une file pleine ?
3. implantez les fonctions de création de liste circulaire (N cellules), d'ajout et de retrait.