Université A. Mira Béjaia
Faculté de Sciences Exactes
Département Mathématiques
L1 Math (tronc commun)
2025/2026

Module : Algorithmique et Structures de Données 1 (ASD1)

Chapitre 2 : Algorithme séquentiel simple

Dans ce chapitre nous nous intéressons à la présentation de la structure générale d'un algorithme et de sa partie réservée aux déclarations.

1. Structure générale d'un algorithme

Un algorithme se présente comme suit :

Algorithme nom de l'algorithme // **L'en-tête**

Const Liste_des_constantes
Type Liste_des_types
Var Liste des variables
Liste des procédures et fonctions

→ **Partie déclarations**

Début

Instruction 1 ;
Instruction 2 ;
…
Instruction N ;
Fin.

→ **Partie Action (Le corps de l'algorithme)**

- **L'en-tête** permet d'identifier l'algorithme. C'est-à-dire, elle sert à lui donner **un nom**.
- **Les déclarations:** C'est une liste de tous les **objets** (constantes, variables ...) utilisés et manipulés dans la partie action ou traitement de l'algorithme (le corps de l'algorithme).
- **La partie action :** Dans cette partie sont placées **les opérations et les instructions** à exécuter. La partie actions est délimitée par les mots clés '**Debut' et 'Fin'.**

NB : chaque déclaration et chaque instruction doit se terminer par un **' ;'**

Exemple : L'algorithme qui permet de calculer la somme de deux nombres entiers X et Y.

```
Algorithme Somme ;
Var X, Y, S: Entier ;
Début
Lire (X, Y) ;
S←X+Y ;
Ecrire (S) ;
Fin.
```

2. La partie déclaration (variables et constantes)

2.1. Définition d'un identificateur

Un identificateur (identifiant) est un nom qui respecte une syntaxe particulière :

- Il est constitué d'une suite de lettres de l'alphabet (latin) et de chiffres.
- Il commence, obligatoirement, par une lettre.
- Le caractère souligné "_" (Le tiret bas, underscore) peut jouer le même rôle qu'une lettre de l'alphabet. Sur le clavier, le caractère "_" se trouve sur la même touche qui permet d'écrire le chiffre 8.

Exemples : a, b, X1, X2, Num_Etudiant …

Remarques

- Pour faciliter la lisibilité d'un algorithme, il est préférable d'utiliser des noms significatifs.
- Ne doit pas commencer par un chiffre.
- L'identificateur doit être différent de tous les mots clés (Algorithme, Debut, Fin, Si, …).
- Les caractères de l'alphabet grec sont interdits. Si on a besoin d'utiliser α, β, ou δ, on doit les écrire en alphabet latin comme alpha, béta, gamma et ainsi de suite.
- Il est interdit d'utiliser des caractères accentués (é, è, î, ç, ', ",…) dans un identifiant.
- Il est interdit d'utiliser des symboles mathématiques (+, -, * …), des caractères spéciaux ($, #...), ou les ponctuations (« . » « , » « ; » « « ! » « ( ) »…) ou le caractère espace « ».
- Il est interdit d'utiliser des indices ou des exposants.

2.2. Définition d'une variable

Une variable est un espace mémoire identifié par un nom, destiné à stocker une valeur, qui peut être modifiée durant les traitements. Les variables d'un algorithme contiennent les informations nécessaires à son déroulement.

Var identificateur_de_la_variable : Type ;

2.3. Définition d'une constante

Une constante ne prend qu'une seule et unique valeur au cours de l'exécution de l'algorithme.

Const identificateur_de_la_constante= Valeur ;

2.4. Les types

2.4.1. Les types standards (prédéfinis)

Le type correspond au genre d'information utilisé. Les types standards (prédéfinis) en langage algorithmiques sont : **Entier, Réel, Booléen, Caractère, Chaîne (de caractères).** Il est nécessaire de préciser le type d'une variable pour savoir quelle est le nombre d'octets à attribuer à la variable. Chaque type est muni (dispose) d'un ensemble d'opérations.

