Informatique S1 – Programmation C
Exercices

TD 4 : Les boucles *while* et *do while*

Dans ce TD, nous allons réaliser des exercices couvrant l'usage des boucles *while* et *do while* dans le langage C.

Exercice 1

a) Exécuter le programme ci-dessous. Que fait-il ? Combien de fois l'expression « `count = count + 1 ;` » (ligne 16) sera exécutée (minimum / maximum) ?

b) Réécrire le programme ci-dessous en utilisant la boucle ***while*** à la place de la boucle ***do while***.

```
1
2   #include <stdio.h>
3
4   /*  TD 4 : boucle Do-While    */
5
6   int main () {
7       int n = 0;
8       int count = 0;
9
10      do {
11          printf ("Donner un entier > 0 : ");
12          scanf ("%i", &n);
13
14          printf ("Vous avez fourni %i \n", n);
15
16          count = count + 1;
17      } while (n > 0);
18
19      printf ("Vous avez entre %d n. entiers positifs \n", count);
20  }
21
```

Exercice 2

Comparer les deux programmes (code A et code B) ci-dessous lorsque l'utilisateur fourni le chiffre « 1 » à la première demande et lorsque l'utilisateur fourni le chiffre « 0 » à la première demande. Expliquer.

Code A :

```c
#include <stdio.h>

/*  TD 4 : Calculer la somme des numeros entres par l'utilisateur
          version avec while
*/

int main ()
{
    int n = 0;
    int count = 0;
    int sum = 0;

    /* lecture du premier nombre */
    printf ("Entrer un entier (0 pour terminer) : ");
    scanf ("%d",&n);

    while (n != 0)  {
        sum = sum + n;
        count = count + 1;

        /* lecture d'un nouveau nombre */
        printf ("Entrer un entier (0 pour terminer) : ");
        scanf ("%d",&n);
    }

    printf ("La somme des %d entiers fournis est : %d \n", count, sum);

}
```

Code B :

```c
#include <stdio.h>

/*  TD 4 : Calculer la somme des numeros entres par l'utilisateur
          version avec do-while
*/

int main ()
{
    int n = 0;
    int count = 0;
    int sum = 0;

    /* lecture du premier nombre */
    printf ("Entrer un entier (0 pour terminer) : ");
    scanf ("%d",&n);

    do  {
        sum = sum + n;
        count = count + 1;

        /* lecture d'un nouveau nombre */
        printf ("Entrer un entier (0 pour terminer) : ");
        scanf ("%d",&n);
    } while (n != 0);

    printf ("La somme des %d entiers fournis est : %d \n", count, sum);

}
```

Exercice 3

a) Remplir les trous dans le programme ci-dessous, sachant qu'il doit calculer x^y, avec x et y sont fournis par l'utilisateur.

```
#include <stdio.h>

int main () {
   float x, y; /* valeur fournis par l'utilisateur */
   float __;    /* z = x puissance y */
   int i;

   /* lecture des variables */
   ______ ("Entrez x : ");
   scanf ("__", &x);
   printf ("Entrez y : ");
   scanf ("%f", __);

   z = 1;
   i = 1;

   /* on multiplie x y-fois */
   while (_______) {
        z = z * x;
        i = _____;           /* on augmente le compteur */
   }

   /* presentation des resultats */
   printf ("x ^ y = %f \n", ___);
}
```

b) *Question « défi »* : Pouvez-vous proposer un second programme qui fait la même chose sans utiliser une boucle ? Comment ?

Exercice 4

a) Sachant que le factoriel d'un numéro entier *n* est égale à *1 * 2 * … * n-1 * n*, remplir les trous du programme ci-dessous, lequel calcule le factoriel de *n*, avec *n* fourni par l'utilisateur.

```
#include <stdio.h>

/*  TD 4 : Factoriel de n
 *  Factoriel de n = 1 * 2 * ... * n-1 * n
 */

int main () {
  int n;
  int i ___;
  int fact ____;  /* factoriel de n */

  printf ("Entrez n : ");
  ______ ("%d", &n);

   do {
       fact = fact * ____;
       _________;
  } while ( i <= n );

  printf ("Factoriel de %d est %d \n", n, fact);
}
```

b) Réécrire le programme ci-dessus pour qu'il utilise l'instruction « `while` » à la place de « `do … while` ».

