PROCÉDURES ET FONCTIONS

PROCÉDURES ET FONCTIONS

℘ Problème:

- ✓ Écrire un algorithme qui calcul la partie entière de deux nombres réels puis d'afficher la plus grande valeur des deux avant de calculer la somme et la moyenne des deux nombres entiers obtenus.

℘ Solution:

- ✓ Décomposer le problème en sous problèmes et trouver une solution à chacun puis regrouper le tout dans un seul algorithme

Chaque solution partielle donne lieu à un sous-algorithme

PROCÉDURES ET FONCTIONS

℘ Sous-algorithme:

- ✓ Un sous-algorithme est un bloc faisant partie d'un algorithme. Il est déclaré dans la partie entête (avant le début de l'algorithme), ou bien en dehors de l'algorithme, puis appelé dans le corps de l'algorithme.
- ✓ Étant donné qu'il s'agit d'un bloc à part entière, il possède éventuellement un en-tête, une série de traitements, et une gestion des résultats tout comme l'algorithme qui le contient.

PROCÉDURES ET FONCTIONS

℘ Types de Sous-algorithme:

- ✓ Un sous-algorithme peut se présenter sous forme de fonction ou de procédure.
- ✓ Une fonction est un sous-algorithme qui, à partir de donnée(s), calcul et rend à l'algorithme Un et Un seul résultat alors qu'en général, une procédure affiche le(s) résultat(s) demandé(s).

PROCÉDURES ET FONCTIONS

∞ Procédure :

- ✓ Une procédure est un bloc d'instructions nommé et déclaré dans l'entête de l'algorithme et appelé dans son corps à chaque fois que le programmeur en a besoin.

Déclaration d'une procédure :

```
Procédure Nom_Procédure (Nom_Paramètre : Type_praramètre,……)
Variable
        Nom_variable : Type_variable        } Variables locales
        …
Début
        …
        Instructions                        } Corps de la procédure
        …
Fin
```

PROCÉDURES ET FONCTIONS

ꕥ **Exemple :** **Somme des 100 premiers nombres**

```
Algorithme essai
Variable
 I, S : entier
Procedure Somme ( )
Debut #Début de la Procédure
 S ←0
Pour I de 1 a 100 Faire
        S ← S + i
FinPour
Ecrire ('La somme des 100 premiers nombres est', S)
Fin #Fin de la Procédure
Debut #Début de l'algorithme
        Somme
Fin #Fin de l'algorithme
```

PROCÉDURES ET FONCTIONS

∞ Fonction :

- ✓ Une fonction est un bloc d'instructions qui retourne obligatoirement une et une seule valeur résultat à l'algorithme appelant. Une fonction n'affiche jamais la réponse à l'écran.

Déclaration d'une fonction :

```
Fonction Nom_Fonction (Nom_Paramètre : Type_praramètre,……) : type_Fonction
Variable
        Nom_variable : Type_variable        } Variables locales
        …
Début
        …
        Instructions                        } Corps de la procédure
        …
        Nom_Fonction ← Résultat
Fin
```

PROCÉDURES ET FONCTIONS

☙ Exemple 1:

```
Algorithme essai
Variable
I, Som : entier
Fonction Somme(): entier
Variable
S: entier
Debut #Début de la fonction
 S ←0
Pour I de 1 a 100 Faire
        S ← S + i
FinPour
Somme ← S
Fin #Fin de la fonction
Debut #Début de l'algorithme
        Som ← Somme
Ecrire ('La somme des 100 premiers nombres est', Som) ;
Fin #Fin de l'algorithme
```

PROCÉDURES ET FONCTIONS

☙ Exemple 2: Somme de deux nombres

```
Algorithme main
Variable
resultat: entier
A,B: entier
Debut
Ecrire ( "entrer deux nombres")
Lire (A, B)
        resultat ← Somme (A, B)
Ecrire ("La somme de", A, "et", B, "est", resultat) ;
Fin
```

```
Fonction Somme(nbr1: entier, nbr2: entier): entier
Debut
Somme ← nbr1 + nbr2
Fin
```

PROCÉDURES ET FONCTIONS

℘ Portée des Variables:

- ✓ Un sous-algorithme utilise les variables déclarées dans l'algorithme (appelées variables globales).
- ✓ Il peut aussi avoir ses propres variables (dites locales) déclarées dans l'espace qui lui est réservé ;
- ✓ Les variables locales ne peuvent être utilisées que dans ce sous-algorithme et nulle part ailleurs car sa portée (visibilité) est limitée au bloc qui la contient.
- ✓ L'espace de ces variables locales n'est réservé que lorsque le sous-algorithme est appelé et est libéré dès la fin de l'exécution.

