Exercice N°02 :

Soit le programme PASCAL suivant :

```
Program  Exo_2;
Var i, n: integer;
 S: real;  {Variables globales du programme}
FUNCTION  Fact (m: integer): integer;
Var j, f : integer; {Variables locales de la fonction Fact}
Begin
    f:=1;
    for j:=1 to m do
      f:= f*j;
    Fact := f;   {Une fonction se termine toujours
                  par une affectation du résultat}
End;
BEGIN  {Début du programme principal}
   write('Introduire n : ');
   read(n);
   S:=1;
   for i:=1 to n do
    S:=S+Fact(i);
   write('La somme = ',S:6:2);
END.  {Fin du programme principal}
```

1) Exécuter le programme pour n = 3.
2) Dérouler le programme pour n = 3.
3) Réécrire le programme en transformant la fonction Fact en une procédure Fact.
4) Soit la procédure « Puiss » suivante :

```
Procedure Puiss (t: real; k: integer; var P: real);
Var i: integer;
Begin
   P:=1;
   for i:=1 to k do
     P:=P*t;
End;
```

4-a) Dérouler la procédure Puiss pour t=2 et k=4 et déduire ce qu'elle fait.

4-b) Transformer la procédure Puiss en une fonction.

5) On propose de calculer la valeur approximative de l'exponentiel de x avec la formule suivante :

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

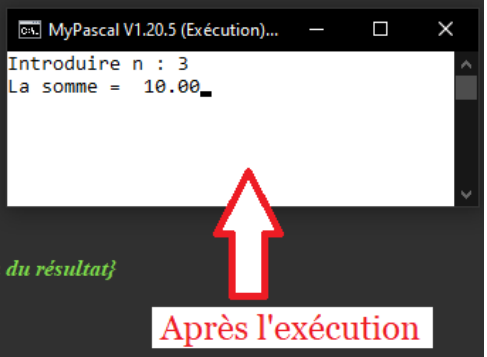
En utilisant les deux fonctions Fact et Puiss, écrire un programme Pascal pour calculer e^x .

Solution :

1) Exécution du programme pour n = 3.

exo2_tp2.pas

```
1  Program exo2;
2  Var i, n: integer;
3   S: real; {Variables globales du programme}
4  FUNCTION Fact (m: integer): integer;
5  var j, f: integer; {Variables locales de la fonction Fact}
6  begin
7      f:=1;
8      For j:=1 to m do
9       f:= f*j;
10     Fact := f; {Une fonction se termine toujours par une affectation du résultat}
11 end;
12 BEGIN {Début du programme principal}
13     write('Introduire n : ');
14     read(n);
15     S:=1;
16     for i:=1 to n do
17      S:=S+Fact(i);
18     write('La somme = ',S:6:2);
19 END. {Fin du programme principal}
```



2) Dérouler du programme pour n = 3

Instructions	Programme principal			La fonction Fact				Affichage
	n	i	S	m	j	f	Fact	
write('Introduire n : ');								Introduire n :
Read(n)	3							
S :=1			1					
For i :=1 S :=S+Fact(i)		1	1+?					
Fact(i) *(l'appel à Fact avec le paramètre i=1)*		1						
=> La transmission des paramètres f:=1 for j:=1 f:=f*i → f=1*1 = 1 Fact := f → fact = 1 *=> Le retour du résultat dans le prog principal*			1+**1**=**2**	**1**	1 1	1 1	**1**	
For i :=2 S :=S+Fact(i)		2	2+?					
Fact(i) *(l'appel à Fact avec le paramètre i=2)*		2						
=> La transmission des paramètres f:=1 for j:=1 f:=f*i → f=1*1 = 1 for j:=2 f:=f*i → f=1*2 = 2 Fact := f → fact = 2 *=> Le retour du résultat dans le prog principal*			2+**2**=**4**	**2**	1 2	1 1 2	**2**	
For i :=3 S :=S+Fact(i)		3	4+?					
Fact(i) *(l'appel à Fact avec le paramètre i=3)*		3						
=> La transmission des paramètres f:=1 for j:=1 f:=f*i → f=1*1 = 1 for j:=2 f:=f*i → f=1*2 = 2 for j:=3 f:=f*i → f=1*2*3 = 6 Fact := f → fact = 6 *=> Le retour du résultat dans le prog principal*			4+**6**=**10**	**3**	1 2 3	1 1 2 6	**6**	
write('La somme = ',S:6:2);								La somme =10.00

3) Réécrire le programme en transformant la fonction Fact en une procédure Fact.

Rappelons les étapes à suivre lors de la transformation de fonction vers une procédure :

a) Modifier le mot clé *Function* vers le mot clé *Procedure* ;
b) Supprimer le type de retour de la fonction lors du passage vers une procédure ;
c) Ajouter un paramètre résultat dans la déclaration de l'entête de la procédure Fact. Il s'agit d'identifier la variable locale de la fonction représentant le résultat retourné par la fonction (on le trouve facilement dans l'instruction d'affectation du résultat de la fonction) ;
d) Supprimer la variable locale représentant le résultat renvoyé dans la déclaration de variables locales.
e) Supprimer l'instruction de retour du résultat dans la fonction (c-à-d l'instruction qui commence par l'identificateur de la fonction), généralement elle se trouve à la fin du corps de la fonction.

