

# **Chapitre 4**

### Algorithmique & Programmation en C

Sous-algorithme (Fonctions et procédures)

# **Définition**



- Un sous-algorithme est un bloc à part entière qui est destiné à réaliser une tâche précise. Cette tâche peut être exécutée plusieurs fois.
- L'exécution d'un sous-algorithme s'effectue par un appel en précisant les paramètres
- Il est déclaré dans la partie en-tête de l'algorithme en déterminant les caractéristiques suivantes:
  - En-tête( l'identificateur, les arguments et le type de résultat à retourner)
  - Corps (Traitements)

# Types de sous-algorithme



Un sous-algorithme peut se présenter sous forme:

- •Fonction: un sous-algorithme qui retourne un résultat après l'exécution.
- Procédure: un sous-algorithme qui ne retourne rien après l'exécution.

## **Procédure**



#### **Déclaration:**

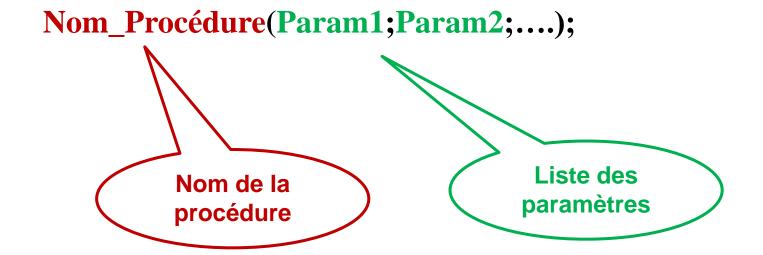
```
Procédure Nom_Procédure(Nom_Argum:Type_Argum;....);
 Variable Nom_Var: Type;
                                               Nom de la
                                               procédure
Début
  instruction1;
                                 Variables
  instruction2;
                                   locale
Fin;
                          Corps de la
                           procédure
```

## **Procédure**



### Appel:

L'appel de la procédure s'effectue (au moment souhaité) par son nom et éventuellement ses paramètres. Cet appel déclenche l'exécution des instructions de la procédure.



## **Procédure**



### **Exemple:** (Calculer la surface d'un rectangle)

```
Algorithme Recangle;
 Variable larg, long: Réel;
 Procédure Surface(A:Réel;B:Réel);
   Variable S:Réel;
  Début
       S \leftarrow A*B;
       Ecrire('La surface est: ',S);
  Fin;
Début
```

```
larg ← 5; long ← 7;
Surface(larg,long); //Appel

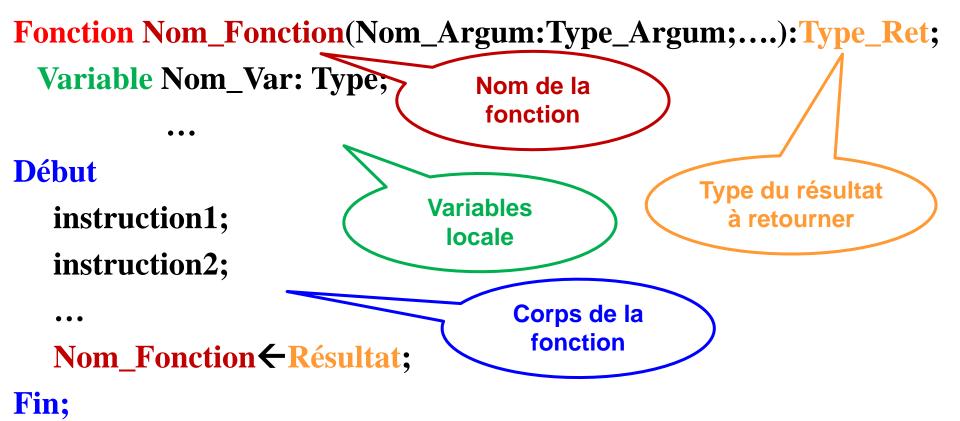
larg ← 8; long ← 9;
Surface(larg,long); //Appel

larg ← 6; long ← 4; //Appel
Surface(larg,long);
Fin
```

