Université Ahmed Zabana de Relizane

Spécialité : LMD MIAS

Année : 1ère (Semestre 1)

Module : Algorithmique et structure de données 1

Introduction à la programmation et à l'algorithmique

I. Introduction

On a noté dans le premier chapitre qu'un ordinateur sert à faire des traitements automatiques sur les informations. Autrement dit, l'activité de programmation permet à un ordinateur de donner des solutions à des problèmes réels. De façon générale, la programmation consiste, à partir d'un problème donné, à réaliser un programme dont l'exécution apporte une solution satisfaisante au problème posé. Le cœur du processus de résolution d'un problème est la conception d'algorithmes. Pour cela, on va essayer d'introduire la notion d'algorithme dans le présent chapitre.

II. Définitions

Programme : C'est une suite, finie et ordonné, d'instructions écrites dans un langage compréhensible par la machine. Cette suite est mise en place afin de donner une solution à un problème spécifique. Généralement, un programme a la structure suivante :

En-tête du programme

Partie déclarative

Partie opérationnelle

En-tête du programme : Dans cette partie, on indique à l'ordinateur le nom du programme afin de lui permettre de le distingué des autres programmes.

Partie déclarative : Toutes les informations (données, variables et constantes), nécessaire à l'ordinateur pour résoudre le problème donné, sont lui indiquées dans la partie déclarative.

Partie opérationnelle : Les opérations sur les informations déclarées dans la partie précédente se trouvent dans la partie opérationnelle. La solution à notre problème est donnée par l'exécution de la suite des opérations de cette partie.

Instruction : Une instruction est un ordre du programmeur à l'ordinateur. Cet ordre est mis en exécution par l'ordinateur afin de donner le résultat souhaitable par le programmeur.

III. Cycle de développement d'un programme

Le cycle de développement d'un "programme informatique " peut se résumer ainsi :

Problème --> Analyse --> Algorithme --> Programme --> Compilation --> Exécution

Problème : en voici un exemple, trier par un ordre croissant les nombres a, b, c .
Analyse : phase de réflexion qui permet d'identifier les caractéristiques du Problème à traiter Ainsi sur notre exemple, comment trier les nombres? à l'aide de …….
Algorithme : Il s'agit d'une description compréhensible par un être humain de la suite des opérations à effectuer pour résoudre le Problème. Néanmoins, Dans notre exemple, cela donnerait :

- Lire les nombres
- Comparer deux nombres quelconques.
- Comparer le plus petit nombre parmi les deux par le troisième nombre
- si le troisième est le plus petit, tu as trouvé l'ordre
- sinon comparer le troisième avec l'autre nombre puis trouver l'ordre.

Programme : fichier texte des instructions et de leurs enchaînements dans un langage informatique .Pour notre exemple, nous traduirions en langage Pascal l'algorithmes pour obtenir un programme Pascal

Compilation : transformation en langage machine d'un programme écrit en langage évolué

Exécution : l'ordinateur exécute les instructions d'un programme en langage "binaire". Ainsi, l'utilisateur "lance" l'exécution de programme compilé pour savoir quel est l'ordre de a,b,c.

IV. Définition d'un Algorithme

Un algorithme est une suite d'instructions, qui une fois exécutée correctement dans un ordre bien déterminé donne une solution au problème posé. Il permet d'expliciter clairement les idées de solution d'un problème indépendamment d'un langage de programmation.

Un algorithme doit être exprimé dans un langage clair et bien défini. C'est le langage Algorithmique.

V. Structure générale d'un algorithme

Un algorithme est composé de trois parties :

1) ***Entête :*** qui spécifie le nom de l'algorithme par exemple : Algorithme tri ;
2) ***Déclaration :*** contient la déclaration de différentes variables utilisées avec leurs types.
3) ***Corps*** : contient la séquence d'instructions entre les deux mots clés : Début et Fin

La structure générale d'un algorithme est la suivante :

Algorithme <Nom de l'algorithme>

Déclaration liste de variables : type de variables ;

Début
instructions...
Fin

Exemple :

```
Algorithme Affichage ;
Début
   écrire ('Bienvenue au Centre Universitaire de Relizane') ;
Fin.
```

Remarque : On observe que la partie des déclarations est vide, car on n'a pas besoin de données pour résoudre ce problème.

VI. Notions de variables, types et valeurs

Une variable sert à stocker l'information traitée par un algorithme. Cette variable est caractérisée par son nom (identificateur) et son type :

Nom : permet de l'identifier parmi d'autres objets.

Type : Détermine la nature de la valeur de la variable que l'on souhaite utiliser

Une fois qu'un type de données est associé à une variable le contenu de cette variable doit **obligatoirement** être du même type.

