Leçon N° 1 :

L'analyse Modulaire

Diviser chacune des difficultés que j'examinerais en autant de parcelles qu'il se pourrait et qu'il serait requis pour les mieux résoudre
DESCARTES

INTRODUCTION

- Exemple 1 : la réalisation d'un projet (travail de groupe,…)
- Exemple 2 : la rédaction de la nouvelle Constitution (travaux des commissions)

➔ « Diviser pour régner » ➔

1. Définition :

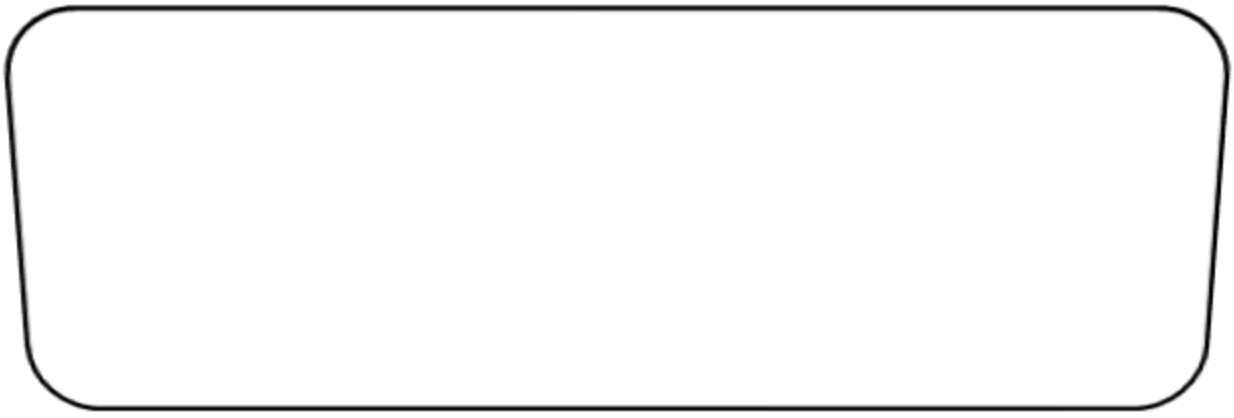
L'analyse modulaire consiste à diviser un problème en sous problèmes de difficultés moindres. Ces derniers peuvent être aussi assujettis à cette division jusqu'à ce qu'on arrive à un niveau abordable de difficulté.

2. Intérêts :

- _____
- _____
- _____
- _____

Activité N°1

Ecrire une analyse, un algorithme et un programme pascal qui permet de saisir deux entiers n et p avec (5 < p< n< 12), calculer le PGCD et PPCM de n et p, puis afficher la combinaison de C_n^p , enfin remplir le tableau T par les facteurs premier n*p et l'afficher.



Notions de sous-programme (voir page 147)