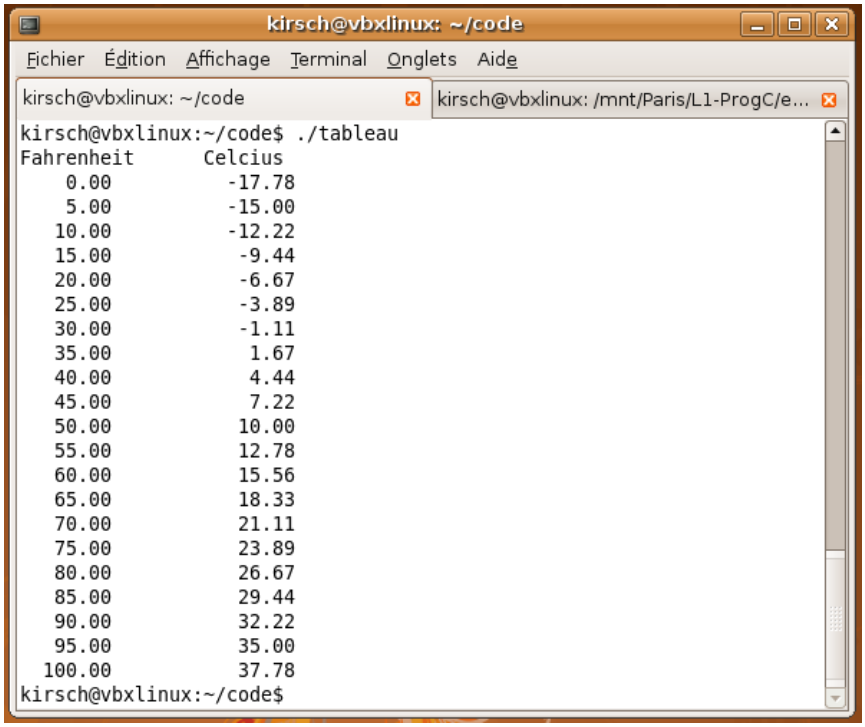
Exercice 5

Sachant que la moyenne d'un ensemble de *n* numéros réels est définie comme la somme de ces numéros divisée par *n* ($moyenne = {\sum_{i=1}^{n} x_i}/{n}$), calculer la moyenne entre 10 numéros réels fournies par l'utilisateur. (Astuce : partir de l'exercice 2).

Exercice 6

Ecrire un programme qui affiche un tableau pour convertir les degrés Fahrenheit en degrés Celsius. Par exemple, votre programme affichera sur la colonne de gauche les nombre de 0 à 100 (de 5 en 5) correspondant aux degrés Fahrenheit, et sur la colonne de droite les valeurs correspondantes en Celcius (voir la figure ci-dessous).

Rappel : $c = 5 * {(f - 32)}/{9}$



Exercice 7

a) Observer le code ci-dessous. Est-il correct ? Que fait-il ? Comment faire pour qu'il soit compréhensible ?

```
#include <stdio.h>
#include <math.h>
int main () {int i; float
x, xpow; float racx;
printf ("Donnez un nb : ");
scanf ("%f",&x); i=0; xpow=1;
while (i<x) {xpow=xpow*x; i=i+1;}
racx=sqrt(x); printf ("%f ^ %f = %f \n sqrt = %f\n",
x, x, xpow, racx);
return (0);
}
```

b) Tracer le programme ci-dessous (en considérant que l'utilisateur a fourni la valeur 4). Est-il équivalent au programme ci-dessus (question 7a) ?

```c
#include <stdio.h>
#include <math.h>

int main ()
{
    int i;
    float x, xpow, racx;

    /* point d'observation 1 */
    printf ("Donnez un nb : ");
    scanf ("%f",&x);

    i=0;
    xpow=1;

    /* point d'observation 2 */
    while (i<x)
    {
        xpow=xpow*x;
        i=i+1;
        /* point d'observation 3 */
    }

    racx=sqrt(x);
    printf ("%f ^ %f = %f \n sqrt = %f\n",
            x, x, xpow, racx);

    /* point d'observation 5 */

    return (0);
}
```

