TD n° 1
(Correction)

0-M02 : Variables, tests, structures de contrôle

Exercice 1 [Échange des valeurs de deux variables]

On considère que l'on a deux variables $var1$ et $var2$ de type *Entier*. Donner un algorithme qui échange les valeurs de ces variables.

Correction :
Il faut utiliser une variable temporaire *tmp* pour ne pas perdre la valeur d'une des deux variables quand on fait l'échange.

```
1 var1 : Entier
2 var2 : Entier
3 tmp : Entier
4 tmp ⟵ var1
5 var1 ⟵ var2
6 var2 ⟵ tmp
```

Exercice 2 [Opérations sur des variables]

Considérons l'algorithme suivant :

```
1 var1 : Entier
2 var2 : Entier
3 var1 ⟵ 0
4 var2 ⟵ 3
5 var1 ⟵ var1 + var2
6 var2 ⟵ var2 + 1
7 var1 ⟵ var1 + var2
```

Quelles sont les valeurs de $var1$ et $var2$ à la fin de l'exécution de cet algorithme ?

Correction :
$var1$ vaut 7 et $var2$ vaut 4.

```
1 var1 : Entier
2 var2 : Entier
3 var1 ⟵ 0
4 var2 ⟵ 3
5 var1 ⟵ var1 + var2 /* var1 vaut 3 */
6 var2 ⟵ var2 + 1 /* var2 vaut 4 */
7 var1 ⟵ var1 + var2 /* var1 vaut 7 */
```

Exercice 3 [Reste d'une division]

Donner un algorithme qui retourne le reste d'une division d'un entier par un autre, sans utiliser l'opération mathématique modulo.

Correction :

```
1  /* Données d'entrée */
2  dividende : Entier
3  diviseur : Entier
4  /* Donnée de sortie */
5  reste : Entier
6  /* Variable interne à l'algorithme */
7  quotient : Entier
8
9  /* quotient est un entier donc on met la partie
   entiere de la division dedans */
10 quotient ⟵ dividende/diviseur
11 /* le reste est la difference entre le dividende et
   la partie entiere de la division multipliée par le
   diviseur */
12 reste ⟵ dividende − quotient ∗ diviseur
13 retourner reste
```

Exercice 4 [Maximum de deux variables]

Donner un algorithme qui retourne le maximum de deux variables $var1$ et $var2$.

Correction :

```
1 var1 : Entier
2 var2 : Entier
3 si  var1 > var2 alors
4 |   retourner var1
5 sinon
6 |   retourner var2
7 finsi
```

Exercice 5 [Racines d'un polynôme]

Donner un algorithme qui calcule les racines réelles d'un polynôme $ax^2 + bx + c$, en prenant a, b et c trois variables d'entrée entières.

Correction :

Il faut se souvenir de la méthode de calcul des racines réelles d'un polynôme du second degré...

On commence par calculer le discriminant : $\Delta = b^2 - 4ac$. Si le discriminant est strictement positif, le polynôme admet deux racines réelles : $x_1 = \frac{-b-\sqrt{\Delta}}{2a}$ et $x_1 = \frac{-b+\sqrt{\Delta}}{2a}$. Si $\Delta = 0$ alors le polynôme admet une racine double $x_0 = -\frac{b}{2a}$. Si le discriminant est strictement négatif, alors le polynôme n'admet pas de racine réelle.

Attention, ils n'ont pas encore vu les fonctions : on utilise donc $\sqrt{discriminant}$ pour éviter l'appel à une fonction sqrt().

```
1  /* Données d'entrée */
2  a : Entier
3  b : Entier
4  c : Entier
5  /* Données de sortie */
6  x1 : Reel
7  x2 : Reel
8  nbDeSolutions : Entier
9  /* Variable interne à l'algorithme */
10 discriminant : Reel
11
12 /* Calcul du discriminant */
13 discriminant ⟵ b × b − 4 × a × c
14 /* Détermination du nombre de solutions */
15 si discriminant < 0 alors
16 |  nbDeSolutions ⟵ 0
17 sinon
18 |  si discriminant == 0 alors
19 |  |  nbDeSolutions ⟵ 1
20 |  |  x1 ⟵ −b/(2 × a)
21 |  sinon
22 |  |  nbDeSolutions ⟵ 2
23 |  |  x1 ⟵ (−b + √discriminant)/(2 × a)
24 |  |  x2 ⟵ (−b − √discriminant)/(2 × a)
25 |  finsi
26 finsi
```

