Université Paris 13 — Éléments d'informatique
Institut Galilée — PC-SPI
Licence 1 - 1er semestre

Travaux pratiques 4 : exercice noté, boucle *while* ; Expressions booléennes ; type de données

Le premier exercice du TP est noté. L'objectif de ce TP est de vous familiariser avec les notions d'itération et de boucles imbriquées et de vous familiariser avec les expressions booléennes et leur utilisation avec la structure de contrôle "while".

Vous allez mettre tous vos programmes écrits dans ce TP dans le répertoire TP4.

1. À partir du début de votre arborescence, créez le répertoire TP4 : `mkdir TP4`
2. Allez dans ce répertoire pour y mettre des fichiers : `cd TP4`

L'étape suivante est à répéter pour chaque nouveau programme (exo1, exo2 etc..) :

3. Créez un nouveau fichier source pour le langage C ou une nouvelle copie d'un programme existant.

   Création `gedit exo1.c &` (vous pouvez utiliser `emacs` ou `kwrite` au lieu de `gedit`)

   Copie Il est plus rapide de repartir d'une copie de votre programme `bonjour.c` du TP2 pour éviter de retaper tout le squelette. Dans le terminal :
   ```
   cp ../TP2/bonjour.c exo1.c
   gedit exo1.c &
   ```
   Vous pouvez-aussi ouvrir `bonjour.c` et utiliser la fonction *Enregistrer sous...* de votre éditeur mais attention à enregistrer la nouvelle copie dans le bon répertoire.

Vous pouvez utiliser à tout moment la commande `ls` (list directory) pour voir la liste des fichiers d'un répertoire.

Les trois étapes suivantes seront à répéter autant de fois que nécessaire pour la mise au point de chaque programme (apprenez à utiliser les raccourcis clavier).

4. Après avoir fini d'écrire votre programme, enregistrez le.
5. Créez un programme exécutable à partir de votre fichier source :
   `gcc -Wall exo1.c -o exo1.exe`
6. Quand l'étape précédente a réussi (il faut lire attentivement les messages affichés), exécutez le programme pour vérifier qu'il fonctionne : `exo1.exe` (ou `./exo1.exe`).

1 ☛ Exercice noté (durée : 45 minutes)

Récupérez l'énoncé de l'exercice noté sur l'ENT.

Remarques :

— La notation tiendra compte de l'indentation, des commentaires, de la compilation et de la réponse au problème.

— Si vous êtes en binôme, indiquez le en commentaire dans le code.

— **Vous déposerez sur l'ENT le fichier source (fichier `.c`)**
ATTENTION : Le TP noté est limité au 45 premières minutes du TP. Passée cette période, vous ne pourrez plus rendre le TP noté.

— Seuls les étudiants présents en TP seront évalués sur cet exercice.

2 Fibonacci

1. ✎✎ Écrire un programme qui détermine la $n^{\text{ième}}$ valeur u_n (n étant fourni en donnée) de la "suite de Fibonacci" définie comme suit (voir aussi le TD4) :
   $u_1 = 1$
   $u_2 = 1$
   $u_n = u_{n-1} + u_{n-2}$ pour $n > 2$
2. ✎✎✎ En utilisant le programme précédent, écrire un deuxième programme calcule la valeur u_n immédiatement inférieure ou égale à une valeur m fournie par l'utilisateur, et l'affiche avec son rang.

3 ✎ Le nombre secret

Nous voulons programmer le jeu du nombre à découvrir. Le joueur doit deviner un nombre secret choisit par l'ordinateur entre 0 et `NB_MAX` (une constante du programme). S'il propose un nombre trop grand, l'ordinateur lui répond "Plus petit", s'il propose trop petit, l'ordinateur lui répond "Plus grand". Dans ces deux cas, il est invité à proposer un autre nombre. Le jeu s'arrête quand il devine juste. Un exemple d'exécution de ce jeu pourrait être :

```
Votre choix ?
8
Plus petit.
Votre choix ?
4
Plus petit.
Votre choix ?
2
Vous avez trouvé le nombre secret.
```

1. Proposez un algorithme en français pour le jeu.
2. Traduisez-le en langage C (dans le fichier `nombre_secret.c`) et exécutez-le.
3. Pourquoi préférez-vous une boucle `while` ici ?

Pour rendre le jeu intéressant, l'ordinateur doit choisir le nombre secret *au hasard*. La librairie C standard propose des fonctions renvoyant des nombres pseudo-aléatoires [1] déclarées dans `<stdlib.h>`. L'ordinateur va utiliser la fonction : `int rand();` qui renvoie un nombre pseudo-aléatoire entier entre 0 et la constante `RAND_MAX` (égale à 2147483647) inclus. Pour renvoyer un nombre pseudo-aléatoire entre 0 et `NB_MAX`, `NB_MAX` inclus (`NB_MAX` << `RAND_MAX`), il suffit de calculer le reste de la division entière de `rand()` par `(NB_MAX + 1)`, c'est-à-dire le nombre renvoyé par rand() modulo `(NB_MAX + 1)` (opérateur % en C). rand vient de random qui veut dire aléatoire en anglais.

Un exemple de programme illustrant l'utilisation de rand pour engendrer un nombre pseudo-aléatoire est le suivant :

1. http ://fr.wikipedia.org/wiki/Générateur_de_nombres_pseudo-aléatoires

```c
#include <stdlib.h> /* EXIT_SUCCESS, rand, srand */
#include <time.h> /* time */

#define NB_MAX 15 /* nombre secret entre 0 et NB_MAX inclus */

int main()
{
    int nombre_secret; /* nombre secret à deviner */

    /* initialisation du générateur de nombres pseudo-aléatoires */
    srand(time(NULL)); /* à ne faire qu'une fois */

    /* tirage du nombre secret */
    nombre_secret = rand() % (NB_MAX + 1); /* entre 0 et NB_MAX inclus */

    /* manche joueur ... */

    return EXIT_SUCCESS;
}
```