Une variable est déclarée à l'aide du mot VAR. L'expression de déclaration de variables s'écrit suivant la règle qui suit :

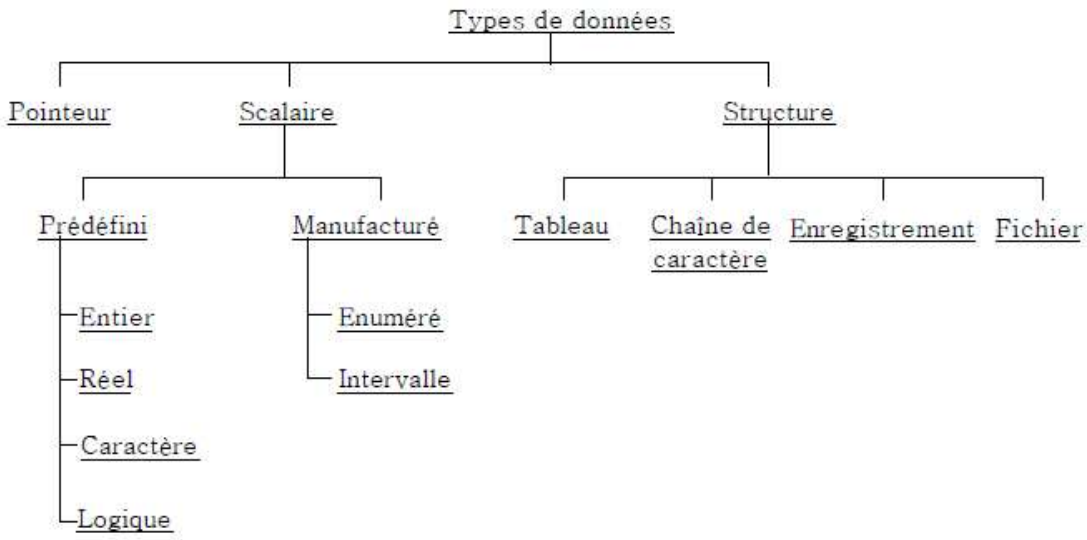
VAR <Nom variable> : <Type variable>

Exemple :

```
VAR
      salaire : Réel
      trouver : Logique
      symbole : Caractère
```

VI.1 Type d'une variable

Les types de données sont résumés dans le diagramme suivant :



Dans cette partie du cours, on s'intéresse au type scalaire.

VI.1.1 Types prédéfinis :

Ces types sont des types de base, ils n'ont pas besoin d'être définis par l'utilisateur. Ils sont :

- **entier** Correspond à l'ensemble des entiers (Z),
- **réel** Correspond à l'ensemble des réels (R),
- **booléen** Correspond aux deux valeurs logiques : {FAUX, VRAI},
- **caractère** pour manipuler des caractères alphabétiques et numériques. Une **chaîne de caractères** permet de représenter des mots ou des phrases.

VI.1.2 Types manufacturé :

Ces types sont définis par l'utilisateur selon ses besoins.

- **Type énuméré** : Les valeurs de ce type sont énumérées (On cite explicitement toutes les valeurs).

Déclaration : TYPE <Nom Type> = (<Val_1>,<Val_2>,…,<Val_n>)

Exemples :

```
TYPE
   jours semaine = (samedi,dimanche,lundi,mardi,mercredi,jeudi,vendredi)
   Fruits = (orange,pomme,banane,mandarine)
```

Remarques : 1. Les valeur de ce type sont ordonnées : samedi < dimanche < lundi < …

- **Type intervalle** : Etant donné un type scalaire (autre que Réel), on peut en définir un type intervalle comme étant un nouveau type. Les valeurs de ce type sont comprises entre une valeur inférieure et une valeur supérieure.

Déclaration : TYPE <Nom Type> = <Val Inf> … <Val Sup>

Exemples :

TYPE
lettres = 'E' … 'R'
années = 1900 … 2006
jours ouvrables = samedi … mercredi

VI.2 Valeur d'une variable

Dans la partie déclaration, les variables n'ont pas de valeur. Attribuer une valeur à une variable se fait par le biais d'une instruction. Initialiser une variable, c'est lui donner une première valeur.

VII. Notion de constante

Une constante est une valeur fixée d'avance à laquelle un nom est attribué. Une constante est déclarée à l'aide du mot CONST.

L'expression de déclaration de constantes s'écrit suivant la règle qui suit :

CONST <Nom constante> = <Valeur constante>

Exemple :

CONST Pi = 3.14
ville = 'BISKRA'

Remarque : les constantes de type caractères ou chaîne de caractères sont enfermées entre apostrophes.

VIII. Opérations élémentaires et expressions

VIII.1 Définitions :

- Un **opérateur** est un symbole d'opération qui permet d'agir sur des variables ou de faire des "calculs"
- Un **opérande** est une entité (variable, constante ou expression) utilisée par un opérateur
- Une **expression** est une combinaison d'opérateur(s) (arithmétiques, logiques, relationnels) et d'opérandes (constantes, variables), elle est évaluée durant l'exécution de l'algorithme, et possède une valeur (son interprétation) et un type

Remarques : Le résultat d'une expression de condition est toujours de type logique.