Exercice 6 [Tri de trois entiers]

Rédiger un algorithme qui affiche trois entiers du plus petit au plus grand.

Correction :

```
1  /* Données d'entrée */
2  entier1 : Entier
3  entier2 : Entier
4  entier3 : Entier
5  /* Données de sortie */
6  tri1 : Entier
7  tri2 : Entier
8  tri3 : Entier
9
10 si entier1 > entier2 alors
11 |  si entier1 > entier3 alors
12 |  |  tri1 ⟵ entier1 /* entier1 est le plus grand */
13 |  |  si entier2 > entier3 alors
14 |  |  |  tri2 ⟵ entier2
15 |  |  |  tri3 ⟵ entier3
16 |  |  sinon
17 |  |  |  tri2 ⟵ entier3
18 |  |  |  tri3 ⟵ entier2
19 |  |  finsi
20 |  sinon
21 |  |  /* On a entier1 > entier2 et entier3 > entier1 donc entier3 est le plus grand */
22 |  |  tri1 ⟵ entier3
23 |  |  si entier1 > entier1 alors
24 |  |  |  tri2 ⟵ entier1
25 |  |  |  tri3 ⟵ entier2
26 |  |  sinon
27 |  |  |  tri2 ⟵ entier2
28 |  |  |  tri3 ⟵ entier1
29 |  |  finsi
30 |  finsi
31 sinon
32 |  si entier2 > entier3 alors
33 |  |  /* On a entier2 > entier1 et entier2 > entier3 donc entier2 est le plus grand */
34 |  |  tri1 ⟵ entier2
35 |  |  si entier1 > entier3 alors
36 |  |  |  tri2 ⟵ entier1
37 |  |  |  tri3 ⟵ entier3
38 |  |  sinon
39 |  |  |  tri2 ⟵ entier3
40 |  |  |  tri3 ⟵ entier1
41 |  |  finsi
42 |  sinon
43 |  |  /* On a entier2 > entier1 et entier3 > entier2 donc entier3 est le plus grand */
44 |  |  tri1 ⟵ entier3
45 |  |  si entier1 > entier2 alors
46 |  |  |  tri2 ⟵ entier1
47 |  |  |  tri3 ⟵ entier2
48 |  |  sinon
49 |  |  |  tri2 ⟵ entier2
50 |  |  |  tri3 ⟵ entier1
51 |  |  finsi
52 |  finsi
53 finsi
54 afficher( tri1" > "tri2" > "tri3 )
```

Exercice 7 [Le temps qui passe]

Écrire un algorithme qui ajoute une seconde à l'heure actuelle. Indication : il n'y a que 60 secondes par minute, il n'y a que 60 minutes par heure, il n'y a que 24 heures dans une journée.

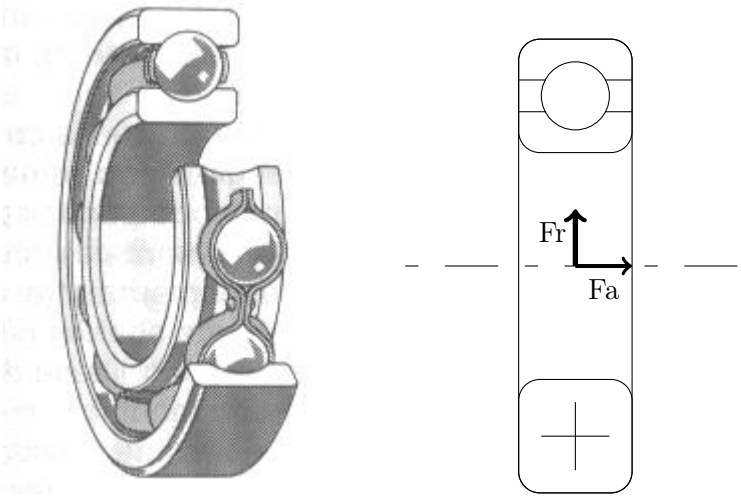
Correction :