En appliquant ces étapes, nous obtenons la procédure suivante :

```
Procedure  Fact (m: integer; VAR f: integer);
var j: integer; {Variables locales de la procédure Fact}
begin
     f:=1;
     For j:=1 to m do
       f:= f*j;
end;
```

Pour insérer maintenant cette procédure dans le programme, trois étapes sont nécessaires :

a) Ajouter dans le bloc de déclaration une variable globale pour chaque appel de la procédure. Dans cet exercice, on a un seul appel de la procédure Fact qui sera utilisé dans le programme principal. Par conséquent, nous allons déclarer une variable « Fc », de même type que le paramètre de sortie « f » et qui sera utilisée lors de passage de paramètre par variable de la procédure Fact.
b) Dans le programme principal, nous allons transformer l'appel de fonction Fact(i) en appel de procédure en rajoutant au premier paramètre d'entrée « i » un paramètre de sortie « Fc ». Cependant, on rappel une différence fondamentale entre l'appel d'une fonction, qui donne toujours une valeur, et l'appel d'une procédure qui représente une instruction à part entière. On obtient donc l'instruction suivante : Fact(i, Fc). La variable « Fc » représente le résultat retourné par la procédure Fact.
c) En dernière étape, nous allons remplacer, dans l'instruction qui calcule S, l'appel de la fonction Fac(i) par la variable « Fc ».

En appliquant ces étapes, nous obtenons le programme suivant :

```
Program  exo2;
Var i, n, Fc: integer; {Ajout de la variable globale Fc qui remplacera le paramètre formel f lors de l'appel de la procédure Fact}
 S: real;
Procedure  Fact (m: integer; VAR f: integer); {Ajout du paramètre formel f en utilisant le passage par Variable}
var j: integer; {Suppression de la variable locale f}
begin
      f:=1;
      For j:=1 to m do
        f:= f*j;
end;
BEGIN  {Début du programme principal}
     write('Introduire n : ');
     read(n);
     S:=1;
     for i:=1 to n do
     begin  {Ajout de Begin et End car on deux instructions dans la boucle For}
       Fact (i, Fc);   {L'appel de la procédure Fact est une instruction, Fc est la variable résultat}
       S:=S+Fc;  {On remplace l'appel de la fonction Fact(i) par la variable résultat Fc}
     end;
     write('La somme = ',S:6:2);
END.  {Fin du programme principal}
```

3-a) Dérouler la procédure Puiss pour t=2 et k=4 et déduire ce qu'elle fait.

```
Procedure Puiss (t: real; k: integer; var P: real);
Var i: integer;
Begin
P:=1;
For i:=1 to k do
   P:=P*t;
End;
```

Instruction	Variables de la procédure Puiss			
	t	k	i	P
{passage de paramètres par valeur}	2	4	-	-
P:=1;				1
For i:=1			1	
P:=P*t;				1*2=2
For i:=2			2	
P:=P*t;				$2*2=2^2$
For i:=3				
P:=P*t;			3	
For i:=4				$2^2*2=2^3$
P:=P*t;			4	
				$2^{3*}2=2^4$

D'après le déroulement, notamment dans la relation entre l'expression finale de P et les paramètres d'entrées *t* et *k*, on déduit que la procédure **Puiss** permet de calculer la puissance d'un nombre *t* par *k*, c'est-à-dire : $\mathbf{t^k}$.

3-b) Transformer la procédure Puiss en une fonction.

Il s'agit d'appliquer les étapes de la transformation de la fonction vers la procédure (réponse à la question 2) dans le sens inverse. Voici donc les étapes à suivre :

1) Modifier le mot clé « *Procedure* » par le mot clé « *Function* » ;
2) Ajouter le type de retour de la fonction « Puiss » avec le même type que celui du paramètre résultat P présent dans l'entête de la procédure Puiss ;
3) Supprimer le paramètre résultat P dans la déclaration de l'entête de la fonction Puiss ;
4) Ajouter la variable P dans le bloc de déclaration de variables locales de la fonction « Puiss » ;
5) Ajouter l'instruction de retour du résultat à la fin du corps de la fonction Puiss, c'est-à-dire l'instruction : Puiss :=P ;

En appliquant ces étapes, nous obtenons la procédure suivante :

```
Function Puiss (t: real; k: integer) : real; {Suppression du paramètre résultat P et l'ajout de type du résultat renvoyé}
Var i: integer;
      P:real; {Ajout de la variable locale P}
Begin
P:=1;
For i:=1 to k do
  P:=P*t;
Puiss := P; {Ajout l'instruction de retour du résultat P dans la fonction Puiss}
End;
```

4) On propose de calculer la valeur approximative de l'exponentiel de x avec la formule suivante :

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

En utilisant les deux fonctions Fact et Puiss, écrire un programme Pascal pour calculer e^x .

```
Program  Calcul_Exponentiel;
Var i, n: integer; {Variables globales du programme principal}
      X, S: real;
FUNCTION  Fact (m: integer): integer;
var j, f: integer; {Variables locales de la fonction Fact}
begin
     f:=1;
     For j:=1 to m do
       f:= f*j;
     Fact := f;
end;
FUNCTION Puiss (t: real; k: integer) : real;
Var i: integer; {Variables locales de la fonction Puiss}
      P:real;
Begin
P:=1;
For i:=1 to k do
  P:=P*t;
Puiss := P;
End;

BEGIN  {Début du programme principal}
    write('Introduire les valeurs de n et x : ');
    read(n, x);
    S:=1;
    for i:=1 to n do
     S:=S+ Puiss(x, i) / Fact(i) ;  {Calcul de S en appelant les deux fonctions Fact et Puiss}

    write('La valeur approximative de Exp (x) = ', S:6:2);
END.  {Fin du programme principal}
```