LEÇON 4 : La Récursivité

UVCI

Table des matières

1- Généralité sur la récursivité

I

Définition : 1.1- Définition

La *récursivité* est une méthode de description d'algorithmes qui permet à une procédure ou fonction de s'appeler elle-même.

Chaque appel à la procédure ou de la fonction est indépendant des autres, avec ses propres variables.

Remarque : 1.2- Remarque

Dans une procédure ou fonction récursive, il y a deux notions à retenir :

- *la procédure ou la fonction s'appelle elle-même : on recommence avec de nouvelles données.*
- *il y a un test de fin : dans ce cas, il n'y a pas d'appel récursif. Il est souvent préférable d'indiquer le test de fin des appels récursifs en début de procédure ou fonction.*

Application 1 :

II

Exercice

La récursivité est :

- Un ensemble de fonctions qui communiquent ensemble
- Un sous-algorithme qui s'appelle lui même
- Un ensemble de fonctions et de procédures qui communiquent ensemble

2- Utilisation de la récursivité

III

2.1- Exemple utilisation de récursivité simple

Énoncé :

Écrire un algorithme récursif qui permet de calculer la factorielle d'un nombre N.

Solution :

fonction factorielle({E} n : entier) : entier *// entête de la fonction factorielle avec un paramètre en entrée, le résultat de la fonction est un entier.*

debut

si n=0 alors *//cette condition permet de mettre fin à la boucle de la fonction (lorsque n = 0)*

renvoyer 1 *// lorsque n est zéro le résultat retourné est 1*

sinon

renvoyer n* factorielle(n-1) *// faire la récursivité de la fonction cette partie sera détaillée dans un tableau ci-dessous.*

finsi

finfonc

*//********** L'algorithme principal **********

algorithme recursivite1

fonction factorielle({E} n : entier) : entier *//prototype de la fonction*

var

nbre,fact : entier *// variable globale*

debut

afficher "Donner le nombre"

saisir nbre

fact ← factorielle(nbre) *//récupère le résultat de la fonction*

afficher "La factorielle de ",nbre," = ", fact *// affiche le résultat*

fin

Nous supposons que nbre = 5 on aura :

n	fonction sans calcul de paramètre	fonction avec calcul de paramètre	opération de la fonction	valeur retournée
5	5*factorielle(5-1)	5*factorielle(4)	5*24	120
4	4*factorielle(4-1)	4*factorielle(3)	4*6	24
3	3*factorielle(3-1)	3*factorielle(2)	3*2	6
2	2*factorielle(2-1)	2*factorielle(1)	2*1	2
1	1*factorielle(1-1)	1*factorielle(0)	1*1	1
0	factorielle(0)	1	1	1

Explication :

Pour calculer factorielle(5) sur le tableau ci-dessus , il faut calculer factorielle(4).

Pour calculer factorielle(4), il faut calculer factorielle(3).

Pour calculer factorielle(3), il faut calculer factorielle(2).

Pour calculer factorielle(2), il faut calculer factorielle(1).

Pour calculer factorielle(1), il faut connaître factorielle(0). factorielle(0) vaut 1.

2.1- Exemple utilisation de récursivité multiple

Énoncé :

Écrire un algorithme récursif pour calculer le nième terme de la suite de Fibonacci.

NB : La suite de Fibonacci est une suite d'entiers dans laquelle chaque terme est la somme des deux termes qui le précèdent. Elle commence généralement par les termes 0 et 1 (parfois 1 et 1) et ses premiers termes sont : 0, 1, 1, 2, 3, 5, 8, 13, 21, etc.

Solution :

La suite des nombres de Fibonacci se définit comme suit : f0= 0 ; f1= 1 ; fn =fn-1 + fn-2 si n > 1

fonction Fibonacci({E}n : entier) : entier

debut

si n <=1 alors *//cette condition permet de mettre fin à la boucle de la fonction (lorsque n <= 1)*

renvoyer n *// La valeur de n*

sinon

renvoyer Fibonacci(n-1)+ Fibonacci(n-2) *// faire la récursivité de la fonction cette partie sera détaillée dans un tableau ci-dessous.*

finsi

finfonc

*//********** L'algorithme principal **********

```
algorithme recursivite2
fonction Fibonacci({E}n : entier) : entier //prototype de la fonction
var
nbre,fib : entier // variable globale
debut
afficher "Donner le nombre"
saisir nbre
fib ← Fibonacci(nbre) //récupère le résultat de la fonction
afficher "La Fibonacci de ",nbre," = ", fib // affiche le résultat
fin
```

Nous supposons que nbre = 6 on aura :

n	fonction sans calcul de paramètre	fonction avec calcul de paramètre	opération de la fonction	valeur retournée
5	Fibonacci(5-1)	Fibonacci(4)	3+2	5
4	Fibonacci(4-1)	Fibonacci(3)	2+1	3
3	Fibonacci(3-1)	Fibonacci(2)	1+1	2
2	Fibonacci(2-1)	Fibonacci(1)	1	1
1	Fibonacci(1-1)	1	1	1

cas : Fibonacci(n-1) => Fibonacci(6-1) = Fibonacci(5)

n	fonction sans calcul de paramètre	fonction avec calcul de paramètre	opération de la fonction	valeur retournée
4	Fibonacci(4-1)	Fibonacci(3)	2+1	3
3	Fibonacci(3-1)	Fibonacci(2)	1+1	2
2	Fibonacci(2-1)	Fibonacci(1)	1	1
1	Fibonacci(1-1)	1	1	1

cas : Fibonacci(n-2) => Fibonacci(6-2) = Fibonacci(4)

Du coup à la fin nous aurons : *Fibonacci(6) = Fibonacci(5) + Fibonacci(4) = 5+3 = 8*

Application 2 :

IV

Exercice

Partie 1 :

Écrire un algorithme récursif pour calculer la puissance d'un nombre.

En respectant les instructions,précédent chaque trou, veuillez ajouter les valeurs manquantes

NB : Toutes les réponses doivent être en minuscule et sans espace

puissance(nbre, exp : entier) : entier

début

si exp = 0 alors

renvoyer 1

sinon

renvoyer nbre*puissance(nbre,)

finsi

fin

Algorithme calpuissance

puissance(nbre, exp : entier) : entier

var

val1,val2,resu :entier

début

val1 ← 15

resu ← 0

val2← 3

resu ← (val1,val2)

afficher "Le resultat est ",

fin

exp	nbre	fonction sans calcul de paramètre	fonction avec calcul de paramètre	valeur retournée	resu	val1
				15*1=15		

Veuillez donner les différentes valeurs et expressions des éléments du tableau ci-dessous à chaque instruction exécutée ci-dessus. Dans notre exercice il n'y a trois(3) itérations