Algorithmique

Chapitre XI

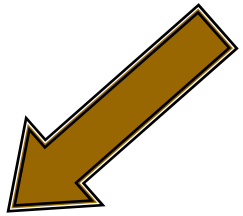
Procédures et fonctions

1 - Introduction

- Objectifs :
  - Décomposer un problème en sous problèmes
  - Généraliser l'utilisation de certaines parties de programmes
  - Eviter de répéter plusieurs fois les mêmes lignes de code
- Exemple : Nous voulons créer un programme qui permet de tester les connaissances d'un élève. Ce programme pose des questions auxquelles il faut répondre par O ou N.

1 - Introduction

Répétition de cette partie de programme



```
Variables Score : numérique
Variable Rep : caractère
Variable Question : Chaîne de caractère
Score ← 0
Question ← « Avez vous le droit de passer au rouge »
Ecrire Question
Répéter
        Lire Rep
Tant que (Rep <> « O » et  Rep <> « N »)

Si Rep =« N » alors Score ← Score + 1 Fsi

Question ← « Devez vous vous engager dans une rue en sens interdit »
Ecrire Question
Répéter
        Lire Rep
Tant que  (Rep <> « O » et Rep <>« N »)
Si Rep =« N » alors Score ← Score + 1 Fsi
…
Ecrire « Votre score est de »,Score
```

1 - Introduction

```
Ecrire Question
Répéter
        Lire Rep
Tant que  (Rep <> « O » et Rep <> « N »)
```

- Objectif : isoler cette partie de programme qui va être répétée
  - ⇒ Utiliser les ***procédures*** ou les ***fonctions***

2- Un premier exemple

```
Procedure LectureOuiNon () // Ligne de déclaration de la procédure
        Début                                   // Début du corps de la procédure
                Ecrire Question
                Répéter
                        Lire Rep
                Tant que  (Rep <> « O » et Rep <> « N »)
        Fin
```

2- Un premier exemple

- Appel de la procédure

…
Question ← « Avez vous le droit de passer au rouge »
LectureOuiNon() **// appel à la procédure LectureOuiNon**
Si Rep =« N » alors Score ← Score + 1 Fsi

Question ← « Devez vous vous engager dans une rue en sens interdit »
LectureOuiNon()
Si Rep =« N » alors Score ← Score + 1 Fsi
…
Ecrire « Votre score est de »,Score

2- Un premier exemple

- Synthèse sur l'exemple 1
  - LectureOuiNon() est une procédure
  - La déclaration de la procédure permet d'attacher un nom à une partie de programme
  - Les appels à la procédure permettent de déclencher son exécution, en interrompant, le temps d'exécuter le corps de la procédure, le déroulement séquentiel des instructions du programme

2- Un premier exemple

- Synthèse sur l'exemple 1
  - Le déroulement des instructions du programme reprend, dès que la procédure est terminée, à l'instruction qui suit l'appel
  - Pour exécuter l'algorithme, il faut commencer au début de la partie du programme que l'on nomme ***programme principal***

3- Deuxième exemple

- Lors d'une alternative, possibilité d'exécuter deux traitements différents comportant de nombreuses lignes
  - ⇒ Utilisation de deux procédures, appelées TraitementOui () et TraitementNon ()

```
Si condition alors
        TraitementOui()
Sinon
        TraitementNon()
Fsi
```

3- Deuxième exemple

1.5 - Synthèse sur l'exemple 2

- Utilisation de procédures :
  - Pour séparer diverses parties d'un algorithme afin de mieux le structurer pour le rendre plus lisible
  - Pour constituer des bibliothèques de procédures permettant de réutiliser le code (bibliothèque graphique, …)

4- Paramètres des procédures

4.1 - Paramètres en entrée : Transmission par valeur

- Dans l'exemple 1, les variables Question et Rep ont permis les communications de valeurs entre la procédure LectureOuiNon() et le programme principal.
- Ces deux variables sont dans le programme principal.
- Elles peuvent être utilisées dans la procédure.
  - ⇒ ***Variables globales***

4- Paramètres des procédures

4.1 - Paramètres en entrée : Transmission par valeur

- Ne pas utiliser trop de variables globales (problèmes si grand nombre de variables)
- Préférer l'utilisation de paramètres
- Procédure ayant ses propres variables
- Correspondance entre ces variables et les valeurs utilisées lors de l'appel de la procédure
- Exemple : passage de la question avec un paramètre

4- Paramètres des procédures

4.1 - Paramètres en entrée : Transmission par valeur

```
Procédure LectureOuiNon(Quest : Chaine de caractère)
Debut
        // Le paramètre est indiqué entre parenthèses
        // son type est soigneusement indiqué
        Ecrire Quest
        Répéter
                Lire Rep
        Tant que  Rep <> 'O' et Rep <> 'N'
Fin
```

4- Paramètres des procédures

4.1 - Paramètres en entrée : Transmission par valeur

- Le programme principal :

```
Variable Score : numérique
Variable Rep : caractère
Score ← 0
LectureOuiNon(« Peut on passer au feu rouge ? »)
Si Rep = 'N' alors
        Score ← Score + 1
Fsi
LectureOuiNon(« Doit on marquer un arrêt au stop ? »)
Si Rep = 'O' alors
        Score ← Score + 1
Fsi
Etc …
```

