Tracer l'exécution d'une boucle POUR

Une boucle **pour** permet de répéter une ou plusieurs instructions un nombre de fois bien définie. Pour rappel, une boucle définie (ou boucle **pour**) dit que l'on doit répéter les instructions pour chaque valeur entière comprise entre une valeur initiale et une valeur finale de la variable de boucle (généralement i).

L'exécution d'une boucle for comprend plusieurs étapes :

1) Initialiser la variable de boucle (**i ← initiale**).
2) Évaluation de la condition de continuation (**i <= finale**); si la condition n'est pas remplie, aller au point 6)(fin de la boucle).
3) Exécuter les instructions du corps de la boucle.
4) Incrémenter la variable de boucle (**i ← i + 1**)
5) Recommencer à au point 2).
6) Fin de la boucle.

Le programme suivant calcule la somme des 5 premiers nombres entiers à l'aide d'une boucle **pour**. La valeur initiale est 1 (**i ← 1**), la valeur finale est 5 (**i <= 5**) et le pas est de 1 (**i ← i + 1**).

```
1  Algorithme Exemple3
2  Variable sum,i: eniter
3  Debut
4     sum ← 0;
5     pour(i ← 1 à 5)faire
6           sum ← sum + i;
7     Finpour
8
9          Ecrire(sum);
10 Fin
```

Le tableau suivant montre la trace de l'exécution de ce programme. On peut remarquer que même si le programme est très court, la trace, elle, est plutôt longue. Cela vient du fait que lorsque l'on fait la trace d'un programme qui contient une boucle, on doit répéter les instructions du corps de la boule, l'instruction d'incrémentation de la variable de boucle et le test de la condition de continuation autant de fois que nécessaire.

N° étape	Instruction	Description / Remarques	État des variables locales (après exécution de l'instruction)				
			sum	i			
1	Variable sum, i : entier	Déclaration des variables	-	-	-		
2	sum ← 0	Affectation d'initialisation de sum par l'expression de valeur 0	0				

3	(pour) i ← 1	affectation de la variable de boucle par la valeur de l'expression 1	0	1			
4	(pour)i <=5	Évalue l'expression 1 <= 5 =vrai	0	1			
5	sum ← sum + i	Evalue l'expression sum + i = 0 + 1 = 1 et affecte la valeur à la variable sum	1	1			
6	(pour) i← i+ 1	Evalue l'expression i+ 1 = 1+1=2 et affecte la valeur à la variable i	1	2			
7	(pour) i <= 5	Evalue l'expression 2<= 5 = vrai	1	2			
8	sum ← sum + i	Evalue l'expression sum + i = 1 +2 = 3 et affecte la valeur à la variable sum	3	2			
9	(pour) i← i+1	Evalue l'expression i+1 = 2+1 = 3 et l'affecte la valeur à la variable i	3	3			
10	pour) i <= 5	Evalue l'expression 3 <= 5 = vrai	3	3			

11	sum ← sum + i	Evalue l'expression sum + i = 3 + 3 = 6 et affecte la valeur à la variable sum	6	3			
12	(pour) i← i+1	Evalue l'expression i+1 = 3+1 = 4 et l'affecte la valeur à la variable i	6	4			
13	(pour) i <= 5	Evalue l'expression 4 <= 5 = vrai	6	4			
14	sum ← sum + i	Evalue l'expression sum + i = 6 + 4 = 10 et affecte la valeur à la variable sum	10	4			
15	(pour) i← i+1	Evalue l'expression i+1 = 4+1 = 5 et l'affecte la valeur à la variable i	10	5			
16	(pour) i <= 5	Evalue l'expression 5 <= 5 = vrai	10	5			
17	sum ← sum + i	Evalue l'expression sum + i = 10 + 5 = 10 et affecte la valeur à la variable sum	15	5			
18	(pour) i← i+1	Evalue l'expression i+1 = 5+1 = 6 et l'affecte la valeur à la variable i	15	6			
19	(pour) i <= 5	Evalue l'expression 6 <= 5 = vrai	15	6			
20	Ecrire(sum)	Evalue l'expression sum = 15 et affiche la valeur	15	-			

Repérez comment on a représenté la répétition des instructions, l'incrémentation de la variable de boucle et le test de la condition de continuation dans la trace.

Exercice 3 : Faites la trace d'exécution du programme Exercice3 et expliquez ce que fait le programme. La fonction ord() permet d'obtenir le code ACSII d' un caractère.

```
Algorithme Exercice3
Variable i, somme: entire;
Letter: Tableau [26] caractère;
Debut
  letter [26] ← {'a','b','c','d','e','f','g','h','i','j',
                 'k','l','m','n','o','p','q','r','s','t',
                 P'u','v','w','x','y','z'}

        pour (i ← 0 à 25 ) faire
            somme ←somme + ord(lettre[i]) ;
        Finpour

        Écrire(somme);
Fin
```