Travail pratique 6

Travail présenté dans le cadre du cours Logique de programmation

> Par Jérémie Normand Groupe 00001

Somme 1 ... 100

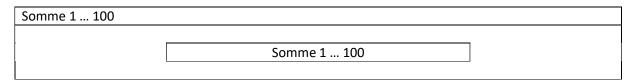
Énoncé

Calculer la somme des entiers de 1 à 100. Afficher le résultat dans une boîte à message.

Données

Entrées	Constantes	Sorties
	NombreDeDepart	Somme
	NombreDeFin	

Conception visuelle



Résultat de la somme des	s nombre de 1 à 100		

Algorithmes

CalculerSommeDeUnACent()

NombreDeDepart \leftarrow 1 NombreDeFin \leftarrow 100

Somme ← 0

Pour NombreCourant ← NombreDeDepart jusqu'à NombreDeFin avec saut de 1 Somme ← Somme + Nombre

Afficher Somme

Essais

Variables	Entrées 1	Sorties 1	Entrées 2	Sorties 2
Somme		5 050		5 050

	Variables	
Instructions	Somme	
0.	0	
1.	1	
2.	3	
99.	4 950	
100.	5 050	

Somme X ... Y

Énoncé

Calculer la somme des entiers compris entre deux nombres entiers positifs non nuls saisis par l'utilisateur. Valider les nombres lus. Afficher le résultat dans une boîte à message.

Données

Entrées	Constantes	Sorties
NombreDeDepart		Somme
NombreDeFin		

Conception visuelle

Somme X Y		
	Nombre de départ (X) Nombre de fin (Y)	
	Somme X Y	ı

Résultat de la somme des	s nombre de X à Y		

Algorithmes

CalculerSommeDeXaY()

Lire NombreDeDepart, NombreDeFin

SontValides ← ValiderNombresPositifsNonNuls(NombreDeDepart, NombreDeFin)

Si SontValides = FAUX

Renvoyer

Somme ← 0

Si NombreDeDepart < NombreDeFin

Pour NombreCourant ← NombreDeDepart jusqu'à NombreDeFin avec saut de 1 Somme ← Somme + NombreCourant

Sinon

Pour NombreCourant ← NombreDeDepart jusqu'à NombreDeFin avec saut de -1 Somme ← Somme + NombreCourant

Afficher Somme

ValiderNombresPositifsNonNuls(Nombre1, Nombre2)

 $\mathsf{SontValides} \gets \mathsf{VRAI}$

Si Nombre1 ≤ 0

SontValides \leftarrow FAUX

Si Nombre2 ≤ 0

SontValides ← FAUX

Renvoyer SontValides

Essais

Variables	Entrées 1	Sorties 1	Entrées 2	Sorties 2
NombreDeDepart	3		100	
NombreDeFin	55		1 100	
Somme		1 537		600 600

		Variables				
Instructions	NombreDeDepart	NombreDeFin	Somme			
0.	3	55	0			
1.	3	55	3			
2.	3	55	7			
•••		•••	•••			
51.	3	55	1 482			
52.	3	55	1 537			

Puissance

Énoncé

Lire deux nombres entiers positifs non nuls, valider les nombres lus et calculer le premier à la puissance du second sans utiliser la méthode Math.Pow. Afficher le résultat dans une boîte à message.

Données

Entrées	Constantes	Sorties
Base		Resultat
Puissance		

Conception visuelle

Puissance		
	Base	
	Puissance	
	Puissance	
Résultat de la puis	ssance	

Algorithmes

CalculerPuissance()

Lire Base, Puissance

SontValides ← ValiderNombresPositifsNonNuls (Base, Puissance)

Si SontValides = FAUX

Renvoyer

Resultat ← Base

Tant que Puissance > 1

Resultat ← Resultat * Base

Puissance ← Puissance – 1

Afficher Resultat

Essais

Variables	Entrées 1	Sorties 1	Entrées 2	Sorties 2
Base	6		11	
Puissance	4		5	
Resultat		1 296		161 051

	Variables			
Instructions	Base	Puissance	Resultat	
0.	6	4	6	
1.	6	4	36	
2.	6	4	216	
3.	6	4	1 296	

Décompte

Énoncé

Lire deux (2) nombres entiers négatifs ou nuls, valider les nombres et afficher dans un contrôle de type ListBox tous les nombres compris entre le plus grand et le plus petit.

Données

Entrées	Constantes	Sorties
NombreX		Decompte
NombreY		

Conception visuelle

Décompte		
	Premier nombre (X) Deuxième nombre (Y)	
	Décompte	

Algorithmes

AfficherDecompte()

Lire NombreX, NombreY

 $SontValides \leftarrow ValiderNombresNegatifsOuNuls(NombreX, NombreY)$

Si SontValides = FAUX

Renvoyer

Decompte ← []

Si NombreX < NombreY

Pour Nombre \leftarrow NombreX jusqu'à NombreY avec saut de 1

Decompte.Ajouter(Nombre)

Sinon

Pour Nombre ← NombreY jusqu'à NombreX avec saut de 1 Decompte.Ajouter(Nombre)

Afficher Decompte

ValiderNombresNegatifsOuNuls(Nombre1, Nombre2)

