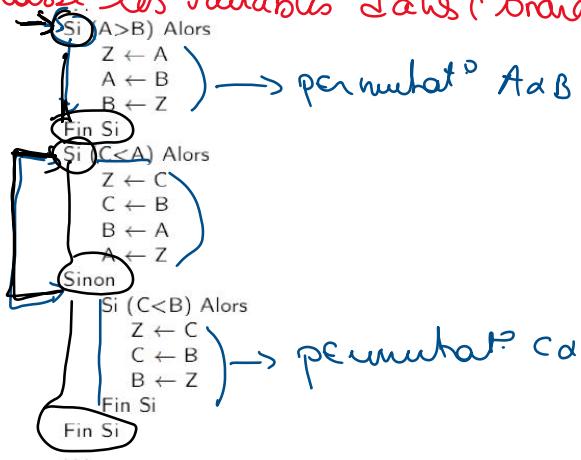


Exercice 1:

Que fait cette séquence d'instructions dans laquelle A, B, C et Z sont des variables réelles ?

→ il classe les variables dans l'ordre croissant: $A < B < C$

A	B	C	Z
7	3	1	na
7	3	1	7
3	3	1	7
3	4	1	7
3	7	1	1
3	7	7	1
1	3	7	1
<u>STOP</u>			



A	B	C	Z
2	4	1	na
2	4	1	1
2	4	4	1
2	2	4	1
7	2	4	1
<u>STOP</u>			

On fera une trace pas à pas de cette séquence avec différentes valeurs de A, B et C :

$$\times A=2; B=5; C=7$$

$$\times A=5; B=1; C=2$$

$$\times A=3; B=1; C=5$$

$$\times A=7; B=3; C=1$$

$$\times A=3; B=6; C=5$$

$$A=2; B=4; C=1$$

A	B	C	Z
2	5	7	na
<u>STOP</u>			

A	B	C	Z
3	1	5	na
3	1	5	3
1	3	5	3
<u>STOP</u>			

A	B	C	Z
3	6	5	na
3	6	5	5
3	6	6	5
3	5	6	5
<u>STOP</u>			

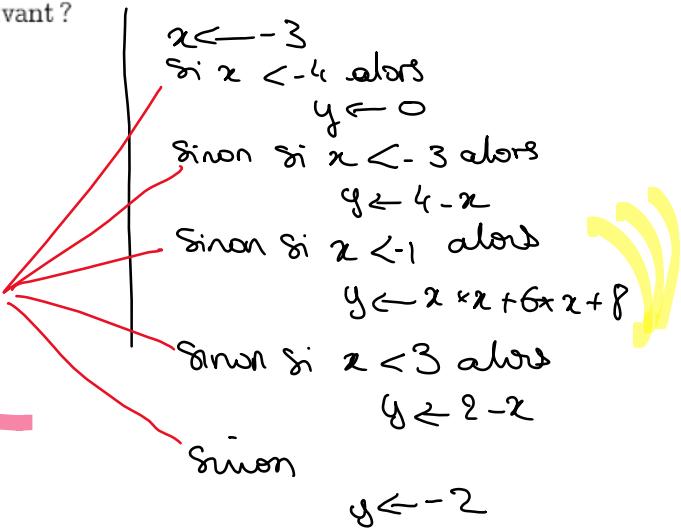
A	B	C	Z
5	1	2	na
5	1	2	5
1	1	2	5
1	1	5	5
1	1	5	2
1	2	5	2
1	2	5	2
<u>STOP</u>			

Exercice 2 :

Quelle est la valeur de y à la fin du programme suivant ?

x=-3
if x<-4 : y=0
elif x<-3 : y=4-x
elif x<-1 : y=x*x+6*x+8
elif x<3 ; y=2-x
else : y=-2

= else
= sinon si



$$\begin{aligned}
&\text{Si } x = -3 \quad y = 0 \\
&\quad y = 4 - x \\
&\quad y = 4 - (-3) \\
&\quad y = 4 + 3 \\
&\quad y = 7
\end{aligned}$$

$$\begin{aligned}
&y = x * x + 6 * x + 8 \\
&= (-3) * (-3) + 6 * (-3) + 8 \\
&= 9 - 18 + 8 \\
&= -1
\end{aligned}$$

$$y = -1$$

En Python
 = : affectation
 == : comparaison

Exercice 3 :

Quelle est la valeur de y à la fin du programme suivant ?

```
x=3
y=-2
if x<y : y=y-x
elif x==y : y=0
else : y=x-y
```

$$y = 3 - (-2) = \underline{\underline{5}}$$

$x \leftarrow 3$
 $y \leftarrow -2$
 si $x < y$ alors
 $y \leftarrow y - x$
 sinon si $x = y$ alors
 $y \leftarrow 0$
 sinon
 $y \leftarrow x - y$

Exercice 4 :

Ecrire une série d'instructions affichant la valeur absolue d'un nombre x .

