Travaux Dirigés d'algorithmique n°4

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

Listes chaînées (suite)

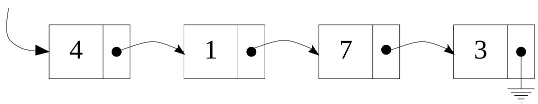
Nous implémenterons dans ce TD quelques opérations supplémentaires sur les listes chaînées.

Nous reprenons les définitions données dans la feuille précédente pour mettre en œuvre la notion de “liste chaînée d'entiers”.

```
typedef struct Cellule
        { int Val;                     /* donnee stockee : un entier.       */
          struct Cellule * PtrSuiv;    /* pointeur sur la cellule suivante. */
        } StrCellule;                  /* definition d'un nouveau type.     */

typedef StrCellule * PtrCellule;       /* definition d'un nouveau type.     */
```

Ainsi, la liste d'entiers $(4, 1, 7, 3)$ est représentée par la liste chaînée d'entiers suivante :



▸ **Exercice 1.** ***(Duplication d'une liste)***

- *Écrire une fonction* `DupListe` *prenant en argument une liste chaînée référencée par le pointeur* `PtrDebut`*. La fonction retournera une liste chaînée (totalement indépendante de la liste originale) qui sera une recopie exacte de la liste chaînée de départ.*
- *Réécrire la fonction précédente de telle sorte que la valeur retournée par la fonction soit VRAI ou FAUX selon que l'opération s'est bien passée ou pas. La liste de départ et la liste construite seront donc passées toutes deux en argument de la fonction.*

▸ **Exercice 2.** ***(Libération d'une liste)*** *Écrire une fonction* `LibereListe` *qui libère tout l'espace mémoire occupé par une liste chaînée.*

▸ **Exercice 3.** ***(Inversion d'une liste)*** *Écrire une fonction* `Miroir` *qui renvoie l'image miroir de la liste chaînée passée en paramètre. Cette fonction ne doit pas détruire la liste chaînée d'origine mais créer une nouvelle liste totalement indépendante.*

▸ **Exercice 4.** ***(Inversion d'une liste sur place)*** *Écrire une fonction* `MiroirSurPlace` *qui renvoie l'image miroir de la liste chaînée passée en paramètre. Cette fonction ne doit pas créer de nouvelles cellules. Elle modifiera simplement le chaînage de la liste chaînée d'origine.*