Université Paul Sabatier TOULOUSE III

S3

Programmation impérative en langage C

feuille de TP n°2 : Boucles

Objectifs du TP :

- *Savoir écrire une boucle en choisissant parmi les 3 syntaxes possibles (for, do...while et while) celle qui est la plus adaptée.*
- *Mettre en oeuvre pour chaque programme un début de spécification qui précise sous forme de cartouche :*

Nom du programme et sa description *Données typées* *Résultats typés*

◇ Exercice **1** : n étant un entier lu au clavier, écrire un programme qui saisit n valeurs réelles positives et qui détermine :

– l'étendue des valeurs (c'est à dire la différence entre la plus grande et la plus petite valeur)
– le nombre de valeurs comprises dans l'intervalle $I = [2, 5]$
– la somme des racines carrées des n valeurs

Le cartouche sera :

Exo1 : Détermine l'étendue(valeur_max-valeur_ min, le nombre de valeurs dans I=[2,5], la somme des racines carrées de n réels *Données : int n, float val* $(x_1, \cdots, x_n)$ *Résultats : float etendue, int nb_val_I, float som_rac*

Exemple d'exécution :

```
Quel est le nombre de valeurs ? 5
Taper la 1 eme valeur : 10.2
Taper la 2 eme valeur : 5.3
Taper la 3 eme valeur : 5.0
Taper la 4 eme valeur : 1
Taper la 5 eme valeur : 2
L'étendue des valeurs est 9.2, le nombre de valeurs dans I est 2, la somme
des racines carrées est 10.146
```

◇ Exercice **2** : Ecrire un programme qui, successivement :

– affiche les 26 mots : 'aa' $\cdots$ 'zz'
– affiche l'alphabet en décalé :

Exemple d'exécution :

```
a
   b
      c
         ...
                z
```

– affiche tous les mots de 'aa' à 'az' puis 'bb' à 'bz' etc.
Exemple d'exécution :

```
aa  ab  ac  ···  ax  ay  az
ba  bb  bc  ···  bx  by  bz
...
za  zb  zc  ···  zx  zy  zz
```

– affiche les tables de multiplication de 1x1 jusqu'à 9x9
Exemple d'exécution :

```
1   2   3   4   5   6   7   8   9   10
2   4   6   8   10  12  14  16  18  20
...
9   18  27  36  45  54  63  72  81  90
```

◇ Exercice **3** : Analyse d'un programme :
On considère le programme :

```
# include < stdio.h>
 int main ()
{
int  x,s,i;
do
    {
    printf("Taper un nombre :");
    scanf("%d",&x);
    }
while (x<0);

s=0;
for (i=1;i<=x/2;i=i+1)
    {
    if  (x % i ==0)
        {
        s=s+i;
        }
    }
printf("s=%d",s);
return (0);
}
```

– Exécuter ce programme "à la main" pour les valeurs : $x = 6$ $x = 17$ $x = 28$ $x = 0$

– A quoi servent les 2 boucles ? Que calcule ce programme ?

◇ Exercice **4** : Un nombre entier positif est dit parfait s'il est égal à la somme de ses diviseurs sauf lui-même.
par exemple : $6(= 1 + 2 + 3)$ et $28(= 1 + 2 + 4 + 7 + 14)$ sont parfaits.
Ecrire un programme qui lit au clavier deux nombres entiers positifs, les ordonne si cela est nécessaire et affiche les nombres parfaits compris entre les deux bornes.

Précisément, on souhaite obtenir l'affichage suivant :

```
Taper un nombre : 500
Taper un nombre : 10
Les nombres parfaits entre 10 et 500 sont : 28 et 496.
```

```
Taper un nombre : 30
Taper un nombre : 50
Il n'y a pas de nombre parfait entre 30 et 50.
```

◇ Exercice **5** : La suite de Syracuse :
La suite définie par :

$$u_{n+1} = \begin{cases} u_n/2 & \text{si } u_n \text{ est pair} \\ 3u_n + 1 & \text{sinon} \end{cases}$$

est appelée suite de Syracuse (du nom de l'université new-yorkaise).
Une conjecture, énoncée en 1950, affirme que quelle que soit le premier terme $u_1 \in \mathbb{N}^*$, cette suite se termine en bouclant sur la séquence : 4 2 1.
Ecrire un programme qui, pour un entier donné u1 par l'utilisateur, affiche les termes puis la longueur de la suite de Syracuse de premier terme u1.(On s'arrête au premier 1 rencontré)
Exemple d'exécution :

```
Taper u1 : 5
La suite de Syracuse est : 5 16 8 4 2 1
Elle a pour longueur : 6
```

Modifier le programme précédent pour déterminer le premier terme d'une suite de syracuse de 354 termes.

◇ Exercice **6** : Ecrire un algorithme qui détermine et affiche le premier multiple de 10 d'une liste de N nombres entiers compris entre 1 et 100 générés aléatoirement.

Attention, la liste peut ne pas contenir de multiples de 10 !
Exemple d'exécution :

```
Taper un nombre N : 5
Entiers aléatoires : 23 11 50
Le premier multiple de 10 est le 3 eme nombre genere : 50
```

```
Taper un nombre N : 6
Entiers aléatoires : 1 29 19 53 65 77
Il n'y a pas de multiple de 10 dans les 6 nombres generes.
```

Rappel : Comment générer des nombres aléatoires ?

- rand() renvoie un entier pseudo-aléatoire compris entre 0 et RAND_MAX
- rand()%100 +1 renvoie un entier compris entre 1 et 100
- pour ne pas obtenir toujours les mêmes valeurs, placer au début de votre programme la clause : **# include <time.h>** puis au début du main, l'instruction : **srand(time(NULL)) ;**