TP 2 : Mes premiers programmes en C

Notions : Buts : E/S, variables, boucles, if, limites du calcul numérique

---

Rappel : voir le document « comment développer un programme en TD sur le site tdinfo »
Pour réaliser vos programmes sur machine, il faut aller dans le répertoire concerné, éditer le fichier source C, compiler (et corriger si nécessaire) puis exécuter le programme. Si les tests ne sont pas tous corrects, il faut corriger et compiler et tester etc .....
Faites un répertoire par TD. Cela correspond à : Ouvrir un terminal (Menu Outils de système ou Menu accessoires selon la version)

1. Dans le terminal
   (a) Aller dans votre répertoire tdinfo : `cd tdinfo`
   (b) Aller dans le répertoire seance02 : `cd seance02`
   (c) Editer le fichier source C nommé `eq1.c` par exemple, à l'aide d'un éditeur de texte tel que atom : `atom eq1.c &`
2. Dans l'éditeur,
   (a) taper votre programme
   (b) sauver votre programme
3. Dans le terminal :
   (a) Compiler votre programme : `clang eq1.c -o eq1 -lm` ou `gcc eq1.c -o eq1 -lm` (-lm est présent pour la bibliothèque Mathématique)
   (b) Exécuter votre programme s'il n'y a pas d'erreur de compilation : `./eq1`
   (c) Sinon, corriger votre programme dans l'éditeur, sauver à nouveau (étape 2) et compiler dans le terminal. (étape 3)

1 Structures conditionnelles

1.1 Vérification de date

1. Ecrivez un programme qui lit une date au clavier sous la forme jour/mois/année et qui répond *correct* si la date est correcte, *incorrect* sinon
   Les contraintes sont les suivantes :
   — Le numéro du mois doit être compris entre 1 et 12.
   — La vérification du numéro du jour dépend du mois : 31/01/2020 est correct car janvier a 31 jours, mais 31/04/2019 est incorrect car avril n'a que 30 jours.
   — Mais aussi des années : attention aux années bissextiles. Ainsi, le 29/02/1999 n'est pas correct car 1999 n'est pas une annee bissextile.

   Remarque : la fonction `scanf` permet de lire des entrées formatées et retourne le nombre d'éléments correctement lus. Dans ce cas, si on souhaite une entrée du type 10/12/2020, on peut utiliser une instruction du type `scanf("%2d/%2d/%4d",&j,&m,&a)` qui imposera une entrée de la forme jj/mm/aaaa. Si la date entrée respecte ce format, la fonction retournera la valeur 3 (3 nombres lus). Sinon, elle retournera un nombre différent de 3. On peut bien sûr effectuer la lecture en utilisant 3 appels différents à la fonction `scanf` et en vérifiant chacun d'eux.
2. Facultatif : Dans le même ordre d'idée, calculez la date du lendemain.

2 Structures itératives

2.1 Plus ou Moins ?

Ecrivez un programme qui permet de jouer au jeu du plus ou moins. Il s'agit de deviner un entier aléatoire entre 1 et 1000,

Votre programme doit donc :

— demander à l'utilisateur une valeur comprise entre 1 et 1000.

— indiquer si le nombre saisi par l'utilisateur est plus grand ou plus petit que le nombre à deviner

— itérer ces 2 opérations jusqu'à ce que la réponse soit exacte.

Lorsque l'utilisateur a découvert le nombre, le programme affichera le nombre d'essais nécessaire pour y parvenir.

2.2 Spaghetti et probabilité

Quand un spaghetti se casse, il se coupe toujours en 3 morceaux. Pour que l'on puisse former un triangle avec les 3 morceaux, il faut que ces 3 morceaux respectent l'inégalité triangulaire : **le plus grand des morceaux est inférieur à la somme des 2 autres**.

Pour déterminer la probabilité qu'on puisse former un triangle avec les 3 morceaux du spaghetti, on peut générer des spaghettis virtuels que l'on coupe aléatoirement en 3 morceaux et vérifier si ce spaghetti coupé forme ou non un triangle. En générant un grand nombre (1000000 au moins) de ces spaghettis virtuels [1], on approche de la probabilité voulue, qui est $ln(2) - 1/2$ soit $0,193147$ [2]

Votre programme doit donc :

— Couper le spaghetti de longueur 1 aléatoirement en 2 morceaux de longueur l1 et l2 : pour cela, il suffit de tirer aléatoirement un nombre réel inférieur à 1 pour l1. Le morceau l2 est de longueur $l2 = 1 - l1$.

— Couper le morceau de longueur l2 aléatoirement en 2 morceaux de longueur l3 et l4 : pour cela, il suffit de tirer aléatoirement un nombre réel x inférieur à 1. Le morceau l3 est de longueur $l3 = x.(1 - l1)$. Le morceau de longueur l4 est de longueur $l4 = 1 - l3 - l1$.

— Déterminer le plus grand des 3 morceaux $l1, l3, l4$.

— Vérifier l'inégalité triangulaire et incrémenter le compteur des spaghettis *triangulisables*

— itérer ces opérations 1000000 fois.

