TD Informatique 06 : Boucles

Consignes générales : N'oubliez pas pour ce TD comme pour les suivants de vous creer un répertoire consacré au TD et d'enregister vos codes desssus.

On rappelle que les commandes à taper dans le terminal pour compiler puis exécuter votre programme C :
— Pour compiler : *gcc -o nom_executable nom_programme.c*
— Pour exécuter : *./nom_executable*
Il est conseillé de toujours écrire l'algorithme en pseudo-code avant de passer sur machine.

Exercice 1

Que va afficher l'algorithme suivant ? ?

```
PROGRAMME test boucle
VARIABLES
   a, b : entier
   c1 : booléen
DEBUT
   a ← 100
   b ← 5
   TANT QUE (a > 0) FAIRE
      a ← a DIV 5
      b ← b * 2
      ÉcriRE ("b="+b)
   FIN TANT QUE
   ÉcriRE ("fin du programme")
FIN
```

Exercice 2

Soit l'algorithme suivant :

```
PROGRAMME test boucle POUR
VARIABLES
   i : réel
DEBUT
   POUR ( ? ? ?) FAIRE
      ÉcriRE (i)
   FIN POUR
FIN
```

Par quoi faut-il remplacer **???** pour afficher les listes suivantes
— 1 2 4 5
— 10 20 30 40
— 8 4 0 -4 -8
— -1 -0.5 0 0.5

Exercice 3

Quelle boucle est la plus adaptée dans les situations suivantes ?
— Demander à l'utilisateur une valeur entre 0 et 100.
— Demander à l'utilisateur de saisir 10 nombres quelconques.
— Demander à l'utilisateur de saisir 10 nombres prositifs.
— Calculer la factorielle d'un nombre.

Exercice 4

1. Que fait l'algorithme suivant ?

```
PROGRAMME algorithme très mysterieux
VARIABLES
   i, m, s : entier
DEBUT
```

```
s ← 0
POUR i DE 0 A 10 FAIRE
   ÉcriRE ("Entrez un nombre.")
   LIRE(m)
   s ← s+m
FIN POUR
ÉcriRE("s="+s) FIN
```

2. Écrire un autre algorithme qui réalise la même chose avec une boucle TANT QUE.

Exercice 5
Affichez sur votre terminal, à l'aide d'un programme en C tous les entiers de 1 à 100 compris.

Exercice 6 (*Table de multiplication*)
Saisir un nombre entier N et afficher la table de multiplication de N (ses multiples entre 0 et 10).

Exercice 7
Écrire un programme qui affiche le plus petit entier n tel que (n+1)*(n+3) dépasse 12345.

Exercice 8 (*diviseur de 3*)

1. Demander à l'utilisateur de saisir un nombre entier positif N.
2. Affichez tous les nombres divisibles par 3 entre 0 et N en utilisant une boucle POUR
3. Idem avec une boucle TANT QUE

Exercice 9 (*Calcul de puissance*)

— Demander à l'utilisateur de saisir deux nombres entiers a et b strictement positifs (assurez vous qu'ils le soient bien avant de passer à la suite !)
— Afficher le résultat de a^b.

Exercice 10 (*Compte à rebours*)

1. Saisir un nombre de secondes et le convertir en heure :min :s. Assurez-vous que les valeurs saisies soit correctes !
2. Décrémenter les secondes en affichant à chaque changement l'heure sous format h :min :s.
   S'arrêter lorsque l'on atteint 00 : 00 : 00 et afficher «BOOM».
3. Est-ce qu'il se déroule une seconde entre chaque affichage ? Pourquoi ?
4. La fonction *sleep()* permet à l'ordinateur de faire "une pause" avant de passer à l'instruction suivante, elle prend en argument le nombre de secondes que l'on veut attendre. Cette fonction appartient à la librairie unistd.h qu'il vous faut donc inclure si vous voulez pouvoir l'utiliser. Modifier votre code pour qu'il se déroule bien une seconde entre deux décrementations du compte à rebours.

Exercice 11 (*calcul de factorielle*) On rappelle que factoriel N (noté N !) se calcule de la manière suivante :N !=2*3*4*5*..*N. Saisir un nombre et afficher sa factorielle.

Exercice 12 (*pliage*) Une feuille en papier fait 0.1 mm d'épaisseur. Écrire un algorithme pour déterminer combien de plis sont nécessaires pour obtenir une épaisseur de 2m ?

Exercice 13 (*Des dessins*)

1. Saisir un nombre entier N strictement positif.
2. Dessinez sur le terminal une ligne de N étoiles.
3. Dessinez sur le terminal un carré plein d'étoiles de coté N.

Exercice 14 (*Table de multiplication pt 2*)
Affichez la table de multiplication des 10 premiers entiers.

Exercice 15 (*Des dessins pt2*)

1. Saisir un nombre entier N strictement positif.
2. Dessiner sur le terminal un triangle rectangle de hauteur N.
   Par exemple, si $h = 5$, on affichera le dessin suivant :
   *
   **
   ***
   ****
   *****

Exercice 16 (*Les triplets*)

1. Écrire un algorithme permettant l'affichage de tous les triplets (a, b, c) tels que a, b, c $\in [1; n]$. Par exemple, pour n=3, le programme doit afficher : (1,1,1) (1,1,2) (1,1,3) (1,2,1) (1,2,2) (1,2,3) (1,3,1) (1,3,2) (1,3,3) (2,1,1) ... (3,3,3)
2. Modifiez le programme précédent pour qu'il génère désormais la liste des triplets (a, b, c) avec c >b > a. Par exemple, pour n=3, le résultat doit être : (1,2,3).
3. Modifier le programme pour afficher, en plus des triplets, le nombre de triplets obtenus.

Exercice 17 (*Nombre premier ?*)

Un nombre est premier s'il n'est divisible que par 1 et par lui-même.
Écrire un programme qui determine si un nombre saisi par l'utilisateur est premier.