Valeur absolue d'un nombre $x = |x| = \begin{cases} x & \text{si } x \geq 0 \\ -x & \text{si } x < 0 \end{cases}$ (ou $|x| = \max\{x, -x\}$) (ou $|x| = \sqrt{x^2}$)

1^{ère} version : création d'une var :

```
x← Saisir ("Donner un nbr")
Si x >= 0 alors
    abs← x
Sinon
    abs← -x
afficher ("La valeur absolue de", x, "est", abs)
```

2^{ème} version : Pas besoin de x
 $x \leftarrow$ Saisir ("Donner un nbr")
 Si $x < 0$ alors
 $x \leftarrow -x$

afficher ("La valeur absolue recherchée est", x)

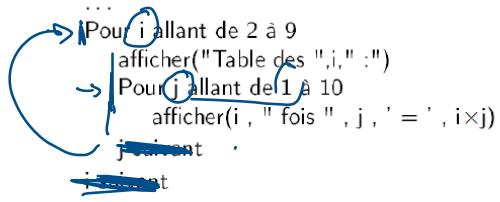
Exercice 5 :

Ecrire une série d'instructions qui saisit une note a (sur 20), puis qui affiche « ajourné » si $a < 8$, « oral » si $8 \leq a < 10$ et « admis » si $a \geq 10$.

```
a← Saisir ("Quelle est votre note?")
Si a < 8 alors
    afficher ("ajourné")
Sinon si a < 10 alors
    afficher ("oral")
Sinon
    afficher ("admis")
Fin Si
```

III. Imbrication

Là encore, les structures peuvent être imbriquées les unes dans les autres (boucle dans une boucle). Par exemple, la séquence suivante affiche les tables de multiplications de 2 à 9 :



$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ en maths

$M = [[1, 2, 3], [4, 5, 6]]$ en algo

Pour i allant de 0 à nb lignes
Pour j allant de 0 à nb colonnes
afficher ---

$i=2 \rightarrow$ table des 2

$$\begin{aligned} 2 \text{ fois } 1 &= 2 \\ 2 \text{ fois } 2 &= 4 \\ 2 \text{ fois } 3 &= 6 \\ \vdots & \\ 2 \text{ fois } 10 &= 20 \end{aligned}$$

$i=3 \rightarrow$ table des 3

$$\begin{aligned} 3 \text{ fois } 1 &= 3 \\ 3 \text{ fois } 2 &= 6 \\ 3 \text{ fois } 3 &= 9 \\ \vdots & \\ 3 \text{ fois } 10 &= 30 \end{aligned}$$

$i=9 \rightarrow$ table des 9

$$\begin{aligned} 9 \text{ fois } 1 &= 9 \\ \vdots & \\ 9 \text{ fois } 10 &= 90 \end{aligned}$$

Exercice 1 :

Que contiennent les variables V et W après cette séquence d'instructions dans laquelle A et B sont des entiers positifs ? (on fera un tracé pas à pas avec A=5 ; B=12, puis avec A=17 ; B=5, et enfin avec A=12 ; B=3)

```

    ...
    W ← 0
    V ← A
    Tant que (V ≥ B)
        V ← V - B
        W ← W + 1
    Fin Tant que
    ...

```

$$\begin{array}{l} A=5 \\ B=12 \end{array}$$

	V	W
	5	0
stop		

	V	W
A = 17	17	0
B = 5	12	1
	7	2
	2	3
stop		

	V	W
A = 12	12	0
B = 3	9	1
	6	2
	3	3
0		4
stop		

→ Cet algorithme effectue la division euclidienne de A par B.

$$A = \underbrace{W \times B}_{\text{Quotient}} + \underbrace{V}_{\text{Reste}}$$

Exercice 2:

Ecrire un algorithme qui calcule la somme des n premiers nombres entiers.

Fonction Somme()
 $n \leftarrow \text{Saisir ("Donner un nb entier")}$ ou Fonction Somme(n) ==

S ← 0 # une somme s' initialise à 0

pour i allant de 1 à n

S ← S + i

fin pour

retourner (S) # ou afficher (S)

S	i
0	1
1	2
2	3
3	4

$$S = 1+2+3$$

$$n = 3$$

Exercice 2 bis :

Ecrire un algorithme qui calcule la somme des n premiers carrés: $1^2 + 2^2 + \dots + n^2$.

Fonction SommeCarre(n)

S ← 0

Pour i allant de 1 à n

S ← S + i * i

Fin Pour

Retourner (S)

Exercice 3 : Factorielle

Ecrire un algorithme demandant à l'utilisateur un entier n, et calculant la factorielle $n!$ de n, définie par: $0! = 1$ et $n! = 1 \times 2 \times \dots \times n$ pour $n \geq 1$

$$4! = 1 \times 2 \times 3 \times 4 = 24$$

Fonction Factorielle(n)

F ← 1 # Un produit s'initialise à 1.

Pour i allant de 1 à n

F ← F * i

Fin Pour

Retourner (F)

Indice et 5

Nb secret ← alea(1, 1000) # Pour la création
du nb secret

random