Algorithmique et structure de données

BTS DSI 1ere année LAAYOUNE

Notions de base

Un ordinateur est une machine électronique programmable servant au traitement de l'information codée sous forme binaire, c'est-à-dire sous forme de tout ou rien (soit le courant passe, soit il ne passe pas).

Un programme est un assemblage et un enchaînement d'instructions élémentaires écrit dans un langage de programmation, et exécuté par un ordinateur afin de traiter les données d'un problème et renvoyer un ou plusieurs résultats.

Notions de base

Un algorithme représente l'enchaînement des actions (instructions) nécessaires pour faire exécuter une tâche à un ordinateur(résoudre un problème).

Un algorithme s'écrit le plus souvent en pseudo-langage de programmation (appelé langage algorithmique)

Notions de base

Un algorithme n'est donc exécutable directement par aucune machine. Mais il a l'avantage d'être traduit facilement dans tous les langages de programmation.

L'algorithmique, l'art d'écrire des algorithmes, permet de se focaliser sur la procédure de résolution du problème sans avoir à se soucier des spécificités d'un langage particulier.

Etapes de réalisation d'un programme:

Enoncé du problème

↓ *Spécification*

Cahier des charges

↓ *Analyse*

Algorithme

↓ *Traduction en langage*

Programme source

↓ *Compilation*

Programme exécutable

↓ *Tests et modifications*

Version finale et résultats

La réalisation de programmes passe par l'écriture d'algorithmes.

