TP -passage de paramètres par adresse

Objectif(s)

✯ Comprendre et mettre en oeuvre les appels de fonctions avec des paramètres passés par adresse.

Exercices obligatoires

Exercice 1 – Pour s'entrainer

Un étudiant doit écrire une fonction permettant d'échanger le contenu de deux variables et un programme principal pour constater que sa fonction fonctionne correctement.

Il a écrit le programme suivant

```
#include <stdio.h>

void echange(int a, int b) {
    int tmp;
    tmp=a;
    a=b;
    b=tmp;
}

int main() {
    int a,b;

    a=16; b=4;
    printf("avant echange a=%d b=%d\n", a,b);
    echange(a,b);
    printf("après echange a=%d b=%d\n", a,b);
}
```

La trace d'exécution est la suivante :

```
avant echange a=16 b=4
après echange a=16 b=4
```

alors que le résultat attendu est le suivant :

```
avant echange a=16 b=4
après echange a=4 b=16
```

Question 1

Corrigez le programme en utilisant le passage par adresse dans la fonction `echange`

Exercice 2 – Résolution d'une équation du second degré

Cet exercice reprend l'exercice effectué en TD, sauf que nous allons aussi traiter le cas $a = 0$. Pour rappel :

L'objectif de l'exercice est d'écrire une fonction `solve2d` qui prend en entrée 5 paramètres a, b, c, $x1$ et $x2$ de type `double` ou `double *`.

La fonction `solve2d` résout l'équation $ax^2 + bx + c = 0$ et retourne le nombre de solutions de l'équation.

Si a vaut 0 et b vaut 0, l'équation n'a pas de solution [1]. Si a vaut 0 mais pas b, la solution est $\frac{-c}{b}$.

Si $a \neq 0$, pour résoudre l'équation, il faut calculer le discriminant : $\Delta = b^2 - 4ac$

Le nombre de solutions est donné par le signe de Δ.

1. sauf si c vaut aussi 0, cas que l'on ne traite pas).

— $\Delta > 0$, il y a 2 racines à l'équation : $x1 = \frac{-b-\sqrt{\Delta}}{2a}, x2 = \frac{-b+\sqrt{\Delta}}{2a}$

— $\Delta = 0$, il y a 1 racine à l'équation : $x1 = \frac{-b}{2a}$

— $\Delta < 0$, il y a 0 racine réelle.

Lorsque la fonction se termine, plusieurs cas sont possibles :

— L' équation a 2 solutions : les paramètres $x1$ et $x2$ contiennent les 2 solutions de l'équation

— L' équation a 1 solution : le paramètre $x1$ la solution de l'équation. $x2$ a une valeur quelconque.

— L'équation n'a aucune solution : les paramètres $x1$ et $x2$ ont une valeur quelconque.

Quelques traces d'exécution sont données pour exemple. Les affichages et les saisies de valeurs doivent être fait dans la fonction `main`, les calculs dans la fonction `solve2d`.

```
$ solve2d
Saisir une valeur différente de 0 pour a : 1
Saisir une valeur pour b : 2
Saisir une valeur pour c : 1
une solution -1.000000e+00

$ solve2d
Saisir une valeur différente de 0 pour a : 1
Saisir une valeur pour b : -3
Saisir une valeur pour c : 2
deux solutions 1.000000e+00 et 2.000000e+00

$ solve2d
Saisir une valeur différente de 0 pour a : 1
Saisir une valeur pour b : 1
Saisir une valeur pour c : 1
Pas de solution
```

Nous vous rappelons au passage que la fonction `sqrt` est contenue dans la bibliothèque `math.h`, que l'on peut utiliser à condition d'ajouter l'option `-lm` en fin de ligne de compilation.

Renforcements

Exercice 3 – Le temps qui passe

Question 1

Écrivez une fonction `seconde_vers_horaire` qui convertit un temps écoulé exprimé en secondes depuis minuit en heures, minutes et secondes. La fonction prend en entrée 4 paramètres :

ecoule : nombre de secondes depuis le début de la journée (entier compris en 0 et 86400)

heure : heure de la journée

minute : minute de la journée

seconde : seconde de la journée

Par exemple, pour un temps écoulé de 47531s, la fonction permettra d'indiquer qu'il correspond à 13h12min et 11s, en mettant à jour les variables correspondantes passées en paramètre. [2]

Écrivez une fonction de test que vous appelerez dans le `main`.

Question 2

2. **indice :** la fonction modulo pourrait vous aider.

Écrivez une fonction `une_seconde_plus_tard` qui prend en entrée un horaire (soit 3 valeurs) exprimé en heure, minute, seconde et calcule l'horaire une seconde plus tard. La fonction ne fera pas d'affichage, mais aura simplement un impact sur les variables passées en paramètre.

Écrivez une fonction pour tester votre fontion. La fonction demandera à l'utilisateur de saisir 3 valeurs pour les heures, les minutes et les secondes et affichera l'horaire une seconde plus tard. On supposera que la saisie des heures, minutes, secondes sera correcte.

Question 3

Écrivez une fonction qui fait le processus inverse de la question 1, à savoir prend les trois paramètres heure, minute et seconde et calcule le nombre de secondes que cela représente. Testez votre fontion.

Question 4

Écrivez une fonction `duree` qui prend en paramètres trois horaires hh :mm :ss (9 paramètres) et qui calculent le temps qui sépare les horaires 1 et 2 (en prenant éventuellement en compte un jour d'écart si le deuxième horaire est plus tôt que le premier). Le temps qui les sépare sera renvoyé par le 3e horaire.

Question 5

Écrivez une fonction `saisir_horaire` qui permet de faire une saisie d'horaire (au format hh :mm :ss) en vérifiant la validité des valeurs. L'horaire saisi est enregistré dans les variables passées en paramètre.