SontValides ← VRAI

Si Nombre1 > 0

SontValides ← FAUX

Si Nombre2 > 0

SontValides ← FAUX

Renvoyer SontValides

Essais

Variables	Entrées 1	Sorties 1	Entrées 2	Sorties 2
NombreX	-4		-1	
NombreY	0		-4	
Decompte		[-4; -3; -2; -1; 0]		[-4; -3; -2; -1]

		Variables				
Instructions	NombreX	NombreY	Decompte			
0.	-4	0	[]			
1.	-4	0	[-4]			
2.	-4	0	[-4; -3]			
3.	-4	0	[-4; -3; -2]			
4.	-4	0	[-4; -3; -2; -1]			
5.	-4	0	[-4; -3; -2; -1; 0]			

Étoiles

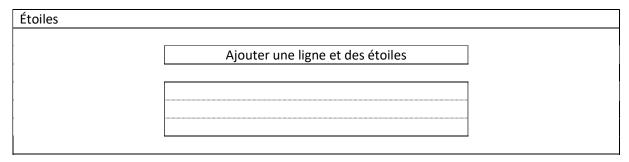
Énoncé

Ajouter des étoiles (*) dans un contrôle de type ListBox à chaque fois que vous appuyez sur le bouton. Chacune des lignes supplémentaires compte une étoile additionnelle par rapport à la ligne précédente.

Données

Entrées	Constantes	Sorties
		NombreDeLignesEtEtoiles

Conception visuelle



Algorithmes

NombreDeLignesEtEtoiles \leftarrow 0

AjouterLigneEtEtoile()

 $Nombre De Lignes Et Etoiles \leftarrow Nombre De Lignes Et Etoiles + 1$

Ligne ← ""

Pour Etoile ← 1 jusqu'à NombreDeLignesEtEtoiles avec saut de 1

Ligne ← Ligne + "*"

Afficher Ligne

Essais

Variables	Entrées 1	Sorties 1	Entrées 2	Sorties 2
NombreDeLignesEtEtoiles	3	4	0	1

	Variables
Instructions	NombreDeLignesEtEtoiles
0.	0
1.	1
2.	2

Répète

Énoncé

Lire un nombre entier et un nombre de répétitions supérieur à 1. Afficher ce nombre dans un contrôle de type TextBox autant de fois que lu en entrée. Par exemple : pour un nombre lu à 3 et un nombre de répétition lu à 5, la chaîne 3 3 3 3 est affichée.

Données

Entrées	Constantes	Sorties
Nombre		Chaine
NombreDeRepetition	-	

Conception visuelle

Répète			
	Nombre Nombre de répétitions		
	Répéte	r nombre	

Algorithmes

```
RepeterNombre()
   Lire Nombre, NombreDeRepetitions
   EstValide ← ValiderNombrePositifSuperieurAUn(NombreDeRepetitions)
   Si EstValide = FAUX
       Renvoyer
   Chaine ← ""
   Pour NombreCourant ← 1 jusqu'à NombreDeRepetitions avec saut de 1
       Chaine ← Chaine + Nombre + " "
   Afficher Chaine
```

ValiderNombrePositifSuperieurAUn(Nombre)

EstValide ← VRAI Si Nombre < 1 EstValide ← FAUX

Renvoyer EstValide

Essais

Variables	Entrées 1	Sorties 1	Entrées 2	Sorties 2
Nombre	-12		3	
NombreDeRepetitions	1		5	
Chaine		"-12 "	•	"3 3 3 3 3 "

		Variables	
Instructions	Nombre	NombreDeRepetitions	Chaine
0.	3	5	1111
1.	3	5	"3 "
2.	3	5	"3 3 "
3.	3	5	"3 3 3 "
4.	3	5	"3 3 3 3 "
5.	3	5	"3 3 3 3 3 "

Renversant

Énoncé

Lire un nombre entier positif supérieur ou égal à 10 et l'afficher à l'envers. Par exemple : le nombre lu à 12345, le résultat affiché sera 54321. Vous devez utiliser la division et le modulo donc travailler sur un nombre entier non pas sur une chaîne.

Données

Entrées	Constantes	Sorties
Nombre		Inverse

Conception visuelle

Renversant	
	Nombre
	Inverser

Algorithmes

```
InverserNombre()<sup>i</sup>

Lire Nombre

EstValide ← ValiderNombreSuperieurOuEgalADix(Nombre)

Si EstValide = FAUX

Renvoyer

Inverse ← 0

Tant que Nombre > 0

Restant ← Nombre % 10

Inverse ← (Inverse * 10) + Restant

Nombre ← Nombre / 10

Afficher Inverse

ValiderNombreSuperieurOuEgalADix(Nombre)

EstValide ← VRAI
```

Renvoyer EstValide

EstValide ← FAUX

Si Nombre < 10

Essais

Variables	Entrées 1	Sorties 1	Entrées 2	Sorties 2
Nombre	137		12345	
Inverse		731		54321

Trace

	Varia	ables
Instructions	Nombre	Inverse
0.	137	0
1.	13	7
2.	1	73
3.	0	731

Mise en commun

Conception visuelle

Somme 1 100	Ajouter une ligne et des étoiles
Nombre X Nombre Y	
	 Nombre
Somme X Y	Nb de répétitions
Base Puissance	Répéter nombre
Puissance	
	Nombre
Nombre X Nombre Y	Inverser
Décompte	

ⁱ https://www.c-sharpcorner.com/code/3430/reverse-number-in-python.aspx