Séance 2 : Tableaux, chaînes de caractères et écriture binaire

L2 – Université Côte d'Azur

Exercice 1 — Lire et ranger dans un tableau

Écrivez un programme, qui implémente les deux procédures suivantes :

1. la procédure `saisie_tab` lit sur l'entrée standard `n` entiers et les range dans un tableau `tab` d'entiers, `n` et `tab` étant passés en paramètre ;
2. la procédure `affiche_tab` écrit sur la sortie standard les `n` premiers éléments d'un tableau `tab` d'entiers, `n` et `tab` étant passés en paramètres.

Que se passe-t-il si on saisit plus d'entiers que le tableau peut contenir d'éléments ? Et si on essaie d'afficher plus d'éléments que ne contient le tableau ? Expliquez.

Exercice 2 — Travailler avec des chaînes de caractère

1. Écrivez un programme, qui contient la fonction `chercher_caractere` : cette fonction renvoie vrai si un caractère `c` apparaît dans une chaîne `s`, faux sinon (`c` et `s` étant passés en paramètre de la fonction).
2. Écrivez un programme, qui contient la fonction `multiple_de_3` : cette fonction renvoie vrai si une chaîne de caractères (passée en paramètre) contenant un nombre positif (supposé valide syntaxiquement) est multiple de trois, faux sinon.
   Rappel : Pour qu'un nombre soit multiple de trois, il suffit que la somme des chiffres de ce nombre soit un multiple de trois. 12345678 multiple de 3 ? $1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = 36, 3 + 6 = 9$, donc multiple de 3.

Exercice 3 — Travailler avec des bits

1. Écrivez le code des fonctions suivantes (`x` est une variable, et `pos` est la position du bit à modifier) :
   (a) `int get_bit(unsigned int x, int pos)` : retourne la valeur du bit situé à la position `pos`,
   (b) `unsigned int set_bit(unsigned int x, int pos)` : retourne un entier qui est égal à l'entier `x` pour lequel on a mis le bit à la position `pos` à 1,
   (c) `unsigned int clear_bit(unsigned int x, int pos)` : retourne un entier qui est égal à l'entier `x` pour lequel on a effacé le bit à la position `pos`,
   (d) `unsigned int toggle_bit(unsigned int x, int pos)` : retourne un entier qui est égal à l'entier `x` pour lequel on a inversé le bit à la position `pos`,
   (e) `unsigned int define_bit(unsigned int x, int pos, int bool)` : retourne un entier qui est égal à l'entier `x` pour lequel on a défini le bit à la position `pos` suivant le résultat de `bool`.
2. Écrivez un programme, qui écrit sur la sortie standard l'équivalent binaire d'un nombre entier passé en paramètre. Exemples : 8 s'écrira 1000, 13 s'écrira 1101, etc.
3. Écrivez un programme qui procède à la multiplication binaire de deux nombres.
4. Écrivez un programme qui procède à la division binaire de deux nombres.
5. Écrivez un programme qui effectue le miroir d'un entier non signé sur 32 bits. Par exemple si les entiers sont représentés sur 4 bits : 8 s'écrira 1000 son miroir est 0001 = 1 (question de contrôle terminal en 2013).
6. Écrivez un programme qui isole les 10 bits les plus à gauche, les 10 bits les plus à droite et les 12 bits restants au milieu d'un entier non signé.