- D'où l'intérêt de l'Algorithmique

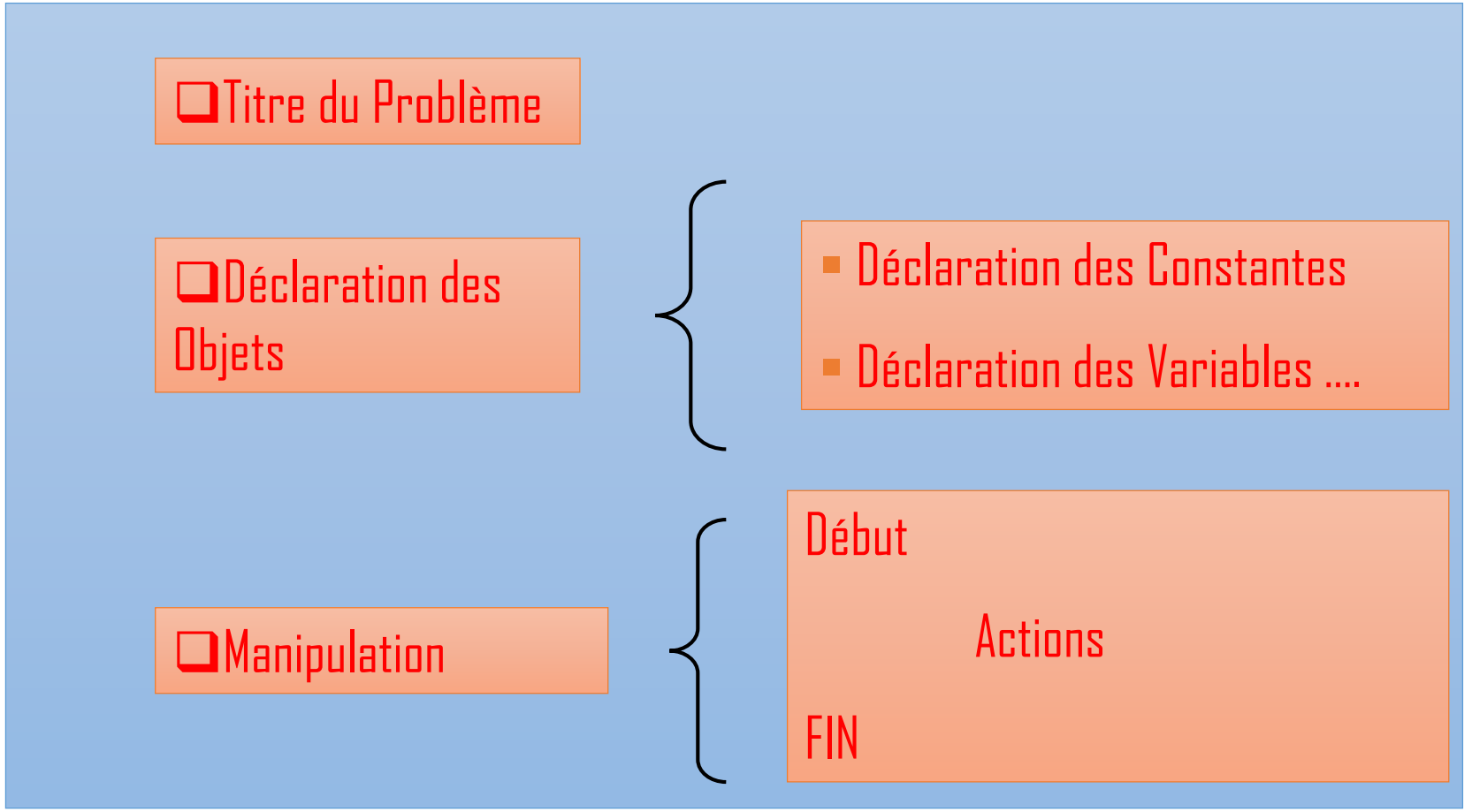
Notions de base

Langage de programmation : On appelle langage de programmation tout ensemble fini de mots réservés qui permettent de traduire les instructions de l'algorithme afin de l'exécuter par l'ordinateur

Programme source : Le programme source est le premier résultat de la traduction d'un algorithme en un langage évolué : Un nouvel ensemble d'instructions non exécutables directement par la machine

Compilateur : On appelle compilateur tout programme spécial qui permet d'avoir un programme exécutable à partir d'un programme source: Le programme ainsi obtenu est appelé programme Objet

Structure générale d'un algorithme:



Notion de variables et déclarations:

Les programmes ont pour but de traiter différentes données afin de produire des résultats. Les résultats peuvent eux-mêmes être des données pour d'autres programmes.

Notions de base

Une variable peut être représentée par une case mémoire, qui contient la valeur d'une donnée.

Chaque variable possède un nom unique, qui permet de l'identifier dans un algorithme, appelé identificateur par lequel on peut accéder à son contenu .

Par exemple, on peut avoir en mémoire une variable Nom et une variables Age qui contiennent les valeurs « Hamide » et 22

Remarques:

- Attention à ne pas confondre la variable et son contenu.

- Deux variables peuvent avoir la même valeur, mais une variable ne peut pas avoir plusieurs valeurs en même temps.

- En revanche, la valeur d'une variable peut varier au cours du programme. L'ancienne valeur est tout simplement écrasée et remplacée par la nouvelle.

- Les variables dont la valeur ne change pas au cours de l'exécution du programme sont appelées variables constantes ou plus simplement constantes.