Un sous – programme peut être une procédure ou une fonction

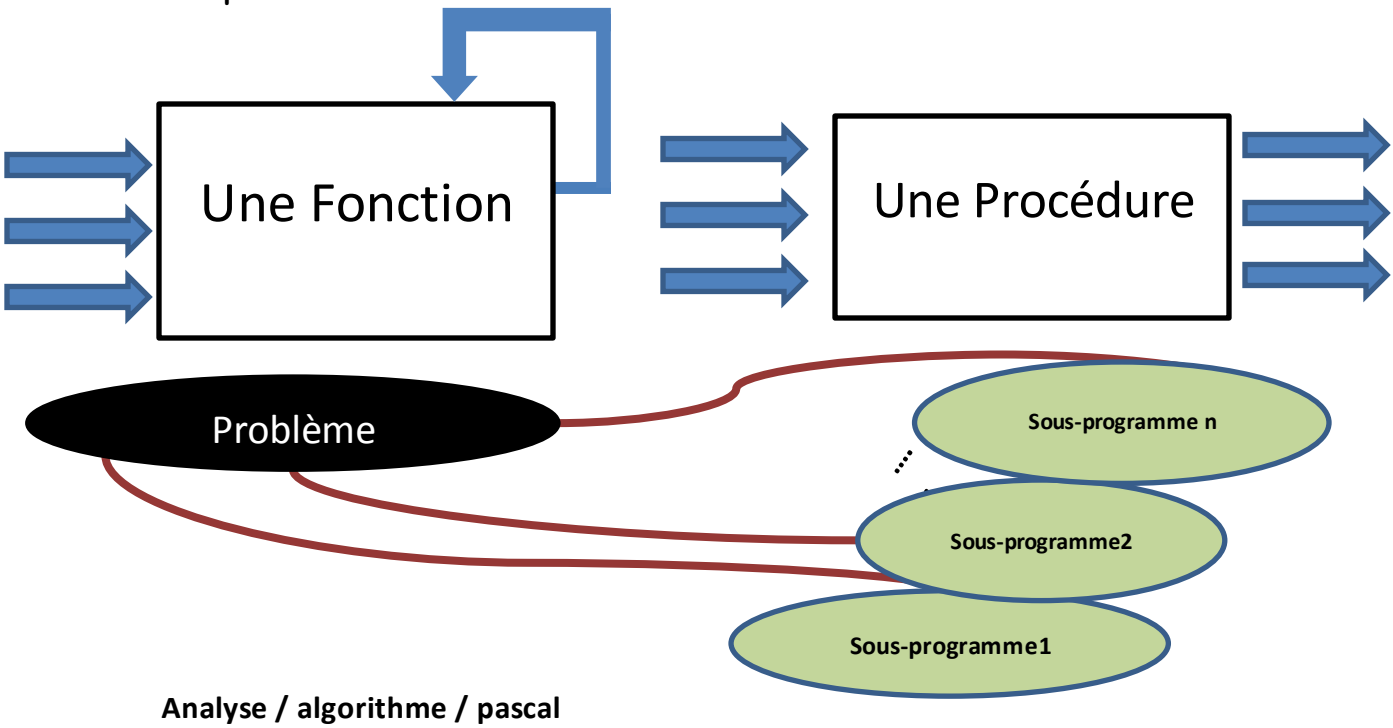
➔ Pour mieux assimiler la différence entre les procédures et les fonctions, compléter le tableau comparatif pour les procédures et les fonctions prédéfinies chaines de caractères.

	Une Fonction	Une Procédure
Nombre de résultat retourné		
L'Appel		
Changement des paramètres		

Un sous-programme ➔un Sous-problème ➔un Module (une procédure ou une fonction)

N.B :
- toute fonction peut être transformée en une procédure
- Une fonction présente un cas particulier de la procédure

Procédure ou fonction ??
- Nombre de résultats ?
- Type de résultats (simples ou structurés) ?
- Modifier les paramétrés d'entrée ?



Analyse / algorithme / pascal

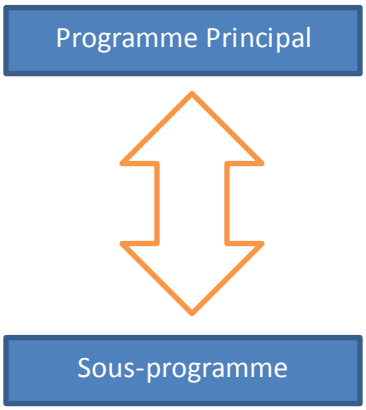
Leçon N° 2:

Les Procédures

Activité N°1

Ecrire une analyse, un algorithme et un programme pascal qui permet de saisir un entier n avec ($4 < n < 13$), remplir un tableau **T** par n entiers de 4 chiffres, puis éliminer les chiffres impairs de chaque l'élément de **T**, enfin afficher le tableau **T**

Décomposer ce programme en modules



L'algorithme de Programme Principal

_____

_____

_____

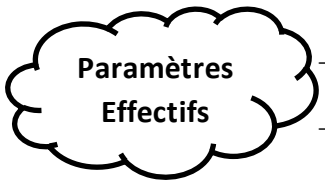
_____

_____

_____

_____

_____



L'analyse de Programme Principal

_____

_____

_____

_____

_____

_____

_____

_____

T.D.O.G (_____)

Objet	Type/Nature	Rôle

1. Définition
2.Syntaxe et vocabulaire

Au niveau Analyse :

DEF Proc nom_Proc (paramètres formels)
Resultat =

Fin Nom_proc

Au niveau Algorithme :

0) DEF Proc nom_Proc (paramètres formels)
1)
2)

n) Fin Nom_proc

Au niveau Pascal :

Uses wincrt ;
const
type
var
. . . . . .

