TP n° 4

I3 : Tableaux et allocations mémoire

Exercice 1 [Moyenne des valeurs d'un tableau]

1. Écrivez une procédure C qui prend en paramètres un tableau d'entiers à une dimension et sa taille, et initialise les valeurs contenues dans ce tableau. Vous pourrez utiliser une initialisation aléatoire entre 0 et 100, en utilisant la fonction `rand()` (utilisation à voir dans l'aide en ligne, attention à ne pas oublier d'initialiser le générateur de nombres aléatoires).
2. Écrivez une procédure C qui prend en paramètres un tableau d'entiers à une dimension et sa taille, et affiche les valeurs contenues dans ce tableau.
3. Écrivez une fonction C qui prend en paramètres un tableau d'entiers à une dimension et sa taille, et calcule la moyenne et l'écart type des valeurs contenues dans ce tableau.
   L'écart type est la moyenne de l'écart des valeurs avec la moyenne, soit en notant x_i les valeurs du tableau et $\bar{x}$ la moyenne des valeurs du tableau :
   $\sigma = \sqrt{\sum_{i=0}^{N-1} \frac{(x_i - \bar{x})^2}{N-1}}$
   Vous pourrez utiliser la fonction `sqrt()` de la bibliothèque de mathématiques (en-têtes : math.h, bibliothèque : libm).
4. Écrivez un programme en C qui demande à l'utilsiateur de saisir la taille d'un tableau sous forme d'un entier, alloue et initialise ce tableau, affiche les valeurs contenues dans ce tableau et leur moyenne.

Exercice 2 [Carré magique]

Un carré (tableau d'entiers de N lignes et N colonnes initialisées) est dit magique lorsque la somme des éléments d'une ligne, d'une colonne ou d'une diagonale quelconque est toujours égale au même nombre.

Par exemple le carré suivant est magique :

15	8	1	24	17
16	14	7	5	23
22	20	13	6	4
3	21	19	12	10
9	2	25	18	11

1. Écrivez une procédure en C qui prend en paramètres un tableau à deux dimensions, les deux dimensions, et affiche les valeurs du tableau ligne par ligne.
2. Écrivez une fonction en C qui vérifie si un carré est magique. Le tableau (à deux dimensions) et ses dimensions sont passés en paramètre et la fonction retourne 0 si le carré est magique, 1 sinon. La fonction doit également retourner 1 si le tableau n'est pas un carré (c'est-à-dire si ses dimensions ne sont pas égales).
3. Écrivez un programme C qui demande à l'utilisateur d'entrer la largeur N du carré, alloue le tableau en mémoire et demande à l'utilisateur de saisir une à une les valeurs du tableau.
4. Affichez les valeurs contenues dans le tableau saisi.
5. Vérifiez si le tableau est un carré magique.

Exercice 3 [Concaténation de chaînes de caractères]

1. Écrivez une procédure C qui demande à l'utilisateur de saisir une chaîne de caractère de longueur quelconque terminée par un point et retourne un pointeur vers cette chaîne de caractères, à la manière de ce qui a été vu dans le TD 4. Pour cela vous utiliserez la fonction C `fgets()`, qui permet de lire un certain nombre de caractères venant sur un flux (ici, l'entrée standard `stdin`). Attention, `fgets()` ajoute le caractère `\0` à la fin de la chaîne saisie.
2. Modifiez la fonction précédente pour qu'elle prenne en paramètre la taille minimum de tableau, qui sera remplacée par la taille réelle de la chaîne de caractères en sortie de la fonction (en comptant le point final).
3. Écrivez un programme en C qui demande à l'utilisateur de saisir deux chaînes de caractères l'une après l'autre, puis les concatène dans une troisième chaîne de caractères en retirant les points finaux. Vous écrirez la concaténation des deux chaînes de caractères dans une fonction séparée qui prend en arguments les deux chaînes de caractères à concaténer et leurs tailles, et retourne le résultat de la concaténation.

Exercice 4 [Longueur d'une chaîne de caractères]

Écrivez une fonction C qui prend en paramètre un pointeur vers le début d'une chaîne de caractères et retourne la longueur de cette chaîne sous forme d'un entier *sans utiliser les fonctions de string.h.*

Exercice 5 [Copie d'une chaîne de caractères]

1. Écrivez une fonction C qui copie une chaîne de caractères dans un tampon mémoire nouvellement alloué et retourne un pointeur vers la copie *sans utiliser les fonctions de string.h.*
2. Écrivez une fonction C qui copie les N premiers caractères d'une chaîne de caractères dans un tampon mémoire nouvellement alloué et retourne un pointeur vers la copie *sans utiliser les fonctions de string.h.* Si la chaîne passée en paramètre a une longueur inférieure à N, on la copie intégralement dans un tampon de la bonne taille (inférieure à N);

Exercice 6 [Comparaison de deux chaînes de caractères]

1. Écrivez une fonction C qui prend en paramètre deux pointeurs vers le début de chaînes de caractères `str1` et `str2` et les compare *sans utiliser les fonctions de string.h.* La fonction doit retourner 0 si les chaînes sont identiques, 1 si le premier caractère différent de `str1` est supérieur à celui de `str2`, -1 si le premier caractère différent de `str2` est supérieur à celui de `str1`. Si les chaînes ne sont pas de la même longueur, la fonction retourne 1 si `str1` est plus longue et -1 si `str2` est plus longue.
2. Écrivez une fonction C qui compare les N premiers caractères de `str1` et `str2`. Si l'une des chaînes de caractères a une longueur inférieure à N, on applique les mêmes règles que pour la question précédente.

Exercice 7 [Découpage de chaînes de caractères]

Écrivez une fonction C qui découpe une chaîne de caractères en segments, chaque segment étant délimité par un jeton passé en paramètre.

Par exemple, la chaîne `"titiGEtotoGEtataGEtutu"` peut-être découpée par le jeton `"GE"` et donnera les quatre chaînes de caractères `"titi"`, `"toto"`, `"tata"` et `"tutu"`.

La fonction doit prendre en paramètre un pointeur vers la chaîne de caractères à découper et un pointeur vers la chaîne de caractères constituant le jeton. Elle retourne un pointeur vers un tableau de chaînes de caractères dont chaque élément est un segment de la chaîne initiale et le dernier élément est le pointeur NULL.