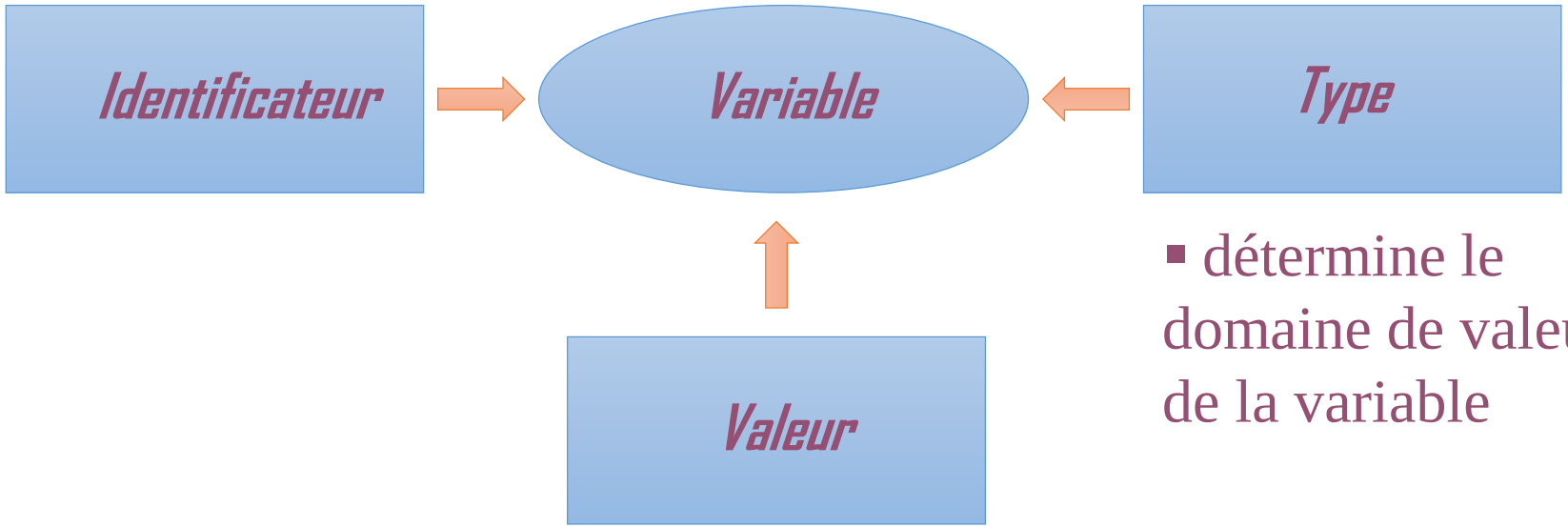
Déclaration des variables :

La déclaration d'une variable indique deux choses:

- Son identificateur (son nom)
- Son type (sa taille)

Un identificateur peut être composé de lettres et de chiffres mais il ne peut pas commencer par un chiffre et ne peut comporter d'espaces.

L'identificateur des variables doit être suffisamment signifiant pour qu'on reconnaisse leur fonction aisément.



- détermine le domaine de valeur de la variable

- incluse dans le domaine de valeur du type

Notions de base

En algorithmique, on distingue 5 types principaux:

- Les ***Caractères*** (lettres, chiffres, ponctuation, code des opérations, espace, retour chariot,… et plus généralement toutes les touches que l'on peut trouver sur une machine à écrire);
- Les ***Chaînes de Caractère*** (suites de caractères);
- Les ***Entiers*** (les nombres sans virgule);
- Les ***Réels*** (les nombres à virgule et sans virgule);
- Les ***Booléens*** (qui n'ont que deux valeurs possibles: soit VRAI, soit FAUX).

Type Entier:

- C’est l’ensemble des nombres entiers positifs ou négatifs.

Syntaxe de la déclaration :

Variable variable1,variable2,… : Entier

Exemple :

Variable a,b : Entier

Type Réel:

- C'est l'ensemble des nombres réels, c'est à dire les nombres décimaux sans limitation

Syntaxe de la déclaration :

Variable variable1,variable2,… : Réel

Exemple :

Variable x,y : Réel

Type Chaîne de caractères:

- C’est une suite de caractères, c’est à dire des combinaisons de caractères (lettres, chiffres, symboles..).

Syntaxe de la déclaration :

Variable variable1,variable2,… : Chaîne Caractère

Exemple :

Variable Nom, Catégorie : Chaîne Caractère

Type Booléen:

- Il s'agit des objets qui ne peuvent prendre que deux valeurs vrai ou faux.

