Univérsité Hassan II Casablanca

Ecole normale supérieure (ENS) Casablanca

Module : Structures de données en C (Info 4)

Année universitaire 2020/2021

Parcours 2éme année LEESM **Prof. Mohamed Lahby**

TD 4

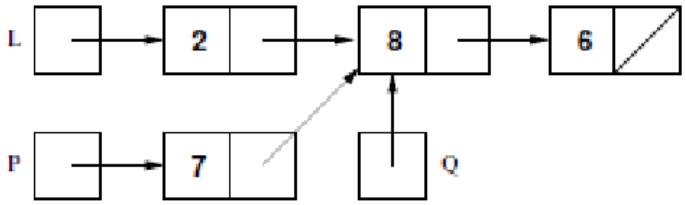
Liste simplement chaînée

Exercice 1

On considère une liste chaînée définie selon le type liste suivant :

```
typedef struct cellule
{
  int info;
  struct cellule *succ;
} liste;
```

On déclare trois listes L, P et Q : liste * L,*P,*Q.
La figure ci-dessous donne les représentations graphiques de 3 listes simplement chaînées L ; P ;Q.
Les séquences d'éléments de ces trois listes sont respectivement (2 ;8 ; 6) ; (7 ; 8 ; 6) et (8 ; 6).



Q1. On exécute l'instruction $(Q->succ)->info = P->info$. Quelles sont les nouvelles séquences d'éléments des trois listes L ; P et Q ?

Q2. On revient à l'état décrit par la figure, puis exécute l'instruction $L = Q->succ$. Quelles sont les séquences d'éléments de L ; P ;Q ?

Q3. On revient à l'état décrit par la figure, puis exécute l'instruction $(L->succ) = Q->succ$. Quelles sont les séquences d'éléments des listes L ; P ;Q ?

Q4. On revient à l'état décrit par la figure, puis exécute l'instruction $P = P->succ$. Quelles sont les séquences d'éléments de L ; P ;Q ?

Q5. On revient à l'état décrit par la figure, puis exécute l'instruction $P->succ = Q->succ$. Quelles sont les séquences d'éléments de L ; P ;Q ?

Exercice 2 : *"Création d'une cellule"*

Q1. Ecrire une fonction d'entête **liste* AlloueCellule(....)** qui crée une nouvelle cellule et retourne l'adresse de la cellule crée.

Q2. En déduire une fonction d'entête **liste* CreerCellule(...)** prenant en argument un entier val et retournant un pointeur sur une nouvelle cellule contenant la valeur val. Nous veillerons a ce que le champ Suivant de cette cellule soit initialise a la valeur NULL

Exercice 3 : *"Insertion d'un élément"*

Q1. 1.Ecrire une fonction d'entête **liste* InsertDebut(.....)** qui insère, en tête de liste, un entier dans une liste chaînée.

Q2. En déduire une fonction qui crée (par insertions en tête successives des éléments) une liste chaînée d'entiers non nuls saisis au clavier.

Q3. Ecrire une fonctiond'entête **liste* InsertOrd(.....)** qui retourne une liste triée obtenue par l'insertion d'un nouvel élément à une liste triée.

Exercice 4 : *"Affichage d'une liste"*

Q1. Ecrire une fonction itérative d'entête **int Taille_it(liste * L)** qui retourne le nombre d'éléments d'une liste.

Q2. Ecrire une fonction récursive d'entête **int Taille_re(liste * L)** qui retourne le nombre d'éléments d'une liste.

Q3. Ecrire une fonction d'entête **void AffListe(liste * L)** qui affiche les éléments de la liste chaînée passée en paramètre. Vous donnerez une version itérative et une version récursive de cette fonction.

Q4. Ecrire une fonction d'entête **void AffListeInv(liste * L)** qui affiche, dans l'ordre inverse du chaînage, les éléments de la liste chaînée passée en paramètre.

Exercice 5 : *"Suppression d'une liste"*

Q1. Ecrire une fonction d'entête **liste* SupprimDebut(liste*L)** qui supprime le premier élément d'une liste chaînée.

Q2. Ecrire une fonction d'entête **liste* SupprimFin(liste*L)** qui supprime le dernier élément d'une liste chaînée.

Q3. Ecrire une fonction d'entête **liste* SupprimElement(liste*L, int n)** qui supprime la première occurrence d'un entier n dans une liste chaînée.

Q4. Ecrire une fonction d'entête **liste* SupprimTout(liste*L, int n)** qui qui supprime toutes les occurrences d'un entier n dans une liste chaînee.

Exercice 6 : *"Libération d'une liste"*

Q1. Ecrire une fonction d'entête **void LibereListe(liste * L)** qui libère tout l'espace mémoire occupe par une liste chaînée.

Exercice 7 : *"Somme,Moyenne"*

Q1. Ecrire une fonction d'entête **int sommeElement(liste * L)** qui retourne la somme des éléments d'une liste chaînée.

Q2. Ecrire une fonction d'entête **float MoyennneListe(liste * L)** qui retourne la moyenne de tous les éléments de liste