PROCÉDURES ET FONCTIONS

☙ Exemple :

```
Algorithme essai
Variable
I, Som : entier
Fonction Somme(): entier
Variable
S: entier
Debut #Début de la fonction
 S ←0
Pour I de 1 a 100 Faire
        S ← S + i
FinPour
Somme ← S
Fin #Fin de la Procédure
Debut #Début de l'algorithme
        Som ← Somme
Ecrire ('La somme des 100 premiers nombres est', Som) ;
Fin #Fin de l'algorithme
```

PROCÉDURES ET FONCTIONS

℘ Exercice:

Ecrire une fonction qui calcule la partie entière d'un nombre positif.

Méthode:

L'utilisateur vas entrer un nombre réel et l'algorithme doit afficher la partie entière de ce nombre

Exemple:

Donner un nombre : 25,896

la partie entière de 25,896 est 25

PROCÉDURES ET FONCTIONS

℘ Réponse:

```
Algorithme main
Variable
A, resultat: entier
Debut
Ecrire ( "entrer un nombre")
Lire (A)
    resultat ← entiere (A)
Ecrire ("La partie entiere de", A, "est", resultat) ;
Fin
```

```
Fonction entiere(x: réel): entier
Variable
y: entier
Debut
y ← 0
Tantque y < x faire
    y ← y + 1
FinTanque ;
entiere ← y
Fin
```

PROCÉDURES ET FONCTIONS

℘ Mode de passages de paramètres :

- ✓ Un sous-algorithme avec paramètres est très utile parce qu'il permet de répéter une série d'opérations complexes pour des valeurs qu'on ne connaît pas à l'avance.
- ✓ Il existe deux types de passage de paramètres :
  - ➢ Passage par valeur
  - ➢ Passage par variable (dite aussi par référence ou encore par adresse).

PROCÉDURES ET FONCTIONS

℘ Passage paramètres par valeur :

- ✓ Lors de l'appel de la fonction, les valeurs des paramètres effectifs sont copiés dans des variables locales à la fonctions.
- ✓ le contenu des paramètres effectifs ne peut pas être modifié par les instructions de la fonction ou de la procédure ; car nous ne travaillons pas directement avec la variable, mais sur une copie.

PROCÉDURES ET FONCTIONS

℘ Exemple:

```
Algorithme positif
Variables
M : entier
Procedure P1 (nombre : entier)
Debut
Si nombre < 0 Alors
nombre ← - nombre
FinSi
Ecrire (nombre)
Fin
```

```
Debut
Lire (M)
P1 (M)
Ecrire (M)
Fin
```

PROCÉDURES ET FONCTIONS

℘ Passage paramètres par variable :

- ✓ Il s'agit non plus d'utiliser simplement la valeur de la variable, mais également son emplacement dans la mémoire (d'où l'expression « par adresse »).
- ✓ Le paramètre formel se substitue au paramètre effectif durant le temps d'exécution du sous-programme et à la sortie il lui transmet sa nouvelle valeur.
- ✓ Un tel passage de paramètre se fait par l'utilisation du mot-clé Var.

PROCÉDURES ET FONCTIONS

℘ Exemple:

```
Algorithme positif
Variables
M : entier
Procedure P1 (Var nombre : entier)
Debut
Si nombre < 0 Alors
nombre ← - nombre
FinSi
Ecrire (nombre)
Fin
```

```
Debut
Lire (M)
P1 (M)
Ecrire (M)
Fin
```

PROCÉDURES ET FONCTIONS

☙ Exercice:

Ecrire une procédure qui permet de permuter deux nombres positif.

Les deux nombre doivent être permuter dans leurs espace mémoire.

PROCÉDURES ET FONCTIONS

℘ Réponse:

```
Algorithme main
Variable
X, Y: entier
Debut
Ecrire ( "entrer deux nombre")
Lire (X, Y)
Ecrire ("La valeur des deux nombre avant permutation X=", X, "Y=" , Y )
permutation( X, Y)
Ecrire ("X=", X, "Y=" , Y )
Fin
```

```
Procedure permutation(var a: entier, var b: entier)
Variable
tmp: entier
Debut
tmp ← a
a ← b
b ← tmp
Ecrire ("La valeur des deux nombre apres permutation X=", a, "Y=" , b )
Fin
```