Sommaire

Série 2 : Les Sous-Programmes : Procédure & Fonctions

Exercice 1 : Passage de Paramètres

Exécuter le programme suivant :

```
01 Program Exo_1;
02 Uses wincrt ;
03 var
04    a, b, c : integer;{Variables Globales du programme}
05
06 Procedure Proc1(x:integer ; y:integer ; s:integer) ;
07  Begin
08    s := x+y ;
09  End;
10
11 Procedure Proc2(x:integer ; y:integer ; var s:integer);
12  Begin
13    s := x+y;
14  End;
15
16 BEGIN {Début du Programme Principal}
17    a:=10; b:=5; c:=0;
18    Proc1(a, b, c);
19    Writeln('La somme est : ', c);
20
21    a:=10; b:=5; c:=0;
22    Proc2(a, b, c);
23    Writeln('La somme est : ', c);
24 End. {Fin du Programme Principal}
```

La partie déclaration
On a déclaré trois variables globales et deux procédures

Le Corps du Programme
La partie instructions

L'appel à la procédure Proc1, en transmettant les paramètres : a, b et c

L'appel à la procédure Proc2, en transmettant les paramètres : a, b et c

- C'est quoi la différence entre les deux procédures *Proc1* et *Proc2* ?

- Quels sont les paramètres à passage par valeur et ceux à passage par variable ?

- Quels sont les paramètres formels des deux procédures ?

- Et quels sont les paramètres effectifs ?

Solution

1- La différence entre les deux procédures *Proc1* et *Proc2* réside dans le troisième paramètre **s** . Dans Proc2, le paramètre **s** est défini en utilisant le mot clé **var**. Par contre, dans Proc1 il est défini sans utilisation du mot clé **var**. Ceci aura un effet dans l'exécution des deux procédures, c'est quoi cette effet ? (voir les autres réponses)

2- Les paramètres à passage par valeur et ceux à passage par variable :

Pour la procédure Proc1 :

- Passage par Valeur : les paramètres x, y et s
- Passage par Variable : aucun paramètre

Pour la procédure Proc2 :

- Passage par Valeur : les paramètres x et y
- Passage par Variable : les paramètres s

La définition de type de passage de paramètres (par valeur ou par variable) se fait dans la partie déclaration, où on déclare les procédures ou les fonctions. Dans le programme précédent, la partie déclaration se situe entre les lignes *03* et *14*.

3, 4- Les paramètre formels et les paramètres effectifs

Pour la procédure Proc1 :

- Paramètres formels : x, y et s
- Paramètres effectifs : a, b et c (dans l'appel de la ligne n° 18)

Pour la procédure Proc2 :

- Paramètres formels : x, y et s
- Paramètres effectifs : a, b et c (dans l'appel de la ligne n° 22)

Les paramètres formels sont les paramètres utilisés pour déclaré un sous-programme (procédure ou fonction). Donc pour chercher les paramètres formels d'un sous programme, il faut voir dans la partie déclaration.

Et pour les paramètres effectifs, il faut regarder dans la partie du corps du programme (partie instruction), pour d'éventuels appels à un sous programme. Pour chaque appel d'une procédure ou d'une fonction, on doit indiquer les paramètres effectifs. Par exemple, dans la ligne n°22, on a l'instruction : ***Proc2(a, b, c) ;*** ce qui signifie, l'appel à la procédure Proc2 avec les paramètres effectifs a, b et c. Dans ce cas là, la valeur de a sera transmise au paramètre formel x, celle de b sera transmise au paramètre formel y et la valeur de c sera transmise à s.

5- Déroulement du programme

Dans cette exercice (et dans cette série), nous avons aussi des sous-programmes définis au sein du programme principal. Donc, il faut séparer entre les variables globales (variable du programme principal) et les variables locales de chaque sous-programme (fonction ou procédure), comme illustré dans le tableau suivant :

Instructions	***Programme Principal***			***Procédure Proc1***			***Procédure Proc2***		
	a	*b*	*c*	*x*	*y*	*s*	*x*	*y*	*s* ***(var)***
a:=10; b:=5; c:=0;	10	5	0						
Proc1(a, b,c) (L'appel à Proc1)	"	"	"						
=> La trasmission des paramètres s:=x+y				10	5	0 **15**			
write(c) (la valeur de **s** dans **Proc1** n'a pas été retournée à la variable globale **c**)	"	"	**0**						
a:=10; b:=5; c:=0;									
Proc2(a, b,c) (L'appel à Proc2)									
=> La trasmission des paramètres s:=x+y							10	5	0 **15**
write(c) (la valeur de **s** dans **Proc2** a été retournée à la variable globale **c**)	"	"	**15**						

Remarques :