4- Paramètres des procédures

4.1 - Paramètres en entrée : Transmission par valeur

- Synthèse :
  - Le ou les paramètres sont écrits entre parenthèses
  - Dans la déclaration de la procédure utilisation d'un ***paramètre formel***
  - Lors de l'appel de la procédure, utilisation d'un ***paramètre effectif***
  - Dans le programme, LectureOuiNon(« Peut on passer au feu rouge? ») est automatiquement affecté au paramètre Quest

  => Disparition de la variable globale « Question » définie tout à l'heure

4- Paramètres des procédures

4.2 - Paramètres en sortie : transmission par adresse

- Exemple : Passage en paramètre de la réponse Rep
  - On peut écrire :

    Procédure LectureOuiNon (Quest : Chaine de caractère, Rep : caractère)

    ⇒ PB : La valeur de Rep n'est pas connue lors de l'appel
- Deux types de communication de valeurs par paramètre apparaissent :
  - Les paramètres en entrée
  - Les paramètres en sortie

4- Paramètres des procédures

4.2 - Paramètres en sortie : transmission par adresse

- Paramètre en entrée : La valeur du paramètre effectif est affectée avant l'exécution de la procédure au paramètre formel (qui est une variable appartenant à la procédure)
- Paramètre en sortie : Le paramètre formel est une autre désignation du paramètre effectif, valable pendant la durée de la procédure.

4- Paramètres des procédures

4.3 - Remarques

- Dans les langages de programmation on parle de :
  - Passage de paramètre par valeur
    - ⇒ l'appel de la procédure fait affecter une valeur à la variable qui est désignée par le paramètre formel
  - Passage de paramètre par adresse
    - ⇒ L'ordinateur utilise le paramètre formel comme une variable stockée en mémoire à l'adresse du paramètre effectif. Cette adresse est transmise à la procédure au moment de son appel
- ***Le passage par adresse permet de faire entrer et sortir des valeurs d'une procédure***

4- Paramètres des procédures

4.4 - Notations

- Exemple : entrée et / ou sortie d'un paramètre numérique :
  - Entrée de la valeur : →P : numérique
  - Entrée et sortie de la valeur : ↔ P : numérique

4- Paramètres des procédures

4.5 - Exemple

- Procédure permettant de remplacer un nombre entier par le premier nombre premier qui est lui est supérieur ou égal :
  - Définition de la procédure :

    Procedure NombrePremierSuivant (↔ Val : numérique)
  - Programme principal :

```
Variables Nb : numérique
Ecrire « Entrez un nombre »
Lire Nb
NombrePremierSuivant(Nb)
Ecrire « Le résultat est : », Nb
```

5 - Une procédure particulière : la fonction

- Procédure particulière qui peut renvoyer un résultat
- Leur appel ne constitue pas à lui seul une instruction, mais figure dans une expression
- Leur exécution produit un résultat qui prend la place de la fonction lors de l'évaluation de l'expression
- Exemple : la fonction racine qui permet de calculer la racine carrée d'un nombre

  Rac ← Racine_Carrée (A)

5 - Une procédure particulière : la fonction

- Pour définir une fonction, il suffit :
  - De la déclarer comme une procédure, en la faisant précéder du mot fonction et suivre du type résultat
  - De faire figurer dans le corps de la fonction le mot résultat
  - Le mot résultat est suivi de la valeur ou de l'expression qui donne le résultat à fournir au programme

5 - Une procédure particulière : la fonction

- Exemple : Algorithme où l'on calcule la somme des entiers de 1 à N en utilisant la fonction Somme.

```
Fonction Somme (→Max : Numérique) : Numérique
Début
        Acc ← 0
        Répéter pour i = 1 à Max
                Acc ← Acc + i
        FinPour
Résultat Acc
Fin
```

5 - Une procédure particulière : la fonction

- Programme principal de l'exemple

```
Variable N : numérique
Ecrire « Somme des entiers de 1 jusqu'à ? »
Lire N
Ecrire Somme (N)
```

- L'annonce du type lors de la création de la fonction permet de vérifier que celui-ci est correct lors de l'appel
- Remarque :
  - Ne déclarer les paramètres des fonction qu'en entrée seulement
  - Préférer les procédures avec plusieurs paramètres en entrée / sortie s'il est nécessaire de renvoyer plusieurs résultats

6 - Variables locales et variables globales

- Exemple : Somme des entiers de 1 à N

```
Fonction Somme (→Max : Numérique) : Numérique
Début
        Variable i , Acc: numériques
        Acc ← 0
        Pour i de 1 à Max Faire
                Acc ← Acc + i
        finPour
Résultat Acc
Fin
```

- Les variables i et Acc sont déclarées à l'intérieur de la fonction => **Variables LOCALES**

6 - Variables locales et variables globales

- Dans une procédure, il est possible d'utiliser des variables du programme principal
  - ⇒ **variables GLOBALES**
- Attention :
  - aucune variable locale ne doit porter le même nom dans la procédure
  - **Ne pas utiliser de variables globales à l'intérieur de procédures => problèmes d'effets de bord**

7 - Décomposition d'un algorithme

- Possibilité d'appeler une fonction ou une procédure à l'intérieur d'une autre fonction ou d'une autre procédure
- Permet de créer des algorithmes clairs et agréables à lire
- Utilisation pour décomposer un problème en sous problèmes
- Question : mais où s'arrêter dans la décomposition ?
- Exemple :

```
Procédure Proc1 (…)
Début
        …
        Proc2 (…)
        …
Fin
```