NOTE : Afin de compiler sous Linux, les programmes qui utilisent la bibliothèque « *math.h* » (`#include <math.h>`), il faut utiliser l'option « -lm » du compilateur *gcc* : `gcc -lm -o programme programme.c` .

Exercice 8

Tracer le programme ci-dessous. Qu'affichera-t-il lorsque l'utilisateur lui fournit la séquence de numéros 4, 1 et 0 ?

```c
#include <stdio.h>
#include <math.h>
/*  TD 4 : Calculer les racines carres  */
int main () {
   float x, racine;

   /* point d'observation 1 */
   do {
     printf ("Entrer un nb (0 pour terminer) : ");
     scanf ("%f", &x);

     /*point d'observation 2 */
     racine = sqrt (x);

     printf ("La racine carre de %f est %f \n", x, racine);

   } while (x != 0);

   /*point d'observation 3 */
}
```

Exercice 9

Trouver l'erreur dans le code ci-dessous, sachant qu'il sert à calculer la *suite de Fibonacci* (définie ci-dessous) tant que l'utilisateur accepte de continuer en répondant avec un « 0 » (zéro) à la question posée :

Série de Fibonacci : $u_1 = 1$

$u_2 = 1$

$u_n = u_{n-1} + u_{n-2}$ pour $n \geq 2$

```
#include <stdio.h>

/*  TD 4 : Suite de Fibonacci
 *  La suite de Fibonacci est un probleme mathematique defini comme suit:
 *    u0 = u1 = 1
 *    un+2 = un + un+1, pour tout n >= 0
 */

int main () {
 int u0, u1;
 int un;
 int op;
 int i;


 u0 = 1;    /* u0 = 1 */
 u1 = 1;   /* u1 = 1 */
 i = 2;

 printf ("Serie de Fibonacci \n u0 = 1 \n u1 = 1 \n");

 do {
   /* un = un-1 + un-2 */
   un = u0 + u1;

   printf (" u%d = %d\n", i, un);

   u0 = u1;    //on garde les 2 dernier valeurs
   u1 = un;

   printf ("Voulez-vous continuer ( 0 = oui, 1 = non) : ");
   scanf ("%d", &op);
 } while (op = 0);

}
```

Exercice 10 (Avancé) : Approximation de Pi

Dans la mathématique, des nombreuses suites ou séries convergent vers π ou vers un multiple de π. Ces séries sont souvent à l'origine de calculs de valeurs approchées de ce nombre. Parmi ceux-là se trouve la *Fonction zêta de Riemann $\zeta(2)$*, laquelle peut être définie comme suit :

$$\zeta(2) = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \cdots + \frac{1}{k^2} + \cdots = \frac{\pi^2}{6}$$

Réaliser un programme qui calcule cette série jusqu'un numéro *k* fourni par l'utilisateur et qui compare le résultat obtenu à la valeur de $\pi^2 / 6$.

Vous pouvez utiliser comme base le programme ci-dessous, qui affiche la valeur de π et de π^2:

```c
#include <stdio.h>
#include <math.h>

int main () {

 double pi_carre = 0.0;
 double pi = M_PI;    /* valeur de PI selon la bibliotheque math.h */

 /* pi au carre */
 pi_carre = pow (pi, 2);

 printf ("PI = %lf \t PI^2 = %lf \n", pi, pi_carre);

 return (0);
}

```