Travaux Dirigés d'algorithmique n°7

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

Piles et récursivité

Dans cette séance de travaux dirigés, nous allons nous intéresser au parcours de listes chaînées. L'affichage des éléments d'une liste, dans l'ordre du chaînage, peut naturellement s'écrire de manière itérative. Par contre, nous verrons que l'affichage dans l'ordre inverse nécessite l'utilisation d'une nouvelle structure de donnée : la pile. Nous utiliserons ensuite les piles pour réaliser une petite calculatrice qui évalue des expressions arithmétiques écrites en notation postfixée.

▸ **Exercice 1.** *On considère la structure de liste chaînée suivante :*

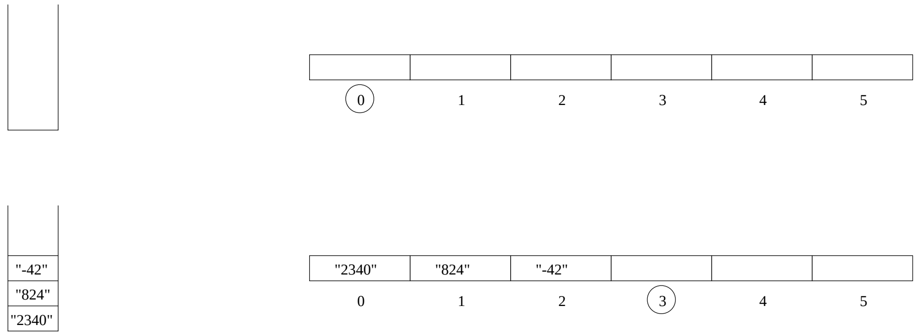
```
typedef struct Cellule {
   int Val;                     /* donnee stockee : un entier.       */
   struct Cellule * PtrSuiv;    /* pointeur sur la cellule suivante. */
} StrCellule;                   /* definition d'un nouveau type.     */

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

- *Écrire une fonction* `Affiche` *qui prend en argument une liste chaînée* `l` *et qui affiche les éléments de* `l` *dans l'ordre du chaînage.*
- *Écrire une fonction récursive* `AfficheInvRec` *qui prend en argument une liste chaînée* `l` *et qui affiche les éléments de* `l` *dans l'ordre inverse du chaînage.*

▸ **Exercice 2.** *Une* pile *(d'entiers, par exemple) est un type abstrait qui fonctionne comme une pile d'assiettes : on ne peut accéder qu'à l'élément du dessus. Les opérations qu'on peut faire sur une pile d'entiers sont :*

- *mettre un entier sur le sommet de la pile ;*
- *récupérer la valeur de l'entier qui est sur le sommet de la pile, et retirer cet entier de la pile.*

L'objet de cet exercice est l'implémentation des piles. Par soucis de simplification on suppose que le nombre d'éléments d'une pile est majoré par une constante `MAXPILE`*, et les éléments de la pile seront stockés dans un tableau de* `MAXPILE` *éléments.*



- *Donner un type C pour représenter les piles dont les éléments sont du type* `EltPile`.
- *Écrire une fonction* `InitPile` *qui initialise une telle pile.*
- *Écrire une fonction* `EstVidePile` *qui teste si une telle pile est vide.*
- *Écrire une fonction* `Empiler` *qui ajoute un élément sur le sommet d'une telle pile. Cette fonction retourne un entier qui indique si oui ou non l'opération s'est bien passée.*
- *Écrire une fonction* `Depiler` *qui effectue l'opération inverse d'*`Empiler`*. Cette fonction retourne un entier qui indique si oui ou non l'opération s'est bien passée, et met la valeur de l'entier dépilé dans un entier passé en argument à la fonction par adresse.*

▸ **Exercice 3.** *On souhaite maintenant réaliser, de manière itérative, l'affichage des éléments d'une liste dans l'ordre inverse du chaînage. Pour cela, le plus efficace est d'utiliser une pile (pourquoi ?). Concevoir un algorithme simple pour effectuer cette opération et l'implémenter.*

▸ **Exercice 4.** *On souhaite maintenant réaliser en C une calculatrice en notation postfixée (l'opérateur se trouve après les opérandes), qui dispose des opérations usuelles* $+, -, /, *$. *Pour simplifier les opérations de saisie au clavier on se limitera aux chiffres. Par exemple, l'expression* $123 * 4 + +$ *vaut* $1 + ((2 * 3) + 4) = 11$.

Écrire une fonction qui prend en argument une chaine de caractères représentant une expression arithmétique en notation postfixée et qui retourne sa valeur. Il faut employer une pile, dont l'utilisation est illustrée par l'exemple suivant. On veut évaluer l'expression postfixée $123 * 4 + +$. *Au départ la pile est vide. On lit le premier caractère de l'expression. C'est le chiffre 1, on le met dans la pile. On passe au caractère suivant, c'est encore un chiffre qu'on empile, et pareil pour le suivant. À la lecture du* $*$, *la pile contient* $1, 2, 3$ *(le sommet de la pile est à la droite). On voit qu'on veut faire un produit, on dépile le 3, puis le 2, on en fait le produit et on met le résultat sur la pile, qui contient donc* $1, 6$. *On continue jusqu'à ce qu'on arrive en fin d'expression. La pile ne contient alors plus qu'un entier, 11, qui est la valeur de l'expression.*