Chaînes de caractères

Contrairement à ceux des autres séries, les exercices de cette feuille illustrent surtout une certaine manière de traiter les chaînes de caractères caractéristique de C, mais dont la transposition à d'autres langages se révèle difficile.

1 Ecrivez la fonction

```
int strlen(char *s);
```

qui détermine et renvoie le nombre de caractères d'une chaîne de caractères.

2 Ecrivez la fonction

```
int strcmp(char *s1, char *s2);
```

définie par :

$$strcmp(a,b) = \begin{cases} une\ valeur\ négative, si\ a < b \\ 0, si\ a = b \\ une\ valeur\ \ positive, si\ a > b \end{cases}$$

l'ordre sur les chaînes étant l'ordre lexicographique habituel.

3 Ecrivez la fonction

```
char *strcpy(char *dest, char *srce);
```

qui copie la chaîne *srce* dans l'espace pointé par *dest* et rend comme résultat l'adresse de cet espace.

4 Ecrivez la fonction

```
char *strcat(char *s1, char *s2);
```

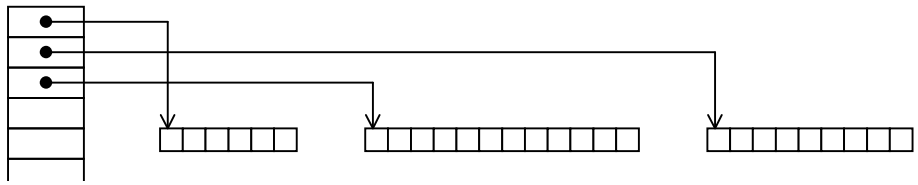
qui rend la chaîne s_1 concaténée avec s_2 (s_1 et s_2 sont mises bout à bout), sans recopie de s_2. Quel avertissement doit-on faire aux utilisateurs de cette fonction ? Illustrez votre réponse par une mauvaise et une bonne utilisation de *strcat*.

5 On souhaite disposer d'un programme qui acquiert et mémorise un texte, formé de lignes lues au clavier. L'utilisateur frappe sur la touche « Entrée » à la fin de chaque ligne. Une ligne réduite au mot *FIN* termine le texte.

On supposera qu'une ligne ne dépasse jamais 200 caractères, et qu'il y a au plus 1000 lignes. Pour la lecture d'une ligne, on utilisera la fonction *gets*. A la fin de la lecture, votre programme affichera les lignes lues.

A. Ecrivez ce programme en mémorisant les lignes dans un tableau rectangulaire de caractères.

B. Ecrivez une deuxième version du programme, dans laquelle on mémorise les lignes dans un tableau de chaînes, selon la structure suggérée par la figure suivante :



Soit *n* le nombre de lignes lues, *m* leur longueur moyenne. Evaluez l'encombrement des structures de données dans l'un et l'autre cas.

6 Dans le cadre de l'exercice précédent, on suppose que l'utilisateur peut taper plusieurs fois la même ligne. Modifiez votre programme pour que

A. Chaque ligne apparaisse dans le tableau autant de fois qu'elle a été tapée, mais en occupant aussi peu de mémoire que possible.

B. Chaque ligne apparaisse une seule fois dans le tableau.

7 Modifiez le programme précédent pour qu'il ordonne le tableau de mots lus, avant de les afficher. Vous pouvez adapter à ce problème n'importe quelle méthode de tri que vous connaissez par ailleurs.

☺☹☺