Université Hassiba Benbouali de Chlef
Faculté des Sciences
Département d'informatique

ASD II
Niveau: Licence 2
2022-2023

TP3: Liste Chaînée

Partie 1 (Liste simplement chaînée)
On considérera dans cette partie une liste simplement chaînée déclarée comme suite :

```
typedef struct Element
{
    int donnee;
    struct Element * suivant;
} Element;
```

Implémenter en C les fonctions suivantes et écrire une fonction int main (void) permettant de les tester.

- Création d'une liste vide: Element *** cree_liste_vide();**
- Ajouter un élément en début de liste : **Element * ajout_debut(Element * liste, int e);**
- Suppression d'un élément en début de liste: **Element* supprime_debut(Element * liste);**
- Afficher les éléments de la liste : **void afficher(Element * liste);**
- nombre d'éléments de la liste:**int taille(Element * liste);**
- Renvoyer le Jème élément de la liste : **int jieme(Element * liste, int j);**
- Ajouter un élément en position j : **Element * ajout_position(Element * liste, int j, int e);**

Partie 2 (Liste doublement chaînée)

L'objectif de ce TP est d'écrire une nouvelle implantation de liste chaînée, différente de celle vue en cours(Liste simplement chaînée), mais ayant la même interface (mêmes services proposés aux utilisateurs du module). L'implantation que vous allez écrire est celle d'une liste doublement chaînée, dans laquelle:

1. chaque cellule contient un pointeur sur la cellule suivante et un pointeur sur la cellule précédente,
2. la structure Liste contient un pointeur sur la première cellule et un pointeur sur la dernière cellule.

Ainsi, il est possible de parcourir la liste dans les deux sens.

Un autre intérêt de cette implantation est que l'insertion d'un élément en fin de liste (ajoutEnQueue) peut se faire en temps constant, c'est-à-dire en un nombre d'opérations qui ne dépend pas du nombre d'éléments dans la liste. Etait-ce le cas avec l'implantation vue en cours ? Pourquoi ?

Récupérez les deux fichiers Liste.h et Liste.c. Enregistrez les dans votre répertoire Ecrivez-y les #include requis, puis le code de la procédure d'initialisation d'une liste (uniquement cette procédure-ci pour l'instant !). Enregistrez le fichier mais ne le fermez pas, vous y reviendrez plus tard. Créez un nouveau fichier main.c, et écrivez-y un programme principal minimal, qui teste la procédure d'initialisation.

Implantez progressivement les autres fonctions et procédures décrites dans le .h. Attention : vous ne devez PAS écrire toutes les procédures d'un coup. Testez vos sous-programmes au fur et à mesure,

en les appelant dans le main. Vérifiez que vous procédures fonctionnent aussi dans les cas particuliers : liste vide, liste ne contenant qu’un seul élément, etc.