1) Le type Entier

Le type Entier est utilisé pour manipuler des nombres entiers : -17, -8, 0, 13, 22, …
Le type entier est muni des opérateurs suivants :

- Les opérateurs arithmétiques: + (addition), - (soustraction), * (multiplication).
- La division entière, **notée DIV** , tel que **n div p** donne la partie entière du quotient de la division entière de n par p. **Exemple : 23 Div 5 = 4.**
- Le modulo, **noté MOD**, tel que **n mod p** donne le reste de la division entière de n par p. **Exemple : 23 Mod 5 = 3.**
- Les opérateurs de comparaison classiques : <, <=, =, <>, >=, >

2) Le type Réel

Le type Réel est utilisé pour manipuler des nombres réels (les nombres à virgule) : -3.7, -1.5, 0, 3.0, 18.25, …
Le type Réel est muni des opérateurs suivants :

- Les opérations arithmétiques classiques : + (addition), - (soustraction), * (multiplication), / (division).
- Les opérateurs de comparaison classiques : <, <=, =, <>, >=, >
- D'autres opérations sont définies par des fonctions dont on peut citer :
  - **trunc(x) :** Cette fonction donne la partie entière du nombre réel x.
  - **round(x) :** Cette fonction donne l'entier le plus proche du nombre réel x.
  - **abs(x) :** Cette fonction donne la valeur absolue du nombre réel x.
  - **sqrt(x) :** Cette fonction donne la racine carrée du nombre réel x.

3) Le type Booléen

Il s'agit du domaine dont les seules valeurs sont VRAI ou FAUX. Les opérateurs booléen (logiques) définis et les plus utilisés sont : le NON, le ET et le OU. Ils sont définis par les tables de vérité ci-dessous.
Soient A et B deux variables de type Booléen.

A	NON A
Faux	Vrai
Vrai	Faux

A	B	A ET B
Faux	Faux	Faux
Faux	Vrai	Faux
Vrai	Faux	Faux
Vrai	Vrai	Vrai

A	B	A OU B
Faux	Faux	Faux
Faux	Vrai	Vrai
Vrai	Faux	Vrai
Vrai	Vrai	Vrai

4) Le type Caractère

Le type caractère est un ensemble fini et totalement ordonné de caractères (symboles). Il comporte :

- Les lettres de l'alphabet latin : a .. z, A .. Z.
- Les chiffres : 0 .. 9.
- Les symboles utilisés en tant qu'opérateurs : + - * / < = > …
- Les caractères de ponctuation : . , ; ! ? …
- Les caractères spéciaux : @ % & # …
- Et d'autres.

Une constante de type caractère est représentée par un et un seul caractère encadré par **deux apostrophes simples**. **Exemples : 'a', 'b', 'C', '!', '#', …**

Chaque caractère est stocké en mémoire d'un ordinateur sur un octet.

Remarque

- Les caractères alphabétiques (majuscules et minuscules) se suivent et sont ordonnés dans l'ordre alphabétique ; C'est-à-dire 'A' < 'B' < … < 'Z' < … < 'a' < 'b' < … < 'z'.
- Les caractères numériques (les chiffres) se suivent et sont ordonnés dans l'ordre croissant ; C'est-à-dire '0' < '1' < … < '9'.

5) Le type Chaîne

Une chaîne est une suite de caractères. Une chaîne est encadrée par deux apostrophes. Exemples : 'Université de Bejaia", "Département Mathématique", "ASD1", …

- On appelle longueur d'une chaîne le nombre de caractères de cette chaîne. 'Algorithmique' est une chaîne de longueur 13.
- '' : Représente la chaine vide de longueur 0. Elle ne contient aucun caractère.

Exemples : Const N = 15 // Constante de type ENTIER
Pi = 3.14 // Constante de type REEL
Disponible = 'N' // Constante de type CARACTERE
Module = "ASD1" // Constante de type CHAINE
Var A, B: Entier // Déclaration de 2 variables de type Entier.
X, Y, Z : Réel // Déclaration de 3 variables de type Réel.
Stop : Booléen // Déclaration d'une variable de type Booléen.
Reponse : Caractère // Déclaration d'une variable de type Caractère.
Nom, prenom : Chaine // Déclaration de 2 variables de type Chaine.

