TD N°4 : Le traitement répétitif (les boucles)

Exercice 01_4:

Exécuter les algorithmes suivants avec la valeur 67 et 112 :

```
Algo Exo1_4_a
Variables
  N : Entier
Début
  Lire(N)
  TQ N < 100 :
    N ← N + 10
  FTQ
  Ecrire(N)
Fin
```

```
Algo Exo1_4_b
Variables
  N : Entier
Début
  Lire(N)
  Rpt
    N ← N + 10
  Jsq N ≥ 100
  Ecrire(N)
Fin
```

```
Algo Exo1_4_c
Variables
  N, i : Entier
Début
  Lire(N)
  Pr i ← 1 à 4 :
    N ← N + 10
  FPr
  Ecrire(N)
Fin
```

Quels sont les algorithmes équivalents parmi les trois ci-dessus ?

Exercice 02_4:

Les algorithmes de chacune des premières cases des tableaux suivants sont écrits syntaxiquement corrects.
Repérer les erreurs syntaxiques dans chacune des écritures suivantes (de 2 à 5), s'il y en a.

1

```
Algo Exo2_4_a1
Variables
  N, i, S : Entier
Début
  Lire(N)
  S ← 0
  Pr i ← 1 à N :
    S ← S + 10
  FPr
  Ecrire(S)
Fin
```

2

```
Algo Exo2_4_a2
Variables
  N, i, S : Entier
Début
  Lire(N)
  S ← 0
  Pr i = 1 à N :
    S ← S + 10

  Ecrire(S)
Fin
```

3

```
Algo Exo2_4_a3
Variables
  N, i, S : Entier
Début
  Lire(N)
  S ← 0
  Pour i ← 1 à N
    S ← S + 10
  FinPour
  Ecrire(S)
Fin
```

4

```
Algo Exo2_4_a4
Variables
  N, i, S : Entier
Début
  Lire(N)
  S ← 0
  Pour i ← 1 To N :
    S ← S + 10
  fpr
  Ecrire(S)
Fin
```

5

```
Algo Exo2_4_a5
Variables
  N, i, S : Entier
Début
  Lire(N)
  S ← 0
  Pr i allant de 1 à N :
    S ← S + 10
  FPour
  Ecrire(S)
Fin
```

1

```
Algo Exo2_4_b1
Variables
  N : Entier
Début
  Lire(N)
  TQ N < 100 :
    N ← N + 10
  FTQ
  Ecrire(N)
Fin
```

2

```
Algo Exo2_4_b2
Variables
  N : Entier
Début
  Lire(N)
  TQ N < 100
    N ← N + 10
  FinTQ
  Ecrire(N)
Fin
```

3

```
Algo Exo2_4_b3
Variables
  N : Entier
Début
  Lire(N)
  TantQue N < 100 :
    N ← N + 10
  FinTantQue
  Ecrire(N)
Fin
```

4

```
Algo Exo2_4_b4
Variables
  N : Entier
Début
  Lire(N)
  Tq N < 100:
    N ← N + 10

  Ecrire(N)
Fin
```

5

```
Algo Exo2_4_b5
Variables
  N : Entier
Début
  Lire(N)
  TQ N<100 : et N >0
    N ← N + 10
  FTQ
  Ecrire(N)
Fin
```

1

```
Algo Exo2_4_c1
Variables
  N : Entier
Début
  Lire(N)
  Rpt
    N ← N + 10
  Jsq N ≥ 100
  Ecrire(N)
Fin
```

2

```
Algo Exo2_4_c2
Variables
  N : Entier
Début
  Lire(N)
  Répéter
    N ← N + 10
  Jsq N ≥ 100
  Ecrire(N)
Fin
```

3

```
Algo Exo2_4_c3
Variables
  N : Entier
Début
  Lire(N)
  Repeter
    N ← N + 10
  Jusqu'à N ≥ 100
  Ecrire(N)
Fin
```

4

```
Algo Exo2_4_c4
Variables
  N : Entier
Début
  Lire(N)
  Rpt :
    N ← N + 10
  Jsq N ≥ 100 :
  Ecrire(N)
Fin
```

5

```
Algo Exo2_4_c5
Variables
  N : Entier
Début
  Lire(N)
  Rpt N ≥ 100
    N ← N + 10
  Jsq
  Ecrire(N)
Fin
```

Exercice 03_4: (Dans chacune des questions de l'exercice, n'utilisez que la boucle **pour,** avec la boucle **répéter** pour les cas d'erreurs)

a- Écrire un algorithme qui donne la main à l'utilisateur pour donner 100 nombres entiers, et affiche le double de chaque nombre saisi.

