Introduction à l'informatique

TD4 : Passage par valeur et passage par référence

Rappel : toutes les fonctions doivent être documentées

Exercice 1 (Fonction mystère)**.**
On considère le programme suivant :

```
void mystere(int a, int &b, int &c) {
    int tmp;
    c = a + b;
    tmp = a;
    a = b;
    b = 2 * tmp;
}

int main()
{
    int one, two, three;
    one = 5;
    two = 10;
    three = 15;

    mystere(one, two, three);
    cout << one << " - " << two << " - " << three << endl;
    mystere(two, one, three);
    cout << one << " - " << two << " - " << three << endl;
    mystere(three, two, one);
    cout << one << " - " << two << " - " << three << endl;
    mystere(two, three, one);
    cout << one << " - " << two << " - " << three << endl;

    return 0;
}
```

Exécuter pas à pas ce programme, qu'affiche-t-il ?

Exercice 2 (Points 2D et triangle rectangle)**.**

(1) Écrire une fonction qui calcule la distance entre deux points à partir de leur coordonnées x et y dans le repère cartésien.

(2) Écrire une fonction qui calcule les coordonnées cartésiennes du milieu de deux points à partir de leur coordonnées x et y dans le repère cartésien.

(3) Écrire un fonction de test qui vérifie le bon fonctionnement de ces deux fonctions.

(4) Écrire un programme principal qui initialise 6 nombres réels tels que :

- xa et ya soit les coordonnées cartésiennes du point A (0, 2),
- xb et yb soit les coordonnées cartésiennes du point B (-2, 0),

- xc et yc soit les coordonnées cartésiennes du point C (3, -1),

Vérifier que ABC forme un triangle rectangle et l'afficher si c'est bien le cas.
Puis, le programme principal calculera les coordonnées de M le milieu de BC et vérifiera que les distances entre AM, BM et CM sont bien égales, en l'affichant si c'est bien le cas.

Exercice 3 (Fonctions sur les chaînes de caractères)**.**
Dans cette exercice, il vous sera demander de créer plusieurs fonctions manipulant des chaînes de caractères. Pour vous aider, vous pourrez utiliser les deux fonctions suivantes de la librairie standard sur les chaînes de caractères :

```
string s = "bonjour" ; // soit une chaîne de caractères s

int l = s.length()     // retournera le nombre de caractères de s
                       // ici, l prendra la valeur 7

char c1 = s.at(0);     // retournera le caractère à la position donnée
char c2 = s.at(3);     // le premier caractère est à l'indice 0
                       // ici, c1 prendra la valeur 'b' et c2 la valeur 'j'
```

(1) Écrire une fonction qui teste si une chaîne de caractères contient au moins une lettre majuscule. Cette fonction s'arrêtera dès qu'elle aura trouvé une première lettre majuscule.

(2) Écrire une fonction qui teste si une chaîne de caractères contient que des lettres majuscules.

(3) Écrire une fonction qui compte le nombre d'espace dans une chaîne de caractères.

(4) Écrire une fonction qui transforme les lettres majuscules en lettres minuscules d'une chaîne de caractères. Écrire une version de cette fonction avec un passage par valeur et une version avec un passage par référence.
Les `char` *sont en fait codés par des entiers (c'est le code ASCII). Les lettres majuscules sont comprises entre 65 et 90. Les minuscules entre 97 et 122. On passe des majuscules aux minuscules en ajoutant 32 et en stockant le résultat dans un nouveau* `char`*.*

(5) Écrire une fonction qui enlève les espaces une chaîne de caractères. Écrire une version de cette fonction avec un passage par valeur et une version avec un passage par référence.

(6) Écrire une fonction qui teste si une chaîne de caractères est un palindrome, c'est-à-dire un texte ou un mot dont l'ordre des lettres reste le même qu'on le lise de gauche à droite ou de droite à gauche, comme dans la phrase *« Esope reste ici et se repose »* ou encore *« La mariee ira mal »*. Pour effectuer le test, il faudra enlever les majuscules et les espaces de la chaîne de caractères.

(7) Écrire une fonction qui compte le nombre de voyelles, de consonnes et de caractères autres. On essayera de trouver une solution intelligente qui évite une comparaison avec toutes les lettre de l'alphabet. On pourra, par exemple, créer deux fonctions qui regarde respectivement si un caractère est une lettre et si un caractère est une voyelle.

(8) Écrire une fonction de tests qui testera les toutes fonctions précédentes.

Exercice ♠ 4 (Dates)**.**

(1) Écrire une fonction qui prend en entrée une date sous la forme de trois entiers jour / mois / année, et teste si c'est une date valide. Pour l'instant, on ignore les années bissextiles. Par exemple :

— `date_valide(28, 5, 1973)` renvoie `true`

— `date_valide(31, 2, 2015)` renvoie `false`

(2) Écrire une fonction qui prend en argument une date valide sous la forme de trois entiers et affiche la date du lendemain. Par exemple :
`jour_suivant(18, 12, 2021)` affiche `Le jour suivant est le 19 12 2021`.

(3) Reprendre la question 1 en considérant les années bissextiles (une année est bissextile si elle est divisible par 4, mais pas 100 sauf si elle est divisible par 400).

(4) Reprendre la question 2 pour que le programme n'affiche pas mais modifie les arguments passés par référence.

(5) Écrire une fonction qui prend en argument une date valide et renvoie le jour de la semaine de cette date.

(6) Combien de 1er du mois ont été des dimanches au XX e siècle ? (on donne que le 1er janvier 1901 était un mardi).
Pour les curieux cela correspond au problème Projet Euler 19 qui donne plein de problèmes pouvant être résolus informatiquement.