TP 1 Algorithmique Avancée : Listes Chaînées

Serigne A. Gueye

CERI - Licence 2 Informatique

Exercice 1.

On considère les structures de données suivantes :

```
struct   Maillon
{
         int info;
         Maillon * suiv;
};

class    Liste
{
         Maillon * tete;
         public :
         Liste;
         ~ Liste();
}
```

1/ En s'appuyant sur ce modèle définir une structure de liste doublement chaînée.

2/ Coder les méthodes suivantes :

– $Liste()$: création d'une liste vide,

– $\sim Liste()$: destruction de la liste,

– $Vide()$: test de vacuité,

– $InsertionTete(x)$: insertion de l'élément de valeur x en tête de liste,

– $InsertionQueue(x)$: insertion de l'élément de valeur x en queue de liste,

– $Recherche(x)$: retourne l'adresse du premier élément de valeur x de la liste,

– $Suppression(A)$: suppression de l'élément de la liste se trouvant à l'adresse A.

Pour tester, vous lirez une suite quelconque de nombres à l'écran et créerez en mémoire la liste correspondante.

Exercice 2 [Concaténation de liste].

Soit L une liste chaînée (i.e une instance de $Liste$). Ecrire la méthode

$$Concatenation(T).$$

permettant de concatèner (ou coller) T à L. On ne fera aucune création de maillons. Le dernier élément de L doit faire référence au premier élément de T.

Exercice 3 [Recopie de liste].

Redéfinir le constructeur de la façon suivante :

$$Liste(T).$$

Ce nouveau constructeur doit permettre de créer une liste par recopie d'une liste T donnée en argument.