b- Écrire un algorithme qui donne la main à l'utilisateur pour donner 100 nombres entiers, et affiche le nombre d'éléments pairs parmi les 100 saisis.

c- Écrire un algorithme qui demande à l'utilisateur de saisir 100 caractères, caractère par caractère, et affiche le nombre de fois où l'utilisateur a saisi le caractère **e**.

d- Écrire un algorithme qui demande 100 nombres entiers, et calcule et affiche la somme des éléments pairs parmi les 100 saisis.

e- Écrire un algorithme qui demande 100 nombres entiers, et calcule et affiche le produit des éléments pairs parmi les 100 saisis.

f- Écrire un algorithme qui demande un nombre **N** strictement positif, et calcule et affiche la somme des **N** premiers nombres naturels $(0 + 1 + 2 + 3 + 4 + \cdots + N - 1)$, sans avoir recours à la formule : $\frac{N(N+1)}{2}$

g- Écrire un algorithme qui demande un nombre **N** strictement positif, et calcule et affiche le carré de la somme des **N** premiers nombres naturels $(0 + 1 + 2 + 3 + 4 + \cdots + N - 1)^2$

h- Écrire un algorithme qui demande un nombre **N** strictement positif, et calcule et affiche le produit des **N** premiers nombres naturels non nuls $(1 \times 2 \times 3 \times 4 \times \ldots \times N)$.

i- Écrire un algorithme qui demande un nombre **N** strictement positif, et calcule et affiche la somme des **N** premiers nombres naturels pairs $(0 + 2 + 4 + \cdots 2(N - 1))$.

j- Écrire un algorithme qui demande un nombre **N** strictement positif, et calcule et affiche la somme des premiers nombres naturels pairs inférieurs à **N**.

k- Écrire un algorithme qui demande un nombre **N** strictement positif, et calcule et affiche la somme :

$$S = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{N}$$

l- Écrire un algorithme qui demande un nombre **N** strictement positif, et calcule et affiche la somme :

$$S = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \cdots + \frac{1}{N^2}$$

Exercice 04_4: (Dans chacune des questions de l'exercice, n'utilisez que la boucle **pour,** avec la boucle **répéter** pour les cas d'erreurs)

a- Écrire un algorithme qui demande un nombre entier strictement positif **N,** et calcule $2 \times N$, en ajoutant **2**, **N** fois avec une boucle. L'algorithme doit afficher le résultat final uniquement.
b- Écrire un algorithme qui demande un nombre entier strictement positif **N,** et calcule et affiche $\mathbf{2^N}$.
c- Écrire un algorithme qui demande un nombre réel **B** et un nombre entier strictement positif **P,** et calcule et affiche $\mathbf{B^P}$.
d- Écrire un algorithme qui demande un nombre réel **B** et un nombre entier **P,** et calcule et affiche $\mathbf{B^P}$.

Exercice 05_4 (supplémentaire) : (Dans chacune des questions de l'exercice, n'utilisez que la boucle **pour,** avec la boucle **répéter** pour les cas d'erreurs)

a- Écrire un algorithme qui calcule et affiche le produit de deux nombre entiers strictement positifs, en utilisant uniquement l'opération d'addition : $7 \times 5 = 7 + 7 + 7 + 7 + 7$.
b- Écrire un algorithme qui calcule et affiche le produit de deux nombre entiers, en utilisant uniquement l'opération d'addition).

Exercice 06_4 : (Dans chacune des 3 premières questions de l'exercice, n'utilisez que la boucle **pour,** avec la boucle **répéter** pour les cas d'erreurs)

a- Écrire un algorithme qui calcule et affiche la factorielle d'un nombre naturel N : $5! = 1 \times 2 \times 3 \times 4 \times 5$.
b- Écrire un algorithme qui calcule et affiche B^P, tel que B réel quelconque et P naturel.
c- Écrire un algorithme qui demande à l'utilisateur d'entrer $\boldsymbol{x}$, et calcule et affiche $\boldsymbol{e^x}$, par le développement de Taylor avec 10 fractions :

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots \frac{x^9}{9!}$$

Écrire pour la question **c**, une solution avec des boucles imbriquées et une autre avec une seule boucle.
d- Écrire un algorithme qui demande à l'utilisateur d'entrer $\boldsymbol{x}$, et calcule et affiche $\boldsymbol{e^x}$ par le développement de Taylor, avec une précision de deux (2) chiffres après la virgule.

