Travaux dirigés 5 : fonctions et procédures (2)

Durant les séances précédentes vous avez réalisé plusieurs programmes en C effectuant chacun une tâche. Le but ici est d'utiliser des fonctions pour commencer à réunir plusieurs de ces programmes en un seul, dans lequel l'utilisateur choisira la tâche à effectuer dans un menu. À la fin de l'exécution d'une tâche, le menu est à nouveau affiché pour laisser le choix à l'utilisateur d'exécuter d'autres tâche ou de quitter le programme. Un exemple d'exécution est donné plus bas.

Comme vous allez écrire un gros programme dans un seul fichier source, il est essentiel que vous structuriez bien votre fichier source, et que celui-ci soit correctement indenté. Si vous n'aimez pas gérer vous même l'indentation apprenez à utiliser l'éditeur de texte `emacs` et son indentation automatique.

Vous travaillerez sur trois parties du programme :
- les déclarations du début du programme (fonctionnalités, constantes, fonctions),
- les définitions de fonctions,
- la fonction principale (`main`).

Vous ferez en sorte de pouvoir **tester votre programme le plus tôt possible et le plus souvent possible**, quitte à afficher à l'utilisateur que certaines choses ne sont pas disponibles / pas terminées.

```
****************** MENU ******************
*                                        *
*   1) Tester si un nombre est premier   *
*   2) Deviner un nombre                 *
*   0) QUITTER                           *
*                                        *
************************* votre choix : 1
Donner un nombre entier positif : 34
Le nombre 34 n'est pas premier, 2 divise 34

****************** MENU ******************
*                                        *
*   1) Tester si un nombre est premier   *
*   2) Deviner un nombre                 *
*   0) QUITTER                           *
*                                        *
************************* votre choix : 2

Votre choix ?
8
```

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

****************** MENU ******************
*                                        *
*   1) Tester si un nombre est premier   *
*   2) Deviner un nombre                 *
*   0) QUITTER                           *
*                                        *
************************* votre choix : 0

Sayonara
```

1. Commencer par faire en sorte que :
   (a) le programme affiche un menu proposant 3 choix représentés par des entiers : (1) tester si un entier est premier, (2) deviner un nombre ou (0) quitter. L'utilisateur fera son choix en entrant un entier.
   (b) Si cet entier est 0, mettre fin au programme.
   (c) Si cet entier est 1 ou 2 afficher « non disponible », puis boucler à l'étape 1.

   Il est recommandé de découper ce programme en fonctions et procédures.
   - Déclarer et définir une procédure `afficher_menu()` qui affichera le menu. Tester là.
   - Déclarer et définir une fonction `choix_utilisateur()`, sans paramètres qui renverra une valeur entière saisie par l'utilisateur. Tester là.
   - Déclarer et définir une fonction `executer_menu()` qui :
     - affichera le menu à l'utilisateur et réalisera la saisie de son choix (avec les fonctions et procédures précédentes) ;

  - lorsque ce choix est 1, appelera une procédure `menu_premier()` (pendant la mise au point du programme, faîtes en sorte que cette procédure affiche "non disponible") ;
  - puis, lorsque ce choix est différent de 0, renverra `TRUE` et lorsque ce choix est égal à zéro renverra `FALSE`.
- Modifier la fonction `main` de telle sorte qu'elle fasse appel à la fonction `executer_menu` tant que celle-ci renvoie `TRUE`.

1 Ajouts de fonctionnalités

Vous pouvez maintenant commencer à programmer quelques unes des tâches que propose le menu. Si vous possédez des programmes réalisant ces tâches, inspirez vous en (copier/coller).

2. Redéfinir la procédure `menu_premier()` de manière à tester si un nombre choisi par l'utilisateur est premier. Elle fera appel à une fonction `est_premier()` (dont vous retrouverez la définition et la déclaration dans le TD 4) et à la fonction `choix_utilisateur()` pour le choix de l'entier.
3. Modifier votre fonction `choix_utilisateur()` de telle sorte que :
   - elle prenne en argument deux paramètres entiers a et b ;
   - si l'utilisateur saisit un nombre $n \in [a, b]$ la fonction retourne n sans générer d'affichage ;
   - si l'utilisateur saisit un nombre $n \notin [a, b]$ l'intervalle de saisie est affiché à l'utilisateur et la saisie redemandée, jusqu'à cinq fois.
   - si au bout de cinq fois l'utilisateur n'a toujours pas donné un nombre dans l'intervalle, la procédure renvoie a.
   - Tester. Quel est le problème lorsque l'utilisateur saisie autre chose que des chiffres au clavier ? Nous corrigerons ça un peu plus loin.
4. Ajouter le nécessaire à votre programme pour pouvoir jouer à deviner un nombre.
5. Ajouter une entrée dans le menu pour la simulation d'une population de lapins.
6. Comme vous l'aviez remarqué, la fonction `choix_utilisateur()` échoue à redemander un nombre si l'utilisateur saisit autre chose que des chiffres. Vous pouvez corriger ça en utilisant le code suivant à la place de `scanf("%d", &choix)`.

```
char saisie[256]; /* declaration d'un chaine qui nous servira de
                   * "tampon" */

fgets(saisie, 256, stdin); /* preleve la ligne tapee par
                              l'utilisateur et la place dans le
                                  tampon */
sscanf(saisie, "%d", &choix); /* tente de reconnaitre un entier
                                 dans le tampon */
```

7. Ajouter au menu une entrée `calculette` et écrire une procédure `calculette()` qui affichera le résultat d'une expression `nombre opération nombre` entrée par l'utilisateur, où les nombres sont des `double` et l'opération un caractère parmi `+, -, *, /`.
   Indication : vous pouvez vous inspirer du code suivant :

```
...
scanf("%lg %c %lg", &x, &op, &y);
if ('+' == op)/* faire une addition */
{
    printf("%lg\n", x + y); /* affichage du résultat */
}
```