TEK-UP University

CHAPITRE 1

Initiation à l'algorithmique

ING-3-J

Mme Nour EL AOUINI

SOMMAIRE

1. Définition d’un algorithme

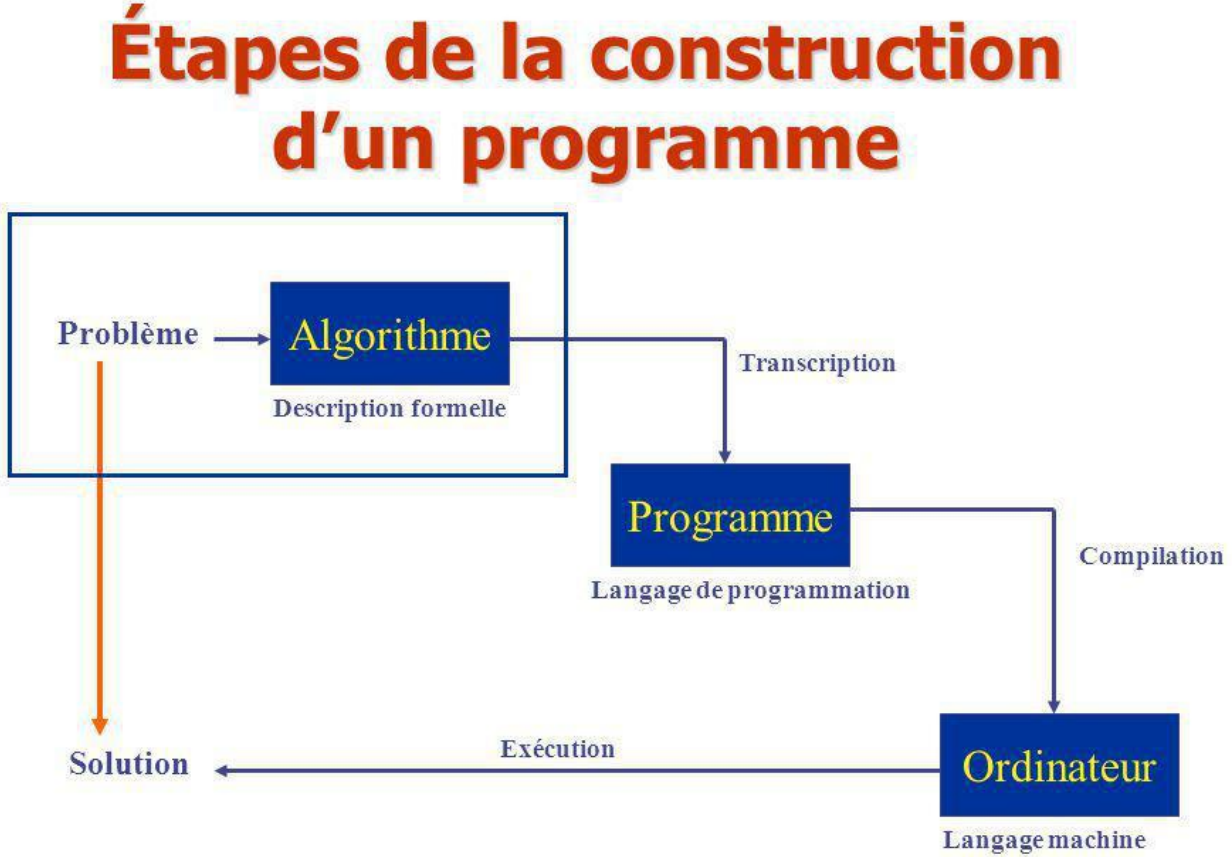
- Un algorithme est une suite finie d’instructions indiquant de façon précise l'ordre dans lequel doit être effectué un ensemble d'opérations dans le but de résoudre un problème particulier.

- L’**algorithmique** est la logique utilisée pour écrire des algorithmes.

- Il est à noter qu’un algorithme est souvent exprimé avec une notation indépendante de tout langage de programmation.

1. Définition d’un algorithme

Un algorithme devrait être traduit en un langage de programmation pour aboutir, après exécution, à une solution.



