Travaux Dirigés d'algorithmique n°7

Cours d'Informatique de Deuxième Année

—Deug SCIENCES M3-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*.

▸ **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.    */
```

- *Ecrire une fonction* `Affiche` *qui prend en argument une liste chaînée* `l` *et qui affiche les éléments de* `l` *dans l'ordre du chaînage ;*
- *Ecrire 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 est une structure de donnée où l'insertion et la suppression d'un élément de type* `EltType` *se fait toujours du même côté. Nous proposons donc d'implémenter une pile à l'aide d'un tableau d'éléments de type* `EltType`*. Il est alors indispensable de garder la position du* sommet *de la pile dans une variable de type entière. Nous définissons ainsi le type suivant :*

```
typedef struct
{ EltType Tab[MAX];             /* tableau qui contient les elements   */
  int Sommet;                   /* sommet de la pile                   */
} PILE;

PILE p ;                        /* p est une variable globale          */
```

- *Ecrire une fonction* `InitPile` *qui initialise la pile* `p` *;*

- *écrire une fonction* `EstVidePile` *qui teste si la pile* `p` *est vide ;*
- *écrire une fonction* `Empiler` *prenant en argument un élément* `e`*, de type* `EltType`*, et qui empile* `e` *dans la pile* `p` *;*
- *écrire une fonction* `Depiler` *qui effectue l'opération inverse d'empiler. Cette fonction retourne l'élément que l'on a "enlevé".*

Exercice 3. *On souhaite maintenant réaliser, de manière itérative, l'affichage des éléments d'une liste dans l'ordre inverse du chaînage. Pour cela, il est nécessaire d'utiliser une* pile *(pourquoi ?).*

Concevoir un algorithme simple pour effectuer cette opération et l'implémenter.