Syntaxe de la déclaration :

Variable variable1,variable2,… : Booléen

Exemple :

Variable Décision : Booléen

Les opérateurs de l'algorithmique :

Les opérations possibles sur les variables dépendent de leur type (voir le tableau)

Type	Exemple	opérations possibles	symbole ou mot clé correspondant
réel	-15.69 , 0.36	addition soustraction multiplication division exposant pourcentage comparaisons	+ – * (et pas x pour ne pas confondre avec la lettre x) / ^ % <, ≤,>, ≥, =, ≠
entier	-10, 3, 689	addition soustraction multiplication division modulo exposant pourcentage	+ – * (et pas x pour ne pas confondre avec la lettre x) DIV cf. ci-après ① MOD cf. ci-après ② ^ %
caractère	'B' 'h' '£' '?' '\n'	comparaisons	<, ≤,>, ≥, =, ≠ cf. ci après ③
chaine	"Bonjour" "93000" "toto@caramail.com"	concaténation longueur extraction	& cf. ci-après ④ longueur(*chaîne*)
booléen	VRAI, FAUX	comparaison négation conjonction disjonction	<, ≤,>, ≥, =, ≠ NON ET OU

- Pour les entiers, la division est notée Div. Elle est nommée division entière et diffère un peu de la division que l'on trouve sur les calculettes. Elle ne donne que le chiffre avant la virgule du résultat (elle renvoie un entier).

- Les entiers supportent une opération supplémentaire appelée modulo, notée mod et qui renvoie le reste de la division entière.

Exemple:

7 / 2 donne 3.5

7 Div 2 donne 3

7 Mod 2 donne 1

Notion de constantes et déclarations:

Les Constantes désignent des références à des valeurs invariantes dans le programme.

Syntaxe de la déclaration :

Constante Nom_Constante = Valeur

Exemple :

Constante Pi ← 3.14

Syntaxe générale de l'algorithmique:

Algorithme Teste
‘’Pour déclarer une constante il est toute à fait normale de lui Affecter une valeur dès sa déclaration’’
Constante
pi← 3.14
‘’Pour déclarer les variable on utilise le mot clé Variable’’
Variable nom , prénom en Chaîne de caractère
Diamètre en Réel
Début
‘’
Les instructions
’’
Fin

Déclaration

Corps du Programme

Les instructions élémentaires :

La partie manipulation utilise les différents objets déclarés dans la partie déclaration et leur applique des opérations afin de retourner le(s) résultat(s) attendu(s) par le programmeur. Pour ce fait, les instructions élémentaires les plus courantes sont :

- l'Affectation: le fait de donner une nouvelle valeur à une variable
- l'Affichage sur l'écran
- la Saisie à travers le clavier

D'autre instructions permettent de lire et d'écrire sur d'autres périphériques: nous les étudierons plus tard.

Instruction d'affectation :

L'affectation consiste tout simplement à placer une valeur dans une variable (ce qui revient à changer le contenu de cette variable)

La nouvelle valeur est évaluée à partir d'une expression, qui peut être :

- soit une autre variable ou constante,
- soit une valeur littérale (valeur donnée explicitement dans le code source d'un programme)
- soit une combinaison de variables, de valeurs littérales et d'opérateurs

Pour affecter une valeur à une variable, on écrit :

Variable ⟵ Valeur

Remarque:

L'affectation est différente de l'égalité mathématique.

L'affichage des informations:

L'instruction d'affichage permet de fournir des résultats sous forme directement compréhensible pour l'utilisateur à travers l'écran.

Syntaxe

Afficher expression1

Exemple :

- Afficher "Bonjour!"

Celle-ci permet d'afficher la chaîne littérale Bonjour! à l'écran

- Afficher a, b

Quand on veut afficher deux objets à la suite, on les sépare d'une virgule
Si a vaut 5 et b vaut 10, on obtient alors à l'écran: 5 10

La Saisie des informations:

L’instruction de saisie permet de communiquer des données au programme.
Cette instruction assigne une valeur entrée au clavier dans une variable.
Tant que l'utilisateur n'entre rien au clavier, le déroulement du programme est stoppé.