2. Structure d’un algorithme

- Deux parties sont essentielles :
  - Une partie déclarative contenant tous les objets qui seront impliqués par les différentes actions de l’algorithme (constantes, types, variables, etc.).
  - Une partie réservée aux actions (instructions), délimitée par les deux mots-clés **Début** et **Fin,**

- **Un algorithme est aussi caractérisé par son nom qui devrait être *significatif*.**

ALGORITHME nom_de_l'algorithme
CONST Définition des constantes **TYPE** Définition de types **VAR** Déclaration de variables
DEBUT Suite d'instructions **FIN**

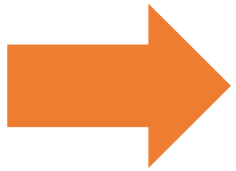
2. Structure d’un algorithme -Exemple

ALGORITHME affichageSalutation

DEBUT

Ecrire("Bonjour tout le monde")

FIN



Cet algorithme permet d'afficher la phrase **Bonjour tout le monde**

3. Les variables

- Une variable est un conteneur manipulé et modifiable par l'algorithme, c'est-à-dire que son contenu, ou sa valeur, peut être changé à tout moment.
- Elle est utilisée dès qu'on a besoin de stocker une information dans un algorithme.
- Ces informations, intermédiaires ou définitives, peuvent être des données issues du disque dur, fournies par l'utilisateur ou des résultats obtenus par un autre algorithme.
- En programmation, une variable est un espace mémoire qui va contenir des données au fur et à mesure que le programme avance dans son exécution. Cependant, à un instant donné, une variable ne peut contenir qu'une seule valeur.

3.1. Déclaration de variables

- Avant de pouvoir utiliser une variable il faut **la déclarer**, donc on doit lui attribuer un nom ou une étiquette et définir son type.

- Le nom d'une variable doit :
  - être significatif (décrit les données manipulées)
  - être différent des mots clés (telque **DEBUT, FIN, VAR**…)
  - commencer par une lettre alphabétique
  - contenir uniquement des lettres, des chiffres et du soulignement _ (pas de lettres accentuées, caractères de ponctuation ni espaces)

- Exemple : VAR *Prix_TTC : Réel*

3.2. Types de variables

- Une variable doit avoir un type.

- Un type est défini par
  - L'ensemble des valeurs que peut prendre les variables de ce type
  - L'ensemble d'opérations qu'on peut appliquer sur les variables du type

- Deux familles de types :
  - les types simples : ce sont des types qui sont supportés et reconnus par la majorité des langages de programmation (entier, réel, booléen, etc.)
  - Les types composés ou complexes : ce sont des types qui sont construits à partir des types simples mais qu'il faut les déclarer dans la partie réservée aux types (tableaux, chaînes, enregistrements, etc.)

4. Les types simples

- Quatre types simples
  - Le Type Entier
  - Le Type Réel
  - Le Type Caractère
  - Le Type Booléen ou Logique.

4.1. Le type Entier

- Le type Entier comprend un sous ensemble fini de nombres entiers, dont la taille varie en fonction des performances techniques de la machine et celles du langage de programmation utilisé.

Entier simple (2 octets)	-32768 à 32767
Entier long (4 octets)	-2147483648 à 2147483647

- Si on a plusieurs variables de même type, pas la peine de les déclarer séparément, il suffit de les regrouper en les séparant par une virgule (VAR *a, b : Entier*)

4.2. Le type Réel

- Le type Réel comprend un sous ensemble fini de nombres réels, dont la taille varie en fonction des performances techniques de la machine et celles du langage de programmation utilisé.

Réel simple	De $-3{,}40\ 10^{38}$ à $-1{,}40\ 10^{-45}$ pour les valeurs négatives
(4 octets)	De $1{,}4\ 10^{-45}$ à $3{,}4\ 10^{38}$ pour les valeurs positives
Réel double	De $1{,}79\ 10^{308}$ à $-4{,}95\ 10^{-324}$ pour les valeurs négatives
(8 octets)	De $4{,}94\ 10^{-324}$ à $1{,}79\ 10^{308}$ pour les valeurs positives

- Un nombre réel s'écrit de façon standard sous forme d'une partie entière et une partie décimale avec un séparateur qui n'est plus la virgule mais le point.
  - 7 → 7.0
  - –3,05 →-3.05

