Cahier de TD n°6

Séance 8 : Structures définissant des listes chaînées

Parmi les structures suivantes (et à part la première qui définit le type complexe), quelles sont celles qui définissent correctement une liste chaînée ? Rappel : les listes chaînées sont susceptibles de stocker des valeurs de n'importe quel type, pas uniquement des entiers.

```
structure t_complex
{
     reel re, im ;
}

structure  t_cell_1
{
     caractere *nom ;
     t_cell_1   suivant ;
}

structure t_cell_2
{
     t_complex z ;
     t_cell_2   *next ;
}

structure t_cell_3
{
     t_cell_3   *prochain ;
     reel *vals_r ;
     entier ident;
     caractere carac;
     entier toto;
}

structure t_cell_4_0
{
     entier valeur ;
     t_cell_4_1 suiv ;
}

structure t_cell_4_1
{
     entier valeur ;
     t_cell_4_0 suiv ;
}
```

VRAI /FAUX

Indiquez si les affirmations suivantes, concernant les listes chaînées et les tableaux, sont vraies ou fausses :

- Une liste chaînée est plus simple à initialiser qu'un tableau ;
- L 'accès à un élément quelconque d'un tableau est plus rapide que l'accès à un élément quelconque d'une liste chaînée ;
- Une liste chaînée est plus pratique qu'un tableau
- Une liste chaînée a une taille maximum
- Il est plus simple d'échanger 2 éléments d'un tableau que 2 éléments d'une liste chaînée
- Il est plus simple d'insérer ou de supprimer un élément dans une liste chaînée que dans un tableau
- La fin d'une liste chaînée se repère par sa taille utile
- Le début (tête) d'une liste chaînée est un tableau
- Le début (tête) d'une liste chaînée est un pointeur vers une cellule

Algorithmes de base pour les listes chaînées.

Dans ce thème, on cherche à réaliser les différentes opérations de base pour initialiser une liste chaînée et réaliser quelques opérations.

Insertion de cellules en tête de liste chaînée

On utilisera une liste chaînée de caractères, qui est vide au départ.

a) définissez le type cellule permettant de créer une liste chaînée de caractères

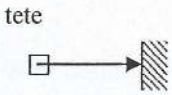
b) écrire une fonction de création de cellule à laquelle on fournit un caractère et qui retourne un pointeur vers une cellule contenant ce caractère

c) écrire une fonction qui ajoute une cellule en tête de la liste chaînée (cela s'appelle : **chaînage en tête de liste**)

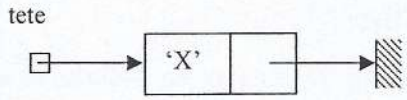
d) écrire un programme principal qui propose des saisies de caractères et qui les range dans une liste chaînée au fur et à mesure : avec un

chaînage en tête, les caractères seront rangés en ordre inverse de leur saisie.

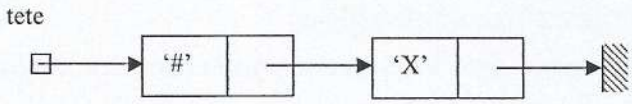
Illustration du chaînage en tête : au départ, la liste chaînée est vide, et on suppose que l'on saisit, dans l'ordre, les caractères 'X', '#' et 'a'.



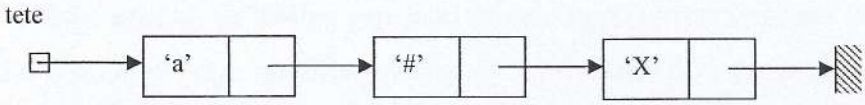
Saisie de 'X'



Saisie de '#'



Saisie de 'a'



Recherche d'un valeur dans une liste chaînée

Nous utiliserons, pour cet exercice, une liste chaînée contenant des valeurs de type entier, qui est déjà initialisée. Les valeurs ne sont pas classées et peuvent apparaître à plusieurs exemplaires dans la liste.

a) Ecrire une fonction à qui l'on fournit une valeur et la liste chaînée et qui indique si la valeur d'y trouve

b) Ecrire une fonction à qui l'on fournit une valeur et la liste chaînée et qui indique le nombre de fois où cette valeur se trouve dans la liste.

Séance 9 : Insertion dans une liste chaînée existante

Nous utiliserons, pour cet exercice, une liste chaînée contenant des valeurs de type réel. Cette liste chaînée contient déjà un certain nombre de cellules, et l'on sait de plus que les

valeurs y sont classées par ordre décroissant. L'utilisateur saisit une nouvelle valeur, et il faut créer une cellule contenant cette valeur, puis l'insérer au bon endroit dans la liste chaînée.

a) à l'aide de schémas, indiquez les opérations (instructions) à écrire pour les 3 cas suivants. Vous indiquerez également les conditions à tester pour repérer ces différents cas.

α) la cellule doit être insérée en tête de liste ;

β) la cellule doit être insérée dans la liste à un endroit quelconque, sauf en tête ou en fin ;

γ) la cellule doit être insérée en fin de liste ;

b) écrire une fonction à qui l'on fournit la liste et la cellule et qui réalise l'insertion de la cellule dans la liste au bon endroit.

Suppression d'une cellule – suppression de doublons

Nous utiliserons, pour cet exercice, une liste chaînée contenant des valeurs de type entier, qui est déjà initialisée. Les valeurs ne sont pas classées et peuvent apparaître à plusieurs exemplaires dans la liste.

Ecrire une fonction permettant de supprimer une cellule de la liste chaînée si celle-ci contient une valeur saisie par l'utilisateur. Ecrire, à partir de cette fonction, une fonction permettant de supprimer tous les doublons d'une liste chaînée.