Procedure nom_proc (paramètres formels) ;
type
var
. . . . . .
begin
. . . . . .

nom_proc (paramètres effectifs)

end ;

Appel de procédure

Analyse de la procédure Saisie

L'algorithme de la procédure saisie

Analyse de la procédure remplir

TDOL

Objet	Type	Rôle

L'algorithme de la procédure remplir

Analyse de la procédure éliminer

L'algorithme de la procédure éliminer

T.D.O.L (_____)

Objet	Type/Nature	Rôle

Analyse de la procédure Afficher

Algorithme de la procédure Afficher

Programme Pascal :

```
Uses wincrt ;
Type
Tab=array[1..12] of integer ;
Var
n :integer ;
t :tab ;
Procedure Saisie( var n :integer) ;
Begin
   Repeat
      Writeln('donner n') ;
      Readln(n) ;
   Until n in [5..12] ;
End ;
Procedure remplir ( n :integer ;var t :tab) ;
Var
I :integer ;
Begin
     For i :=1 to n do
       Repeat
          Writeln('T[ ',i,'] : ') ;
          Readln(t[i]) ;
       Until (t[i] > 999)and (t[i] <=9999) ;
End ;
procedure eliminer (n :integer ; var t :tab) ;
Var
I ,x,e,j :integer ;
ch,ch1 :string ;
Begin
   For i :=1 to n do
     Begin
     Str(t[i],ch) ;
     ch1 :=" ;
     for j :=1 to length(ch) do
        Begin
          If ch[j] in ['0','2','4','6','8'] then
                                  ch1 :=ch1+ch[j] ;
        End ;
     Val(ch1,x,e) ;
     t[i] :=x ;
     End ;
End ;
procedure afficher(n :integer ;t :tab) ;
Var i :integer ;
Begin
      For i :=1 to n do
       Write(t[i],' | ') ;
End ;

begin

     Saisie(n) ;
     Remplir(n,t) ;
     writeln('afficher T avant la Suppression') ;
     Afficher(n,t) ;
     Writeln ;
      Eliminer(n,t) ;
      writeln ;
      writeln('afficher T après la Suppression') ;
      Afficher(n,t) ;

End.
```

3. Déclaration et accès aux objets

a. Les objets globaux :

Un objet Global est déclaré dans la partie déclarative du programme principal. Cet objet est utilisable par le PP et par les différents autres sous-programmes.
Exemples :

b. Les objets locaux :

Un objet Local est un objet déclaré et connu seulement à l'intérieur d'un sous- programme.
Exemples :

c. Niveaux des sous-programmes :(Visibilité des variables)

On peut organiser les appels des sous-programmes en une arborescence hiérarchique. Cette arborescence s'étend de la racine (Identificateur du PP) jusqu'au dernier niveau de nœuds (sous-programme) imbriqués.

Un objet déclaré dans un sous-programme S à un niveau i est inaccessible par les sous-programmes de niveau j < i et par les sous-programmes de niveau k > i qui ne sont pas englobés par S.

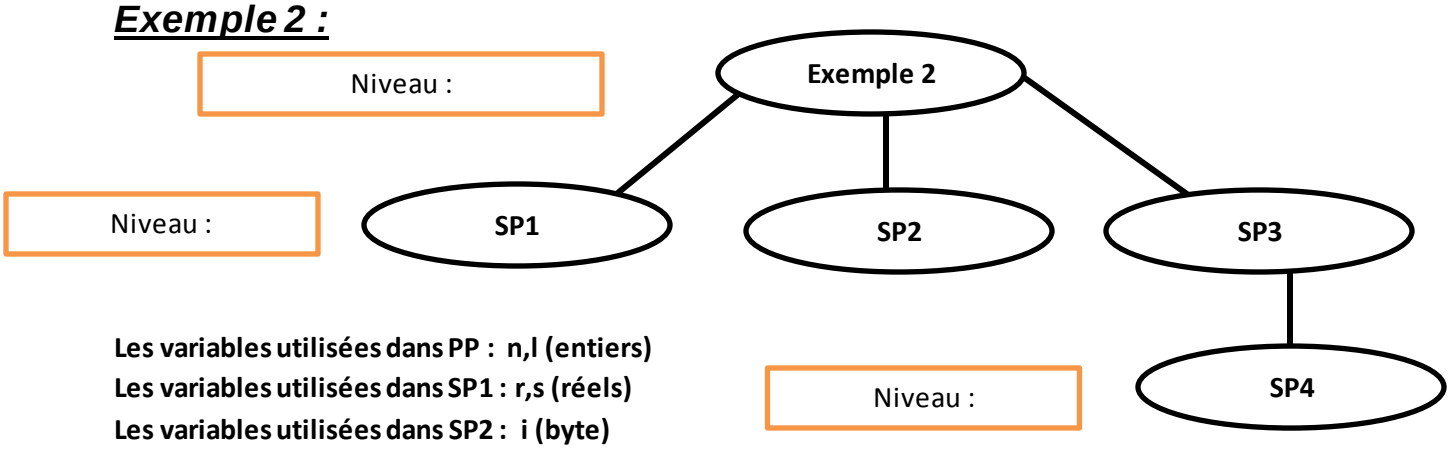
☞ **Pour mieux assimiler**

Exemple 1 :

compléter le tableau suivant

Les variables	Nature (local/global)
n	
i	
X,e	
t	

Exemple 2 :