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$

e- **(supplémentaire)** Écrire un algorithme qui permet de calculer la formule suivante d'élévation d'une somme, avec une précision de trois (3) chiffres après la virgule :

$$(1+x)^n = 1 + \frac{nx}{1!} + \frac{n(n-1)x^2}{2!} + \cdots$$

Exercice 07_4 :
a- Écrire un algorithme qui demande à l'utilisateur de saisir un nombre naturel non nul $\boldsymbol{N}$, et affiche tous ses diviseurs.
b- Écrire un algorithme qui demande un nombre naturel non nul $\boldsymbol{N}$, et affiche le nombre de ses diviseurs.
c- Écrire un algorithme qui demande un nombre naturel $\boldsymbol{N}$, et affiche s'il est premier ou pas. **Note** : un nombre est dit premier s'il est divisible par 1 et lui-même uniquement. Par convention, 1 n'est pas premier.
d- **(supplémentaire)** Écrire un algorithme qui calcule la somme des **Nb** premiers nombres premiers (**Nb** naturel non nul).

Exercice 08_4 :
Calculer et afficher le PGCD de deux nombres entiers strictement positifs, en utilisant l'algorithme d'Euclide. Qu'est ce qu'il faut changer dans le cas des nombres entiers positifs ou nuls ?

Exercice 09_4 (supplémentaire) :
- Deux nombres entiers sont premiers entre eux si leur PGCD (Plus Grand Commun Diviseur) égale 1.
- Soient deux entiers positifs **a** et **b** et soit PPCM (a, b) le plus petit multiple commun et PGCD (a, b) ; Alors on a la relation : $a \times b = PPCM(a,b) \times PGCD(a,b)$.

Écrire un algorithme qui pour deux entiers strictement positifs **a** et **b** vérifie s'ils ne sont pas premiers entre eux et dans ce cas utilise la formule précédente pour calculer le PPCM (**a**, **b**).

Exercice 10_4
Écrire un algorithme qui demande un nombre naturel et affiche sa conversion en binaire.

Exercice 11_4
Soit la suite de Fibonacci définit comme suit : $U_0 = 0$, $U_1 = 1$, $U_n = U_{n-1} + U_{n-2}$ pour tout n > 1
a- Écrire un algorithme qui demande à l'utilisateur un indice **n**, et calcule le terme de la suite correspondant $\mathbf{U_n}$
b- Écrire un algorithme qui affiche les **N** premiers termes de la suite de Fibonacci.
c- Écrire un algorithme qui calcule et affiche la somme des **N** premiers termes de la suite de Fibonacci.

Exercice 12_4 (supplémentaire) :
Écrire un algorithme qui vérifie, dans une suite de caractères lus se terminant par ' > ', entrée caractère par caractère, l'apparition du caractère '!'.

Exercice 13_4 (supplémentaire) :
Il existe une méthode pour multiplier deux nombres où il ne faut que savoir multiplier ou diviser par deux, et additionner. On appelle cette méthode "multiplication à la russe".
- A = le petit nombre / B = le grand nombre
- A est divisé par deux et remplacé par la partie entière de la division jusqu'à ce qu'il soit égal à 1 ;
- En même temps B est remplacé par son double ;
- Résultat = la somme des nombres B quand A est impair.

Exemple :
Pour l'opération $238 \times 13 = 3094$

A	B	Résultat
13	238	238
6	476	238 (ne change pas car A est pair)
3	952	238 + 952 = 1190
1	1904	1190 + 1904 = 3094 (on arrête car A = 1)

Écrire un algorithme qui réalise la multiplication "à la russe" de deux nombres entiers positifs X et Y.

Exercice 14_4 (supplémentaire) :
Un nouvel employé est payé un salaire initial X dinars puis il est augmenté de Y dinars tous les Z mois.
Q1- Écrire un algorithme qui demande à l'utilisateur le nombre de mois de travail depuis son recrutement et affiche le salaire de ce mois-ci sachant que X = 20 000 DA, Y = 500 DA et Z = 3 mois.
Exemples :
- Pour un nombre de mois de 4 à 6, le salaire est 20 500.
- Pour un nombre de mois égal à 10, le salaire est 21 500.

Q2- Écrire un algorithme qui permet à l'utilisateur de saisir X, Y, Z et W (le nombre de mois de travail) pour afficher la somme des salaires depuis son recrutement jusqu'à ce jour.
Q3- voici deux propositions :
- Une augmentation de Y2 DA chaque 6 mois.
- Une augmentation de Y2 DA chaque année.

Quelle est la relation entre Y1et Y2 pour que les deux propositions donnent la même somme de salaires sur 3ans ? Justifiez votre réponse.