Récursivité

Fonctions récursives simples

Les suites numériques du type $U_{n+1} = f(U_n)$ peuvent se programmer de manière assez simple en utilisant la récursivité : programmer à l'aide de fonctions récursives le calcul du terme de rang n (n étant fourni par l'utilisateur) pour les suites suivantes :

U : $U_0 = 4$

$U_{n+1} = U_n/2 - 2/U_n$; pour $n > 0$

F : $F_0 = 0$

$F_1 = 1$

$F_n = F_{n-1} + F_{n-2}$ si $n > 1$

Dressez un schéma des appels de fonction successifs pour matérialiser le nombre d'appel à chaque fonction.

Placement des appels de fonction et autres instructions

Que fait le programme suivant (comportant des fonctions récursives) ?

```
fonction affich_1(entree : caractere *texte, entier pos)
{
      si (pos < longueur_texte(texte)) alors
      {
            afficher(texte[pos]) ;
            affich_1(texte,pos+1) ;
      }

      retourner ;
}

fonction affich_2(entree : caractere *texte, entier pos)
{
      si (pos < longueur_texte(texte)) alors
      {
            affich_2(texte,pos+1) ;
            afficher(texte[pos]) ;
      }

      retourner ;
}

programme affichages

caractere texte[50]← "un texte à afficher" ;

affich_1(texte,0);
affich_2(texte,0);
```

Séance 10

Palindrome

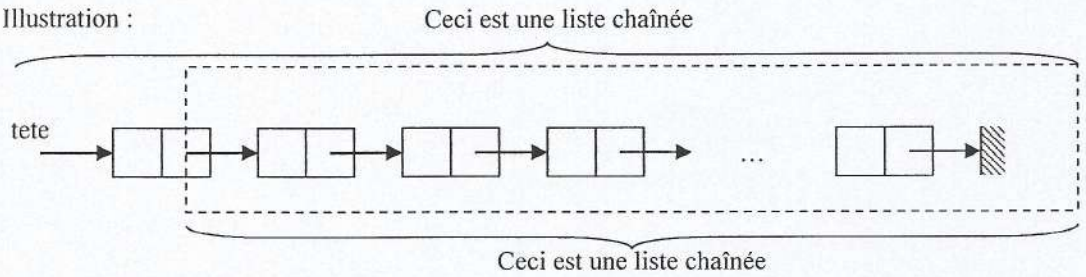
Ecrire une fonction récursive déterminant si un mot est un palindrome. Un palindrome est un mot qui peut être lu indifféremment de la droite vers la gauche ou de la gauche vers la droite. Exemples : Radar, Non, Lsertiofoitresl sont des palindromes (même si le sens de ce dernier exemple n'est pas encore clairement établi dans la langue française, l'ordinateur considèrera qu'il s'agit d'un palindrome, puisqu'il est incapable de déterminer si un mot a un sens ou non).

Récursivité et listes chaînées

L'emploi de fonctions récursives est bien adapté au traitement des listes chaînées car une liste chaînée a une définition récursive :

Une liste chaînée est matérialisée un pointeur vers une cellule (la tête de liste), cette cellule comprenant une valeur et une liste chaînée (pointeur vers une cellule).

Illustration :



Ainsi, pour réaliser l'affichage d'une liste chaînée, on peut se baser sur cette définition récursive : si `tete` est un pointeur vers une cellule, donc permet d'accéder à une liste chaînée, alors, `tete->suivant` (ou `tete->next` selon le nom du champ choisi) est un pointeur vers une cellule, donc permet d'accéder à une liste chaînée.

Ecrire une fonction qui permet d'afficher une liste chaînée de manière récursive. Il faut bien faire attention à repérer la fin de liste.

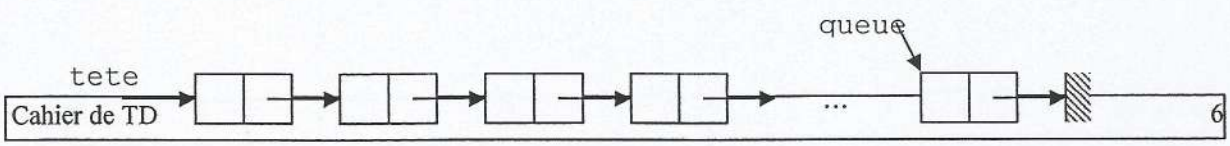
Ecrire une fonction permettant d'afficher les valeurs rangées dans la liste chaînée en ordre inverse de celui de leur classement dans la liste.

Pour cette dernière séance de l'année, nous allons aborder des structures de données plus sophistiquées que les listes chaînées.

Listes chaînées circulaires.

Soit `tete` un pointeur vers une cellule en tête de liste chaînée, et `queue` un pointeur sur la dernière cellule de la liste.

Quel est l'effet de l'instruction `queue->suiv ← tete` ?



Que se passe-t-il si l'on utilise la fonction d'affichage écrite pour une liste chaînée simple sur la liste obtenue ? Comment modifier cette fonction pour qu'elle puisse traiter les deux cas (liste non circulaire et liste circulaire).

Liste doublement chaînée.

Le principal inconvénient, avec les listes chaînées, est que l'on ne peut la parcourir que dans un seul sens : on ne peut pas passer d'un élément au précédent, mais seulement au suivant.

Comment modifier la structure `cellule` pour arriver à progresser dans les deux sens ?

Dans ce cas, comment initialiser la liste obtenue et comment y ajouter une cellule en tête de liste ? Comment y ajouter une cellule en milieu de liste ?