

Structures de Données
L2 2012 – Année 2008/2009
 Sans aucun document ni calculatrice
 Durée : deux heures

Ecrivez les algorithmes en langage algorithmique, langage C (ou facultativement en pseudo code formel), sans mixage. Définissez dans tous les cas les types des paramètres, spécifiez les données, les données modifiées, et les résultats, donnez les entêtes des fonctions.

Exercice 1 : Pretty prints (4 pts)

Ecrire un algorithme pour afficher une liste chaînée d'entiers sous la forme suivante :

Si la liste est linéaire :

▪ `Nom_liste: .->[elt1]->[elt2]->...->[eltn]->[eltlast] |`

Si la liste est circulaire :

▪ `Nom_liste: .->[elt1]->[elt2]->...->[eltn]->[eltlast] ...`

Conseil : apportez une attention particulière aux traitements spécifiques de début, milieu et fin de liste ainsi qu'aux cas spéciaux comme la liste vide ou singleton.

Exercice 2 : Lists constructors (2 pts)

- Ecrire un algorithme pour créer une liste d'entiers à partir d'un tableau d'entiers.
- Ecrire un algorithme pour dupliquer une liste existante.

Exercice 3 : Application pertinentes des SDD séquentielles (3 pts)

Remarque : 6 réponses à donner, points au prorata des réponses satisfaisantes.

Citer, de manière intelligible, au moins deux applications concrètes et pertinentes, de nature technique (c'est-à-dire non métier) respectivement de :

- la **file**, qui ne soient ni le buffer (tampon), ni la gestion générale de la temporisation de flux d'information.
- la **pile**, qui ne soient ni la simulation de la récursivité, ni la gestion générale de la mémorisation d'états.
- la **liste circulaire**.

Exercice 4 : Le classique qui ne vous prend pas par surprise (2 pts)

Ecrire un algorithme pour calculer la somme de 1 à $n \in \mathbb{N}^*$ en simulant la récursivité à l'aide d'une pile.

Remarque : les entêtes des opérations à utiliser pour cet exercice sont fournis :

`void empiler (T val, pile* pp) ; int depiler (T* pval, pile* pp)`

Exercice 5 : Un facile pour faire un minimum de points (1 pt)

Ecrire un algorithme pour inverser le signe des éléments d'une liste circulaire d'entiers.

Exercice 6 : Comprendre et interpréter un énoncé pour ce qu'il est (2 pts)

Ecrire un algorithme qui permet de réaliser le chaînage inverse de l'instance d'une liste doublement chaînée qui n'a été que simplement chaînée.

Conseil : ne perdez pas trop de temps sur cet exercice : revenez-y après le QCM qui suit, s'il vous en reste (du temps).

3/2

Exercice 8 : Questions abstraites sur les types abstraits de données (3 pts)

Entourez la réponse de votre choix (Conseil : commencez au crayon à papier).
Remarque : les matheux ont plus de chance d'emporter les 3 pts.

Une liste est elle forcément chaînée ?

~~OUI~~

~~NON~~

Une file est elle une liste ?

~~OUI~~

NON

Une pile est-elle une liste ?

~~OUI~~

NON

Une liste circulaire est-elle forcément doublement chaînée ?

OUI

~~NON~~

Une liste est-elle un arbre binaire ?

~~OUI~~

~~NON~~

Une liste est-elle un arbre équilibré ?

OUI

~~NON~~

Un arbre binaire est-il une liste ?

~~OUI~~

~~NON~~

Tout arbre peut-il est dénoté (représenté) par une liste linéaire ?

OUI

~~NON~~

4/3

Exercice 9 : Questions d'implantations en C (2 pts)

Entourez la réponse de votre choix (Conseil : commencez au crayon à papier).

Une fonction *modify* est chargée de modifier deux listes l1 et l2.

Le code suivant peut-il effectuer cette modification ?

```
int main()
```

```
{
    list l1 = NULL;
    list l2 = NULL;
    l2 = modify(&l1);
}
```

→ +2/3 (tous)

OUI

~~NON~~

Le code suivant peut-il modifier les listes l1 et l2 ?

```
int main()
```

```
{
    list l1 = newList(1);
    list l2 = newList(2);
    modify(l1, l2);
}
```

OUI

NON

Le code suivant peut-il modifier les listes l1 et l2 ?

```
int main()
```

```
{
    list l1 = NULL;
    list l2 = NULL;
    modify(&l1, l2);
}
```

OUI

~~NON~~

1

Exercice 10 : Questions d'attention au dernier cours (1 pts)

Un seul choix possible.

Que signifie AVL ?

- Athena Vortex Lattice
- Arbre Vertical Linéaire
- Andelson-Velskii and Landis
- Addison Wesley Laboratories