1. Dans le tableau précédant, les zones gris signifie que les **sous-programmes** ne sont pas en exécution et les zone jaune signifie que le **programme principal** a fait l'appel à un sous-programme et attends le retour d'appel (la fin du sous-programme)
2. L'appel à la procédure Proc1 – Proc1(a, b, c) – permet de transmettre les valeurs des paramètres effectif a, b et c vers les paramètres formel x, y et s.
3. L'appel à la procédure Proc2 – Proc2(a, b, c) – permet de transmettre les valeurs des paramètres effectif a, b et c vers les paramètres formel x, y et s.
4. La valeur du paramètre formel s de la procédure Proc1 n'a pas été retourné au paramètre effectif c (la variable globale c). (**La transmission par valeur**).
5. La valeur du paramètre formel s de la procédure Proc2 a été retourné au paramètre effectif c (la variable globale c). (**La transmission par variable**).
6. La transmission (le passage) des paramètres par valeur permet de transmettre les valeurs des paramètres effectif vers les paramètres formel (Paramètres d'entrée)
7. La transmission (le passage) des paramètres par variable permet de transmettre les valeurs des paramètres effectif vers les paramètres formel et, à la fin de l'exécution du sous-programme, de retourner les valeurs des paramètres formel vers les paramètre effectifs correspondant (Paramètres d'entrée et de sortie)

Exercice 2 : Transformation Fonction - Procédure

```
01 Program CombinaisonNK;
02 Uses wincrt ;
03 var
04     n, k, c : integer; {Variables Globales du programme}
05
06 Function fact(n:integer):integer ;
07     Var
08           f, i : integer ; {Variables locales de la fonction fact}
09  Begin
10     F:=1;
11     For i:=1 to n do
12           f:=f*i;
13
14     fact:=f; {Une fonction se termine toujours par une affectation}
15  End;
16
17 BEGIN {Début du Programme Principal}
18     Writeln('Donnez la valeur de n et k :');
19     Read(n, k);
20
21     C:= fact(n) div ( fact(k) * fact(n-k) );
22
23     Writeln('La combinaison de k à partir de n = ', c);
24 End. {Fin du Programme Principal}
```

- Dérouler le programme ci-dessus pour ***n = 5*** et ***k = 3***
- Réécrire le programme en transformant le fonction ***fact*** à une procédure ***fact***.

Soit la procédure ***puissance*** définie comme suit :

```
01 Procedure puissance(x:real; n:integer ; var p:real);
02     Var
03           i : integer ; {Variables locales de la procédure puissance}
04  Begin
05     P:=1;
06     For i:=1 to n do
07           p:=p*x;
08  End;
```

- Transformer cette procédure à une fonction.

Solution

1- Le déroulement du programme

Instructions	***Programme Principal***			***La fonction fact***			
	n	*k*	*c*	*n*	*i*	*f*	*fact*
Read(n, k)	5	3	/				
*C:=fact(n) div (fact(k) * fact(n-k))* Trois appel à la fonction fact. En appliquant l'ordre de Priorité sur l'expression arithmétique : 1- n-k=2 2- fact(k) 3- fact(2) 4- *(fact(k) * fact(n-k)* 5- *fact(n)* 6- *div*	5	3	/				
fact(k) (l'appel à fact avec le paramètre k=3)	5	3					
=> La trasmission des paramètres *f:=1* *for i=1 f:=f*i → f=1*1 = 1* *for i=2 f:=f*i → f=1*2 = 2* *for i=3 f:=f*i → f=2*3 = 6* *fact := f → **fact = 6***				3	/ 1 2 3	/ 1 1 2 6	/ **6**
fact(n-k) (l'appel à fact avec le paramètre n-k=2)							
=> La trasmission des paramètres *f:=1* *for i=1 f:=f*i → f=1*1 = 1* *for i=2 f:=f*i → f=1*2 = 2* *fact := f → **fact = 2***				2	/ 1 2	/ 1 1 2	/ **2**
fact(n) (l'appel à fact avec le paramètre n=5)	5	3					
=> La trasmission des paramètres *f:=1* *for i=1 f:=f*i → f=01*1 = 1* *for i=2 f:=f*i → f=01*2 = 2* *for i=3 f:=f*i → f=02*3 = 6* *for i=4 f:=f*i → f=06*4 = 24* *for i=5 f:=f*i → f=24*5 = 120* *fact := f → **fact = 120***				5	/ 1 2 3 4 5	/ 1 1 2 6 24 120	/ **120**
*C:= 120 div (6 * 2) → C = 120 div 12 = 10* *write (c)*			**10** **10**				

2- Réécriture du programme en transformant la fonction fact à une procédure

```
01 Program CombinaisonNK;
02 Uses wincrt ;
03 var
04     n, k, c : integer; {Variables Globales du programme}
05
06 Function fact(n:integer):integer ;
07     Var
08           f, i : integer ; {Variables locales de la fonction fact}
09  Begin
10     F:=1;
11     For i:=1 to n do
12           f:=f*i;
13
14     fact:=f; {Une fonction se termine toujours par une affectation}
15  End;
16
17 BEGIN {Début du Programme Principal}
18     Writeln('Donnez la valeur de n et k :');
19     Read(n, k);
20
21     C:= fact(n) div ( fact(k) * fact(n-k) );
22
23     Writeln('La combinaison de k à partir de n = ', c);
24 End. {Fin du Programme Principal}
```