Les variables utilisées dans PP : n,l (entiers)
Les variables utilisées dans SP1 : r,s (réels)
Les variables utilisées dans SP2 : i (byte)
Les variables utilisées dans SP3 x,s (char)
Les variables utilisées dans SP4 : t,w (chaines)

Etablissons à partir de la règle de visibilité, deux tableaux récapitulatifs croisés de la visibilité des variables : n,l,r,s,i,x,s,t,w

Variable	Module où cette variable est visible
n	
l	
r	
s	
i	
x	
s	
t	

```
Uses wincrt ;
var . . . . . .
PRocedure SP1(  )
var
. . . . . .
begin

end ;
PRocedure SP2(….)
var
. . . . . .
begin

end ;
PRocedure SP3(  )
var . . . . . .
PRocedure SP4(  )
var . . . . . .
begin

end ;

begin

end ;
```

Begin … **End.** — PP

Modules	Les variables visibles dans ce bloc
PP	
SP1	
SP2	
SP3	
SP4	

4 - Les paramètres et leurs modes de transmission :

a. Les paramètres formels :

Les paramètres formels figurent dans l'entête de la définition d'un sous-programme.
exemple :
Pour leurs (analyse et algorithme) il faut indiquer le nom, le type et le mode de passage de chaque paramètre (voir ci-dessous)

b. les paramètres effectifs :

Les paramètres effectifs figurent dans l'appel d'un sous-programme.
Exemple :

N.B

★ **Les paramètres effectifs & les paramètres formels doivent s'accorder du point de vue *nombre* et *ordre*.**

★ **Leurs types doivent être *identiques* ou *compatibles***

★ **Leurs identificateurs peuvent être différents.**

☞ **Pour mieux assimiler (bac 2010)**

On suppose qu'un programme principal contient trois sous programmes (procédure P1, procédure P2 et Fonction F3). Compléter le tableau suivant par un exemple d'appel de chacun sous programmes au niveau de programme principal. En se basant sur les entêtes et sur la liste des variables globales disponibles

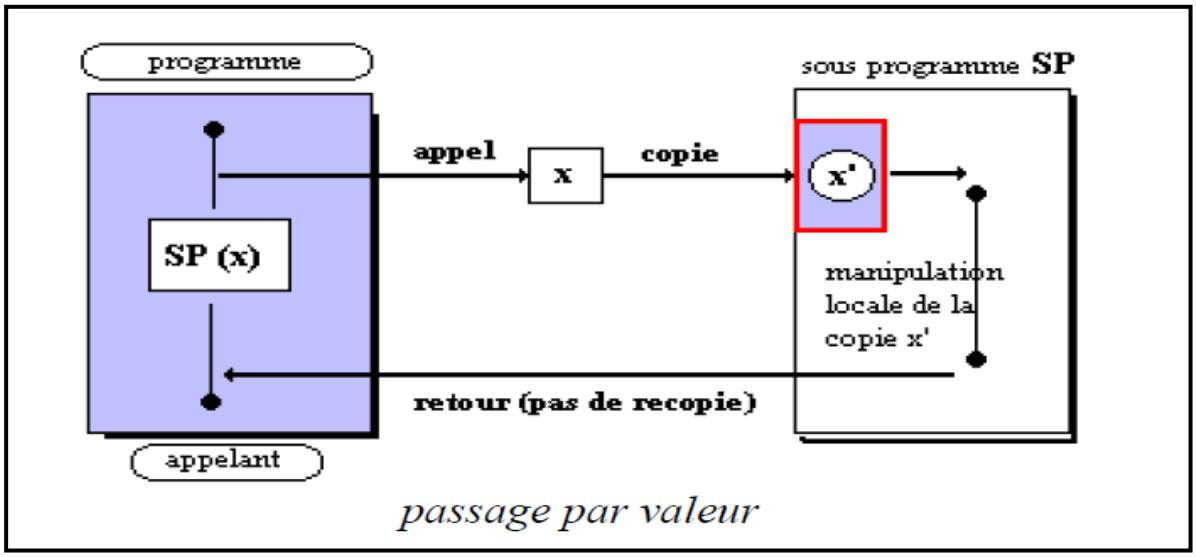
Entête	Variables globales	Exemples d'appels des fonctions dans le Programme Principal
Procédure P1 (n : entier ; ch : chaine)	J, i : entier Y : réel C : caractère Mot : chaine ok : booléen	
Procédure P2 (ch : chaine ; X : real)		
Fonction F3 (n,p : entier ; M :caractère) :booléen		

