TD5: Langage C

Exercice 1 :

Écrire :
- Une fonction, nommée **f1**, se contentant d'afficher "bonjour" (elle ne possèdera aucun argument ni valeur de retour)
- Une fonction, nommée **f2**, qui affiche "bonjour" un nombre de fois égal à la valeur reçue en argument (int) et qui ne renvoie aucune valeur,
- Une fonction, nommée **f3**, qui fait la même chose que **f2**, mais qui, de plus, renvoie la valeur (int) 0.

Écrire un petit programme appelant successivement chacune de ces trois fonctions, après les avoir convenablement déclarées sous forme d'un prototype.

Exercice 2 :

Qu'affiche le programme suivant ?

```
int n=5 ;
main()
{
       void fct (int p) ;
       int n=3 ;
       fct(n) ;
}
void fct(int p)
{
       printf("%d %d", n, p) ;
}
```

Exercice 3 :

Écrire une fonction qui se contente de comptabiliser le nombre de fois où elle a été appelée en affichant seulement un message de temps en temps, à savoir :
- au premier appel : *** appel 1 fois ***
- au dixième appel : *** appel 10 fois ***
- au centième appel : *** appel 100 fois ***
- et ainsi de suite pour le millième, le dix millième appel...
- On supposera que le nombre maximal d'appels ne peut dépasser la capacité d'un long

Exercice 4 :

Écrire une fonction récursive calculant la valeur de la « **fonction d'Ackermann** » **A** définie pour m>0 et n>0 par :

A(m,n) = A(m-1,A(m,n-1)) pour m>0 et n>0
A(0,n) = n+1
A(m,0) = A(m-1,1) pour m>0

Exercice 5 :

Quels résultats fournira ce programme :

```
#include <stdio.h>
int n=10, q=2 ;
main()
{
       int fct (int) ;
       void f (void) ;
       int n=0, p=5 ;
       n = fct(p) ;
       printf ("A : dans main, n = %d, p = %d, q = %d\n", n, p, q) ;
       f() ;
}

int fct (int p)
{
```

```
        int q ;
        q = 2 * p + n ;
        printf ("B : dans fct, n = %d, p = %d, q = %d\n", n, p, q) ;
        return q ;
}
void f (void)
{
        int p = q * n ;
        printf ("C : dans f, n = %d, p = %d, q = %d\n", n, p, q) ;
}
```

Exercice 6 :

Ecrire une fonction qui reçoit en arguments 2 nombres flottants et un caractère et qui fournit un résultat correspondant à l'une des 4 opérations appliquées à ses deux premiers arguments, en fonction de la valeur du dernier, à savoir : addition pour le caractère **+**, soustraction pour **-**, multiplication pour ***** et division pour **/** (tout autre caractère que l'un des 4 cités sera interprété comme une addition). On ne tiendra pas compte des risques de division par zéro.
Ecrire un petit programme (main) utilisant cette fonction pour effectuer les 4 opérations sur deux nombres fournis en donnée.

Exercice 7 :

Transformer le programme (fonction + main) écrit dans l'exercice précédent de manière à ce que la fonction ne dispose plus que de 2 arguments, le caractère indiquant la nature de l'opération à effectuer étant précisé, cette fois, à l'aide d'une variable globale.

Exercice 8 :

Ecrire 2 fonctions à un argument entier et une valeur de retour entière permettant de préciser si l'argument reçu est multiple de 2 (pour la première fonction) ou multiple de 3 (pour la seconde fonction).
Utiliser ces deux fonctions dans un petit programme qui lit un nombre entier et qui précise s'il est pair, multiple de 3 et/ou multiple de 6, comme dans cet exemple (il y a deux exécutions) :

```
donnez un entier : 9
il est multiple de 3

donnez un entier : 12
il est pair
il est multiple de 3
il est divisible par 6
```