### **Fonction**



#### **Déclaration:**

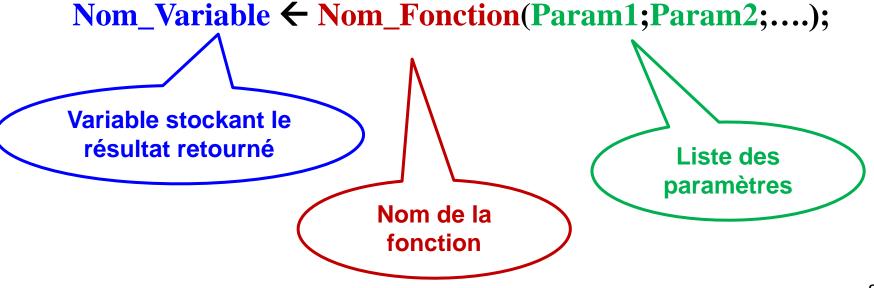


### **Fonction**



### Appel:

L'appel de la fonction s'effectue (au moment souhaité) par son nom, ses paramètres et en affectant le résultat retourné à une variable. Cet appel déclenche l'exécution des instructions de la fonction.



8

### **Fonction**



### **Exemple:** (Calculer la surface d'un rectangle)

```
Algorithme Recangle;
 Variable larg,long,res: Réel;
 Fonction Surface(A:Réel;B:Réel):Réel;
   Variable S:Réel;
  Début
       S \leftarrow A*B;
       Surface \leftarrow S;
  Fin;
Début
```

```
larg \leftarrow 5; long \leftarrow 7;
   res←Surface(larg,long); //Appel
   Ecrire('Résultat', res);
  larg \leftarrow 8; long \leftarrow 9;
   res ← Surface(larg,long); //Appel
  Ecrire('Résultat',res);
  larg \leftarrow 6; long \leftarrow 4; //Appel
  res← Surface(larg,long);
   Ecrire('Résultat', res);
Fin
```

# **Exercice**



On veut résoudre une équation du premier degré en utilisant la notion des sous-algorithmes.

Ecrire deux sous-algorithmes comme suit:

- Une fonction **lecture**(): saisir et retourner un réel
- Une procédure **solution**(float,float): résoudre une équation du premier degré.

# **Exercice** (Solution)



### Fonction lecture():

```
Fonction lecture():Réel;
Variable val: Réel;
Début
Ecrire('Entrer une valeur');
Lire(val);
lecture←val;
Fin;
```

# **Exercice** (Solution)



### Procédure solution():

```
Procédure solution(a: Réel, b: Réel);
 Variable x: Réel;
Début
   Si a <> 0 alors
       x\leftarrow-b/a; Ecrire(x);
   Sinon
       Si b=0 alors Ecrire('IR');
       Sinon Ecrire('Ensemble vide');
       Finsi
   Finsi
Fin;
```

# **Exercice** (Solution)



### **Conception:**

```
Algorithme Equation;
 Variable A,B: Réel;
 Fonction lecture():Réel;
 Variable val: Réel;
 Début
   Ecrire('Entrer une valeur');
   Lire(val);
   lecture←val;
  Fin;
  Procédure solution(a:Réel, b:Réel);
  Variable x: Réel;
  Début
```

```
Si a <> 0 alors
          x\leftarrow-b/a; Ecrire(x);
     Sinon
           Si b=0 alors Ecrire('IR');
          SinonEcrire('Ensemble vide');
         Finsi
      Finsi
   Fin;
Début
  A \leftarrow lecture();
   \mathbf{B} \leftarrow \mathbf{lecture}():
   solution(A,B);
Fin
```

# Récursivité



14

### Fonction récursive:

C'est une fonction qui contient un appel à elle même.

```
Fonction Nom_Fonction(Nom_Argum:Type_Argum;....):Type_Ret;
 Variable Nom_Var: Type;
                                   Appel de la fonction
                                       elle même
Début
  instruction1;
  instruction2;
  Nom_Fonction←Nom_Fonction(param,...);
```

Prof: R. EL AYACH

Fin:

# Récursivité



```
Exemple: Calculer: 1*2*3*....*N
```

```
Fonction Factoriel(N:Entier;....):Entier;

Début

Si N = 1 alors Factoriel ← 1;

Sinon Factoriel ← N*Factoriel(N-1);

Finsi

Fin;
```