On incrémente les secondes et on regarde si on dépasse les 59 secondes. Si c'est le cas, on incrémente les minutes et on met les secondes à 0. On regarde alors si on dépasse les 59 minutes, et même chose avec les heures.

```
1  secondes : Entier
2  minutes : Entier
3  heures : Entier
4
5  secondes ⟵ secondes + 1
6  si secondes > 59 alors
7  |  secondes ⟵ 0
8  |  minutes ⟵ minutes + 1
9  |  si minutes > 59 alors
10 |  |  minutes ⟵ 0
11 |  |  heures ⟵ heures + 1
12 |  |  si heures > 23 alors
13 |  |  |  heures ⟵ 0
14 |  |  finsi
15 |  finsi
16 finsi
```

Exercice 8 [Calcul de la durée de vie d'un roulement à billes]

Prenons un roulement à billes à contact droit. Ces roulements supportent des charges radiales et, dans une certaine mesure, axiales. On note F_a la force axiale subie par le roulement et F_r la force radiale.



Les roulements à billes ont les valeurs caractéristiques suivantes :
- Une capacité de chargement statique, notée C_0, dont la valeur est fournie par le fabriquant
- Une capacité de chargement dynamique, notée C, dont la valeur est fournie par le fabriquant
- Un facteur limitatif, noté e, dont la valeur est fournie par le fabriquant
- Un coefficient radial du roulement, noté X, déterminé par le calcul
- Un coefficient axial du roulement, noté Y, dont la valeur est fournie par le fabriquant

Pour estimer par le calcul la durée de vie d'un roulement à billes, on procède comme suit :

1. On calcule le rapport F_a/C_0

2. Si ce rapport est inférieur ou égal au facteur limitatif, alors la charge axiale n'influe pas. On calcule alors la charge dynamique équivalente par la formule $P = F_r$
3. Sinon, on calcule la charge dynamique équivalente par la formule $P = F_r X + F_a Y$, en prenant $X = 0,56$ et la valeur de Y fournie par le fabriquant
4. La durée de vie du roulement à billes en millions de rotations est obtenue par la formule $L = (C/P)^3$

Écrire un algorithme qui effectue un calcul de roulement pour un type de roulement donné (on suppose que C_0, e et Y sont connus) prenant en entrée les forces axiale et radiale appliquées au roulement.

Correction :

```
1  /* Données d'entrée */
2  Fr : Reel
3  Fa : Reel
4  /* Paramètres connus */
5  C0, C, e, X, Y : Reels
6  /* Variables internes à l'algorithme */
7  rapport : Reel
8  P : Reel
9
10 rapport ⟵ Fa/C0
11 si rapport ≤ e alors
12 |  P = Fr
13 sinon
14 |  X ⟵ 0,56
15 |  P ⟵ Fr * X + Fa * Y
16 finsi
17 L ⟵ (C/P)^3
18 retourner L
```

Exercice 9 [Minimum d'un tableau]

Écrire un algorithme qui retourne la valeur minimum contenue dans un tableau d'entiers.

Correction :

```
1  début
2  |  /* Variables d'entrée */
3  |  tableau : Tableau d'entiers
4  |  taille : Entier
5  |  /* Variable de sortie */
6  |  minimum : Entier
7  |  /* Initialisation */
8  |  minimum ⟵ tableau[0]
9  |  for i ← 1 to taille − 1 pas 1 do
10 |  |  if tableau[i] < minimum then
11 |  |  |  minimum = tableau[i]
12 |  |  endif
13 |  endfor
14 |  retourner minimum
15 fin
```