— la probabilité est alors le quotient du compteur des spaghettis *triangulisables* sur le nombre total (ie 1000000)

Vous pourrez avoir besoin des fonctions suivantes :

— Génération aléatoire : `int rand()` : cette fonction retourne un nombre entier tiré au hasard entre `1` et `RAND_MAX`. Attention à la division entière pour obtenir un nombre réel inférieur à 1.

3 Séries et limites de suites

3.1 Nombre d'or et suite de Fibonacci

Au XIIIe siècle, Fibonacci, appelé aussi Léonard de Pise, celui là même qui avait amené en occident les chiffres hindous au retour d'un long séjour à Alger, tenta de résoudre le problème de la prolifération des lapins. Il voulait se rendre compte de la puissance de cette prolifération, la formuler, alors qu'elle dépassait très vite l'imagination. La myxomatose lui vint alors en aide...

Il partit d'un fait très rare dans l'espèce humaine : que des oncles et des neveux soient de même âge. Dans ce cas, possédant initialement un couple de lapins, combien de couples obtient-on en douze mois si chaque couple engendre tous les mois un nouveau couple à compter du second mois de son existence ?

1. ou bien venir avec des spaghettis réels
2. En insistant gentiment, votre prof de proba vous expliquera pourquoi

Il étudiait donc la suite :

$$\begin{cases} U_n = U_{n-1} + U_{n-2} \\ U_1 = 1 \\ U_0 = 0 \end{cases}$$

Cette suite donne aussi accès au nombre d'Or, qui en architecture et dans les arts graphiques, donne le rapport entre la longueur et la largeur du rectangle le plus agréable à l'œil. Ce nombre est la limite de la suite :

$$\begin{cases} V_n = \dfrac{U_n}{U_{n-1}} \\ V_0 = 0 \end{cases}$$

soit

$$\frac{1+\sqrt{5}}{2} = 1,6180339887498948482045868343656381177203091798057628621354486227052604621890244970720720 41.$$

Pour calculer une limite, il faut utiliser l'écart relatif entre 2 termes consécutifs et le comparer à une valeur faible, compatible avec la précision des nombres réels que vous utilisez (10^{-7} en float, 10^{-15} en double) c.-à-d. tester :

$$|V_n - V_{n-1}| > \epsilon.(|V_n| + |V_{n-1}|)$$

1. Ecrire le programme qui affiche à l'écran les n premiers termes de la suite U_n , $n > 1$ étant donné au clavier. Ce programme fait intervenir une boucle. En plus de l'indice de boucle et du nombre n, vous ne devez utiliser que 3 variables pour calculer les différents termes U_n de la suite. Ces trois variables (`a,b,c` par exemple) représenteront u_2, u_1, u_0 au départ de la boucle (itération 0), puis u_3, u_2, u_1 à l'itération suivante (itération 1), puis u_4, u_3, u_2 ... il suffit de modifier dans le bon ordre les valeurs de `a` et de `b` à la fin de la boucle, puisqu'il faut qu'au début de la 2^{ieme} itération de la boucle, `b` contienne u_3 et `a` contienne u_2 de manière à ce que `a+b` donne u_4 lors de cette itération. **Attention :** regardez attentivement ce qui se passe autour des valeurs de U_{44}. Cette suite est très rapidement croissante et positive en théorie.
2. Ecrire le programme qui calcule la limite de la suite V_n et donne le nombre d'or. La valeur absolue d'un réel x est obtenue par `fabs(x)`.

4 Quelques particularités du calcul numérique

4.1 Vitesse de convergence

Soient les 2 suites U_n et V_n définies par :

$$\begin{cases} U_0 = 1 \\ U_n = U_{n-1} \times \dfrac{2n \times 2n}{(2n-1) \times (2n+1)} \end{cases}$$

$$V_n = \sum_{i=0}^{n} \left(\frac{4}{8.i+1} - \frac{2}{8.i+4} - \frac{1}{8.i+5} - \frac{1}{8.i+6} \right) \cdot \left(\frac{1}{16} \right)^i$$

La suite U_n due à John Wallis en 1655 converge vers $\pi/2$ et la suite V_n due à David Bailey, Peter Borweinet et Simon Plouffe en 1995 converge vers π.

Les premières décimales de π sont :
3.14159265358979323846264338327950288419716939937510582097494459230781640628620899862803 48

1. Ecrire un programme calculant la limite des 2 suites. Attention au calcul de la suite U_n aux alentours de $n = 23200$: qu'observez vous ?
2. Comparer expérimentalement les vitesses de convergence et la précision de la limite de ces 2 suites.

4.2 Erreur de calcul

Soit x un nombre réel positif et les suites x_n et ρ_n définies par :

$$\begin{cases} x_0 = x \\ x_n = \sqrt{\sqrt{x_{n-1}}} \end{cases} \qquad \rho_n = \frac{x_{n+1} - 1}{x_n - 1}$$

La suite x_n converge vers 1 et la suite ρ_n converge vers $1/4$.

1. Ecrire un programme affichant les premiers termes des suites x_n et ρ_n.
2. Quelle observation pouvez vous faire ?