4.3. Le type Caractère

- Le type Caractère est ensemble de caractères comportant :
  - les 26 lettres alphabétiques en majuscules (de ‘A’ jusqu’à ‘Z’)
  - les 26 lettres alphabétiques en minuscules (de ‘a’ jusqu’à ‘z’)
  - les 10 chiffres (‘0’ jusqu’à ‘9’).
  - des caractères spéciaux.
- La valeur d’un caractère est toujours délimitée par deux apostrophes
- Une variable de type caractère contient à un instant donnée un et un seul caractère
- Exemple : *VAR c : Caractère*

  c ← ‘H’

4.3. Le type Caractère

- Les caractères ont un ordre d'apparition précis résumé selon le codage ASCII suivant.
- De ce fait chaque caractère à un prédécesseur et un successeur

Opération	Symbole
Successeur ou Suivant	Succ ou Suiv
Prédécesseur ou Précédent	Pred ou Prec

- Exemple : x ← 'C' ⟹ Suiv(x) revoie 'D'

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
32	20	[SPACE]	64	40	@	96	60	`
33	21	!	65	41	A	97	61	a
34	22	"	66	42	B	98	62	b
35	23	#	67	43	C	99	63	c
36	24	$	68	44	D	100	64	d
37	25	%	69	45	E	101	65	e
38	26	&	70	46	F	102	66	f
39	27	'	71	47	G	103	67	g
40	28	(	72	48	H	104	68	h
41	29	)	73	49	I	105	69	i
42	2A	*	74	4A	J	106	6A	j
43	2B	+	75	4B	K	107	6B	k
44	2C	,	76	4C	L	108	6C	l
45	2D	-	77	4D	M	109	6D	m
46	2E	.	78	4E	N	110	6E	n
47	2F	/	79	4F	O	111	6F	o
48	30	0	80	50	P	112	70	p
49	31	1	81	51	Q	113	71	q
50	32	2	82	52	R	114	72	r
51	33	3	83	53	S	115	73	s
52	34	4	84	54	T	116	74	t
53	35	5	85	55	U	117	75	u
54	36	6	86	56	V	118	76	v
55	37	7	87	57	W	119	77	w
56	38	8	88	58	X	120	78	x
57	39	9	89	59	Y	121	79	y
58	3A	:	90	5A	Z	122	7A	z
59	3B	;	91	5B	[	123	7B	{
60	3C	<	92	5C	\	124	7C	|
61	3D	=	93	5D	]	125	7D	}
62	3E	>	94	5E	^	126	7E	~
63	3F	?	95	5F	_	127	7F	[DEL]

4.4. Le type Booléen

- Le type booléen , appelé aussi logique, est un ensemble constitué de deux éléments dont la valeur de l'un contredit celle de l'autre.

- Ces valeurs sont représentées par
  - **Vrai**
  - **Faux**

- Exemple : VAR *admis : Booléen*

  admis ← *vrai*

5. Les opérateurs

- Les opérateurs sont des symboles qui permettent d'exécuter des opérations dans un algorithme, on distingue les suivants :
- **Opérateurs arithmétiques :** addition (+), soustraction (-), multiplication (*) et division (/), division entière (div), reste de la division entière (mod ou %) et puissance (^).
- **Opérateurs de comparaison :** strictement inférieur (<), strictement supérieur (>), inférieur ou égal (<=), supérieur ou égal (>=), égal (=) différent (<>)
- **Opérateurs logiques :** ET, OU et NON

Et	vrai	Faux
Vrai	vrai	Faux
Faux	faux	Faux

Ou	vrai	faux
vrai	vrai	vrai
faux	vrai	faux

Non	
vrai	Faux
faux	Vrai

- **Opérateur d'affectation ou d'assignation :** ← sert à modifier la partie gauche de la flèche en lui attribuant la valeur de la partie à droite

5. Les opérateurs

Exemple :

i ←1	*i reçoit la valeur 1*
j ←i	*j reçoit la valeur de i*
k ←i+j	*k reçoit la somme de i et j*
OK ←FAUX	*OK booléen qui reçoit FAUX*
r ←7.2	*r est un réel*
r1 ←r	*r1 reçoit le contenu de r*
*A ← 5*x+3*B*	*A reçoit le résultat de l'expression (5*x+3*B)*

MERCI POUR VOTRE ATTENTION