VIII.2 Les opérations élémentaires

Sur les entiers : Les opérations utilisables sur les entiers sont :

- les opérateurs arithmétiques classiques : + (addition), - (soustraction), * (produit)

- la division entière, notée DIV, telle que n DIV p donne la partie entière du quotient de la division entière de n par p
- le MODulo, noté MOD, telle que n MOD p donne le reste de la division entière de n par p
- les opérateurs de comparaison classiques : <, >, =, ...

Sur les réels : Les opérations utilisables sur les réels sont :

- les opérations arithmétiques classiques : + (addition), - (soustraction), * (produit), / (division)
- les opérateurs de comparaison classiques : <, >, =, ...

Sur les booléens :
Il s'agit du domaine dont les seules valeurs sont *vrai* ou *faux*. Les opérations utilisables sur les booléens sont réalisées à l'aide des connecteurs logiques : ET (pour le *et* logique), OU (pour le *ou* logique), NON (pour le *non* logique).

Sur le caractère :
Il s'agit du domaine constitué des caractères alphabétiques et numériques. Une variable de ce type ne peut contenir qu'un seul et unique caractère.
Les opérations élémentaires réalisables sont les comparaisons : <, >, =, ... selon l'ordre lexicographique

Sur une chaîne de caractères :
Une chaîne est une séquence de plusieurs caractères.
Les opérations élémentaires réalisables sont les comparaisons : <, >, =, ... selon l'ordre lexicographique, et la Concaténation représenter par un +

IX. Instructions sur les variables

IX.1 Instruction d'affectation

C'est pour affecter un (nouveau) contenu : Cela s'effectue en utilisant l'opérateur d'affectation représenté par le symbole ⟵ . L'instruction d'affectation s'écrit suivant la règle qui suit :

<Nom variable> ⟵ <Expression>

C'est-à-dire la variable à gauche reçoit la valeur de l'expression à droite

Exemple :

```
Age ← 24 ;
X   ← 2 ;
A   ← ((A+B)*2) ;
```

IX.2 Les instructions de lecture et d'écriture

a- Instruction de lecture *:* L'instruction de prise de données sur le périphérique d'entrée (en général le clavier) est : **Lire (variable) ;**

L'exécution de cette instruction consiste à affecter une valeur à la variable en prenant cette valeur sur le périphérique d'entrée. Avant l'exécution de cette instruction, la variable avait ou n'avait pas de valeur. Après, elle a la valeur prise sur le périphérique d'entrée

Exemple : Supposons x une variable de type entier alors :

Lire(x); Affectera une valeur de type entier taper au clavier à la variable X.

La fin de la valeur est reconnue en tapant la touche ENTREE

Remarque : L'instruction de lecture bloquera l'exécution de programme jusqu'à ce que la touche ENTREE soit tapée.

b- Instruction d'écriture : L'instruction de restitution de résultats sur le périphérique de sortie (en général affichage sur l'écran) est :

Ecrire (liste d'expressions);
Ecrire (variable);

Cette instruction réalise simplement l'affichage des valeurs des expressions décrites dans la liste. Ces expressions peuvent être simplement des variables ayant des valeurs ou même des nombres ou des commentaires écrits sous forme de chaînes de caractères.

Exemple : supposons x=0, y=0 alors:
écrire(x, y+2, "bonjour"); Affiche sur l'écran: 0 2 bonjour

c- Les commentaires

Il est important d'écrire l'algorithme sous une forme claire et logique. Pour cela, il est possible de rajouter des commentaires aux actions algorithmiques.
On convient qu'un commentaire débute par slash suivie d'une étoile * et se termine par une étoile * suivie d'un slash ou double slash pour un commentaire qui s'étale sur une seule ligne:

/* Plusieurs ligne */ **OU**
// Une seule ligne

Un commentaire n'a aucun effet sur l'exécution d'un algorithme mais facilite sa compréhension

X. Trace d'exécution

L'exécution d'un programme (algorithme) se fait instruction par instruction selon l'ordre indiqué par l'algorithme. Elle commence à partir de la première instruction qui suit le mot clé DEBUT et se termine à la dernière instruction qui précède juste le mot clé FIN.

Au début les valeurs des variable sont inconnues « ? ». Quand une variable reçoit une valeur, cette valeur est sauvegardée jusqu'à ce qu'une autre instruction change cette valeur.

Exemple :

```
Algorithme trace
   x : entier
   y : entier
Début
      x ← 12 ;
      y ← x + 4 ;
      x ← 3 ;
Fin.
```

Instructions	x	y
(1)	12	?
(2)	12	16
(3)	3	16