Correction des exercices de renforcement de A1

Table des matières

Exercice 17 :

Énoncé :
L'algorithme ci-dessous a pour but de calculer le prix à payer en fonction du nombre d'objets acheté d'un produit. Le produit coûte 2€50 pièce.
Corriger le pseudo-code suivant :
1 A est un entier et B est un réel.
2 Lire A
3 A prend la valeur 2.50B
4 ECRIRE B

Erreurs :

- choisir un nom de variable significatif ; par exemple :prix et n pour nombre
- expliciter toutes les opérations

Correction :
1 n est un entier et prix est un réel.
2 Lire n
3 prix prend la valeur 2.50*n
4 ECRIRE prix

Exercice 18 :

Énoncé :
Écrire un algorithme en pseudo code qui calcule la moyenne de cinq nombres a, b, c, d et e.
Le résultat sera stocké dans une variable moy.

Correction :
méthode sans boucle répétitive :

```
1 Lire a                    // Initialisation par l'utilisateur des 5 variables a, b, c, d et e
2 Lire b
3 Lire c
4 Lire d
5 Lire e
6 moy←a+b+c                 // Initialisation de moy
7 moy←moy/3                 // modification de moy
8 ECRIRE moy                // Affichage de moy
```

méthode avec boucle répétitive :

```
1 moy←0                     // Initialisation de moy
2 for i←1 to 5              // répétition 5 fois du bloc suivant
3       Lire a              // Saisie d'un des nombres par l'utilisateut
4       moy←moy+a           // Modification de moy
5 moy←moy/5
6 ECRIRE moy                // Affichage de moy
```

Exercice 19 :

Énoncé :
Ecrire un algorithme qui renvoie le max de deux nombres a et b. Le résultat sera stocké dans une variable max.

Correction :

```
1 Lire a                    // Initialisation par l'utilisateur des 2 variables a et b
2 Lire b
3 if a<b then               // Comparaison des deux nombres
4       max←b               // stockage du maximum
5 else
6       max←a               // stockage du maximum
7 ECRIRE max                // Affichage du maximum
```

Exercice 20 :

Énoncé :
La factorielle d'un nombre entier non nul (factorielle se note avec un !). Par exemple 4! =4×3×2×1=24.
Ainsi, 1!=1 et pour n>1 : n!=n×(n−1)×...×2×1=n×(n−1)!

Écrire un algorithme qui demande un nombre entier n non nul de départ, et qui calcule n!, c'est-à-dire le produit des entiers jusqu'à ce nombre n.

Correction :
Il suffit d'utiliser une boucle répétitive FOR ou WHILE et d'utiliser le lien :
n!=n×(n−1)×...×2×1=n×(n−1)!
Avec FOR :

```
1 Lire n                    // Initialisation par l'utilisateur de la variable  n
2 res ← 1
3 for i←2 to n              // répétition (n-1) fois du bloc suivant
4       res←res×i           // multiplication pour utiliser  n!=n×(n−1)×...×2×1
5 ECRIRE res
```

Avec WHILE :

```
1 Lire n                    // Initialisation par l'utilisateur de la variable  n
2 res ← 1
3 i← 1                      // Compteur de tour
4 while i<=n                // répétition (n-1) fois du bloc suivant
5       res←res×i           // multiplication pour utiliser  n!=n×(n−1)×...×2×1
6       i← 1+1              // incrémentation du compteur
7 ECRIRE res
```

Exercice 21 :

Énoncé :
Écrire un algorithme qui stocke dans une variable min le minimum de trois variables a, b et c données.

Correction :
Il suffit d'imbriquer deux tests IF THEN ELSE :

```
1 Lire a                    // Initialisation par l'utilisateur des 3 variables  a, b et c
2 Lire b
3 Lire c
4 if a>b then               // Comparaison des deux nombres
5       if b>c then
6               min←c       // stockage du minimum
7       else                // ici la condition « b>c » est fausse donc b<=c
8               min←b       // stockage du minimum
9 else                      // ici la condition « a>b » est fausse donc a<=b
10      if a>c then
11              min←c
12      else                // ici la condition « a>c » est fausse donc a<=c
13              min←a
```

// pas d'affichage demandé ici, seulement un stockage : pas « d'ECRIRE min » ici

Exercice 22 :

Énoncé :
1. Écrire un algorithme en pseudo code qui renvoie le min des éléments d'une liste nommée liste1 ayant n éléments.
2. Écrire une trace d'exécution en prenant la liste1 [7,4,3,8,9] de l'exemple.

Correction :
1. Il suffit de balayer toutes la liste en comparant chaque élément avec le maximum temporaire

```
1 Lire liste1                    // Initialisation par l'utilisateur de la liste listel
2 Lire n                         // Initialisation par l'utilisateur de la taille n de la liste liste1
3 min ← liste1[0]                // Stockage du premier élément de la liste
4 for i←1 to n-1                 // Balayage de tous les éléments de la liste ;le dernier est
liste1[n-1].
5       if min>liste1[i]
6               min← liste1[i]   // Nouveau minimum temporaire trouvé
7 ECRIRE min                     // Affichage un minimum trouvé
```

2. Écrire une trace d'exécution en prenant la liste1 de l'exemple.
Correction :

#ligne	min	i	liste[i]	Commentaires
1				Initialisation : liste1=[7,4,3,8,9]
2				Saisie de n=5.
3	7		7	Le premier élément de la liste est liste1[0] soit 7
4		1		première itération avec i=1 :
5			4	7>4 VRAI : on rentre dans la ligne 6
6	4			max←4
4		2		deuxième itération avec i=2 :
5			3	4>3 VRAI : on rentre dans la ligne 6
6	3			max←3
4		3		troisième itération avec i=2 :
5			8	8>3 FAUX : on ne rentre pas dans la ligne 6 : retour à la ligne 4
4		4		dernière itération avec i=4 :
5			9	9>3 FAUX : on ne rentre pas dans la ligne 6
7	3			Affichage du résultat obtenu