Syntaxe:
Saisir variable1

La Saisie des informations:

Exemples:

- Saisir x

Cette instruction va lire la valeur saisie au clavier et l'affecte à la variable x

- Saisir x, y

Cette instruction lit la première valeur saisie au clavier et l'affecte à x, puis lit la
deuxième valeur saisie et l'affecte à y

Exemple complet :

Nous allons écrire un algorithme qui calcule l'âge de l'utilisateur en fonction de son année de naissance.

```
Algorithme âge
CONSTANTE année <- 2021 : entier
VARIABLE d_d_n : entier // date de naissance
DEBUT
Afficher "Entrez votre année de naissance sur 4 chiffres"
Saisir d_d_n
Afficher « Votre Age est : " , année – ddn , "ans."
FIN
```

Les conditions :

On appelle condition simple toute expression de la forme :

Variable 1 Opérateur Variable 2

Opérateur	Signification
+	Addition
-	Soustraction
*	Multiplication
/	Division
% ou mod	Modulo : le reste de la division de 2 valeurs entières
div	Division entière

Les opérateurs de Comparaison :

Pour exprimer les conditions, on utilise les opérateurs conditionnels suivants :

Opérateur	Signification
= =	Égal
<	Inférieur
>	Supérieur
<=	Inférieur ou égal
>=	Supérieur ou égal
<>	différent

Les opérateurs logiques de relation :

On peut combiner des conditions à l'aide des opérateurs logiques :

Opérateur	Signification
Et	Et logique
Ou	Ou logique
Non	Négation logique

Priorités des opérateurs

Priorité de *, / div et % par rapport à + et -

5 + 9 * 3 = 32 et non 42

5*9+3 = 48 et non 60

Pour les opérateurs de même priorité, associativité à partir de la gauche

15 / 5 * 3 = 9 et non 1

5 – 2 + 4 = 7 et non –1

On peut utiliser des parenthèses pour changer l'ordre des opérations :

15 / (5 * 3) = 1

(5 + 9) * 3 = 42

Les structures de contrôle

Parfois, il est nécessaire que le processeur n'exécute pas toutes les instructions, ou encore qu'il recommence plusieurs fois les mêmes instructions. Pour cela, il faudra casser la séquence. C'est le rôle des structures de contrôle.

Il existe deux grands types de structures de contrôles

- les structures conditionnelles vont permettre de n'exécuter certaines instructions que sous certaines conditions
- les structures répétitives, encore appelées boucles, vont permettre de répéter des instructions un certain nombre de fois, sous certaines conditions

Les structures conditionnelles :

Les structures conditionnelles permettent d'exécuter des instructions différentes en fonction de certaines conditions. Une condition (encore appelée expression conditionnelle ou logique) est évaluée, c'est à dire qu'elle est jugée vrai ou fausse. Si elle est vraie, un traitement (une ou plusieurs instructions) est réalisé; si la condition est fausse, une autre instruction va être exécutée, et ensuite le programme va continuer normalement.

Il existe 2 types principaux de structures conditionnelles

- les structures alternatives (Si…Alors…Sinon)
- les structures conditionnelles au sens strict (Si…Alors)

L'instruction Si :

Pour exprimer le fait que des instructions vont être exécutées dans un cas alors que d'autres instructions peuvent être exécutées dans l'autre cas, on utilise une structure alternative.

Syntaxe :

```
Si Condition_1  alors
        Actions 1
Sinon Si    condition_2  alors
          Actions 2
Sinon
                Option Facultative
Finsi
```

L'instruction Suivant Cas :

L'instruction Suivant cas constitue une structure alternative à la forme en bloc [Si … Alors …Sinon…] et permet de formuler de manière plus simple le choix du groupe d'instructions.

Syntaxe :

```
Suivant Cas  variable  Faire
  Cas  Valeur 1
          Actions 1
  Cas  Valeur 2, Valeur3, Valeur 4
          Actions 2
  Cas  Valeur 5 à Valeur 7
          Actions 3
    ..
    ..
  Sinon Cas
          Actions N
 Fin Suivant
```