TP3 – La structure de contrôle *tant que*

Exercice 1

Soit le programme :

```
/**
 *@brief Que fait ce programme ? là est la question …
*/
#include <stdio.h>
#include <stdlib.h>

int main (){
        int i, n , somme;

        printf(" entrez un entier naturel : \n") ;
        scanf("%d",&n) ;
        i=1;
        somme=0 ;
        while ( i<= n ) {
                somme=somme+i ;
                i=i+1;
        }
        printf("résultat : %d\n",somme) ;
        return EXIT_SUCCESS ;
}
```

Questions :

1) Testez ce programme avec les cas suivants :
   0 1 2 5
2) En une phrase, que fait ce programme ?

Exercice 2

En vous inspirant de l'exemple précédent, écrivez et testez un programme qui calcule et affiche à l'écran la factorielle d'un entier positif ou nul entré au clavier.
Rappel : n! = 1*2*3* …*(n-1)*n = (n-1)! * n. 0! =1
N.B. attention, en exécutant ce programme, à ce que le résultat n'aille pas au-delà de la capacité du type *int*, donc ne saisissez pas un *n* trop grand !

Exercice 3

Écrivez un programme qui lit au clavier le nombre de valeurs d'une suite de valeurs entières, puis les valeurs de la suite, et qui affiche la moyenne de ces valeurs.
N.B. attention à la division par zéro !

Exercice 4

Étudions maintenant le cas où nous ne connaissons pas au départ de la boucle le nombre d'itérations à effectuer, mais nous connaissons la condition d'arrêt :

Écrivez un programme qui lit au clavier une suite de valeurs entières terminée par -1 (-1 est le marqueur de fin, il ne fait pas partie des valeurs dont on veut calculer la moyenne), et qui affiche la moyenne de ces valeurs.

N.B. La suite peut être vide (cas où l'utilisateur du programme tape -1 dès le départ), donc là aussi, attention à la division par 0.

Exercice 5

Écrivez un programme qui lit au clavier le nombre de valeurs d'une suite de valeurs entières, puis les valeurs de la suite, et qui affiche la plus grande de ces valeurs (Pensez au raisonnement appliqué dans la question 2 de l'exercice 3 du TD 2).

Exercice 6

Le salaire net d'un employé se calcule en retirant *taux*% de charges de son salaire brut (par exemple taux =20).
Écrivez un programme qui calcule et affiche le salaire net en fonction du salaire brut pour autant de personnes qu'on le souhaite, puis le salaire net moyen.

L'utilisateur saisit pour chaque personne le nom et le salaire brut (nom puis entrée, salaire brut puis entrée), et tape * à la place du nom pour indiquer la fin des données.

N.B. On considère qu'un nom saisi au clavier ne comporte pas d'espaces et est limité à 19 caractères.

On envisage le cas de la liste vide (cas où l'utilisateur tape '*' dès le départ), donc attention à la division par zéro !

Exercice 7

Écrivez un programme qui fait l'affichage de trois choix :

(1) Faire action 1
(2) Faire action 2
(0) Quitter

Ce programme lit ensuite le choix de l'utilisateur et affiche le message *ad hoc*
(« action1 », « action2 », « au revoir », « Erreur : vous devez saisir 1, 2, ou 0 »).
Cette action est répétée jusqu'à ce que l'utilisateur ait tapé 0.

Exercice 8

Écrivez un programme qui lit un entier naturel au clavier et qui affiche à l'écran un message indiquant si ce nombre a d'autres diviseurs que 1 et lui-même, ou non.

Si le nombre n'a pas d'autres diviseurs que 1 et lui-même, on dit qu'il est un nombre premier.

Attention ! ici, la condition d'arrêt est double :

on arrête la boucle soit parce qu'on a trouvé un diviseur, auquel cas le nombre n'est pas premier, soit parce qu'on a essayé tous les « candidats diviseurs » sans qu'aucun soit diviseur, auquel cas le nombre est premier.

Exercices complémentaires :

Exercice 9

Dans un langage de programmation, on ne dispose pas de l'opération puissance (X^n) avec X un réel et n un entier naturel.
Nous avons écrit en TD (exercice 6 du TD 3) un algorithme qui calcule et affiche la puissance (entière) d'un nombre réel, *X* et *n* étant lus au clavier.

Question : Traduisez en C cet algorithme, et tester-le.

Exercice 10

Ecrivez un programme qui
- lit au clavier le nombre d'étudiants d'un groupe
- pour chaque étudiant, lit son nom et les deux notes qu'il a obtenues respectivement en DS et TP, et affiche son nom et sa moyenne (2/3 DS, 1/3 TP)

N.B. On considère qu'un nom saisi au clavier ne comporte pas d'espaces et est limité à 19 caractères. Ne pas oublier que les valeurs numériques citées dans l'énoncé ne sont que des exemples (donc vous devez utiliser la notion de constante).

Exercice 11

Ecrivez un programme qui compte le nombre d'occurrences de la lettre 'a' dans une suite de caractères entrés au clavier, terminée par un point.

N.B. Pour cet exercice, il faut lire caractère par caractère :

```
char caractereCourant ;
…….
scanf("%c",&caractereCourant) ;
while ( ……) {
        ……
        scanf("%c",&caractereCourant) ;
}
…….
```

Ainsi, l'utilisateur n'aura pas à faire « entrée » après chaque caractère, mais seulement après avoir tapé le point final.

Exercice 12

Ecrivez un programme qui compte le nombre d'occurrences de 'le' dans une suite de caractères entrés au clavier, terminée par un point.

Exemple :
L'utilisateur saisit au clavier : le soleil brille.
Le programme doit afficher : « il y a 3 fois 'le' »

N.B. Même remarque que pour l'exercice 11 : il faut lire caractère par caractère.

Exercice 13

Ecrivez un programme qui effectue l'approximation d'une racine carrée par la méthode d'Héron d'Alexandrie.
Cet algorithme repose sur le principe suivant :
La suite U telle que $U_0 = 1$ et $U_n = \frac{1}{2}(U_{n-1} + a/U_{n-1})$ tend vers racine carrée de a.

On arrêtera la boucle quand la différence en valeur absolue entre le carré de l'approximation et *a* sera inférieure à un *Epsilon* fixé au départ.
La fonction fabs (qui délivre un double) demande la directive #include <math.h>

Exercice 14

Dans cet exercice, le programme ne « tire » pas le dé lui-même ; il se contente de cumuler les points en fonction des données (valeur du dé) que le joueur communique au programme.

Deux joueurs A et B lancent le dé **alternativement** ; le premier qui atteint 50 points a gagné (**même si** l'autre joueur a joué un coup de moins).

Ecrivez un programme qui demande alternativement à chaque joueur de jouer, puis la valeur du dé que ce joueur a tiré, et affiche en fin de partie le joueur gagnant et son score.

Exercice 15

Ecrivez un programme qui calcule une approximation de e^x, sachant que le développement en série entière de *e puissance x* est :
$1 + x/1! + x^2/2! + \ldots + x^n/n! + \ldots$

N.B. On pourra comparer le résultat obtenu avec la valeur que fournit la fonction exp (prévoir d'inclure math.h, et compiler avec l'option -lm).

Exercice 16

Ecrivez un programme qui calcule une approximation de cos x, sachant que le développement en série entière de *cosinus x* est :
$1 - x^2/2! + x^4/4! + \ldots + (-1)^r * x^{2*r} / (2*r)! + \ldots$

N.B. Même remarque que précédemment.