2.4.2. Type Définie

Définition de nouveau type : Il désormais possible de définir des nouveaux type inexistant en algorithmique avec le mot clés **'Type'**.

1) **Le type intervalle :** Un intervalle est défini avec **sa borne inferieur et sa borne supérieur**. Le type des valeurs contenues dans l'intervalle est détecté automatiquement à partir de ses bornes.

Type Type_Inter = borne_inf .. borne_sup;

Exemple: Type positif = 0 .. 10;
négatif= -10..0 ;
VAR x: positif; y : négatif ;

La variable x peut contenir des valeurs entières comprises en 0 et 10.
La variable y peut contenir des valeurs entières comprises en -10 et 0.

2) **Les types énumérés :** parfois il est utile de déclarer une variable qui prend ses valeurs dans un ensemble fini.

Type nom_type = (valeur1, valeur2, . . . , valeurn) ;

Exemple : type jour= (Dimanche, Lundi, Mardi, Mercredi, Jeudi, Vendredi, Samedi);
Mois= (Janvier, Février, mars, Avril, Mai, Juin, Juillet, Aout, Septembre, Octobre, Novembre, Décembre) ;

3. La partie Action (corps de l'Algorithme)

3.1. Instructions de base

Il existe un ensemble d'instructions de base prédéfinis en algorithmique

3.1.1. Instruction d'entrée

L'instruction d'entrée permet de récupérer des valeurs saisies au clavier et de les sauvegarder dans des variables.

Lire (variable1, variable2,…, variablen) ;

L'instruction peut lire une ou plusieurs variables à la fois.

Exemple : Lire (X) ; Lire(X,Y,Z)

A l'exécution de l'instruction **Lire (X),** le programme se met en attente jusqu'à ce que l'utilisateur saisie une valeur au clavier et valide par la touche entré. La valeur saisie est sauvegardée ensuite dans la case mémoire réservée pour la variable X.

3.1.2. Instruction de sortie

L'instruction de sortie permet la communication avec l'utilisateur en affichant à l'écran les résultats de l'algorithme qui peuvent être le contenue des variables ou de simple message.

Ecrire (variable); Ecrire (constante); l'instruction **Ecrire** peut afficher plusieurs variables et plusieurs constantes en même temps séparées par des virgules;

Exemple :

Ecrire (N) ; affiche la valeur de la variable N.

Ecrire (Pi) ; affiche la valeur de la constante pi.

Ecrire (x,y,z) ; affiche les valeurs des variables x, y, et z.

Ecrire ('Bonjour') ; affiche le message Bonjour

3.1.3. L'affectation

L'instruction d'affection permet de modifier le contenu d'une variable. Elle symbolisée par le caractère « ← ».

La syntaxe :

Variable ← expression ;

- Expression peut être une constante, une variable ou une expression qui donne un résultat de type compatible avec la partie gauche de l'affectation.
- La partie gauche de l'instruction d'affectation doit être une variable.

Exemple : X←5 ; //Affecter une valeur à la variable X (X reçoit 5).

Y←Z ; // Copier la valeur de la variable Z dans la variable Y (Y reçoit Z).

X←Y*Y+5-X

3.2. Les expressions

Une expression est une suite d'opérations appliquées sur un ensemble **de facteurs (arguments ou paramètres).** Chaque expression a une valeur et un type.

Exemple : X + 3 est une expression qui représente une addition entre les deux facteurs X et 3**.**

3.2.1. Evaluation des expressions

Une expression arithmétique ou logique est évaluée en respectant l'ordre de priorité des opérateurs. La machine effectue un seul calcul à la fois, de ce fait lorsqu'une expression contient plusieurs opérateurs de même priorité, l'opérateur le plus à gauche est évalué en premier. L'ordre de priorité des opérateurs :

1. ( ). s'il y a plusieurs parenthèses, les plus internes sont prioritaires.
2. NON et (- unaire)
3. ET, *, /, div, mod
4. OU, +, -
5. <, <=, >, >=, =, < >