c. Modes de passage des paramètres

La substitution des paramètres effectifs aux paramètres formels s'appelle passage de paramètre. Il s'agit en fait d'un transfert de données entre le PP et le sous-programme appelé. Pour les sous-programmes, il existe deux modes de passage des paramètres : le **mode de passage par valeur et celui par variable**.

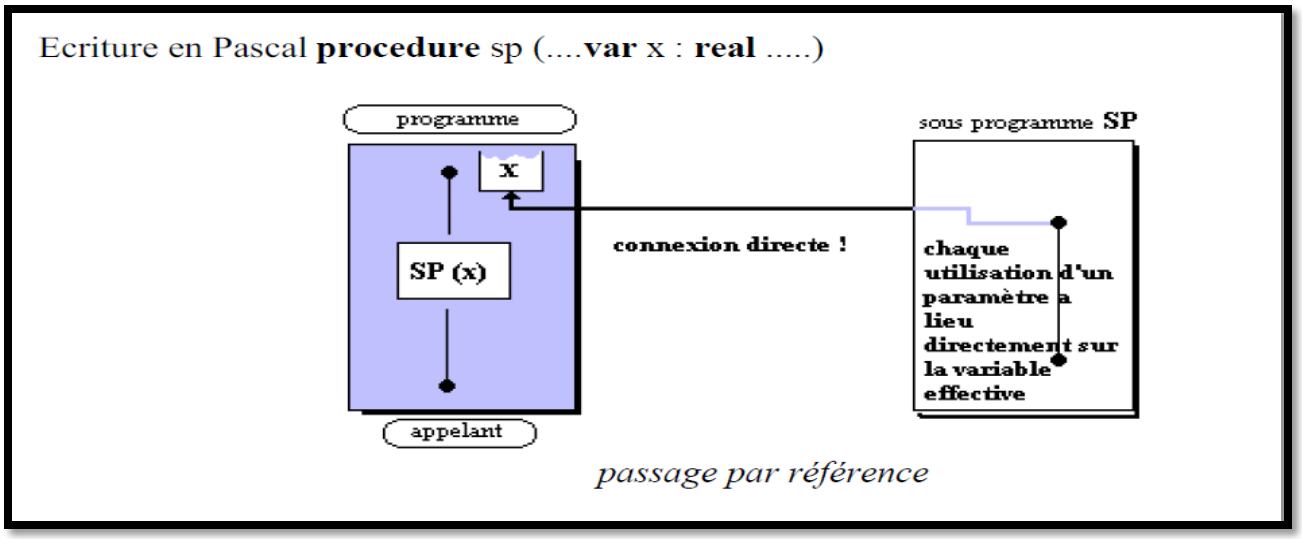
★ **Le passage de paramètres par valeur**

a. Le paramètre formel n'est pas précédé par le mot VAR.
b. La communication est à sens unique (Appelant → appelé).
c. La valeur du paramètre ne change pas.



★ **Le passage de paramètres par variable**

a. Le paramètre formel est précédé par le mot VAR.
b. La communication est à double sens (Appelant ↔ appelé).
c. La valeur du paramètre peut se changer.



☞ Pour mieux assimiler

On donne le programme pascal suivant **:**
(donner les valeurs de x et y
pour chaque opération d'affichage)

```
uses wincrt ;
var
x,y :integer ;

procedure p1(a,b :integer) ;
var aux :integer ;
begin
      aux :=a ;
      a :=b ;
     b :=aux ;
end ;
```

```
procedure p2(var a,b :integer) ;
var aux :integer ;
begin
      aux :=a ;
      a :=b ;
     b :=aux ;
end ;

begin
x :=10 ;
y :=15 ;
writeln('x=',x,'Y=',y)  ;
P1(x,y) ;
writeln('x=',x,'Y=',y)  ;
P2(x,y) ;
writeln('x=',x,'Y=',y)  ;
end ;
```