ALGORTHMIQUE : Notions de base

1) DEFINITIONS.

Algorithme : Description en langage naturel de la suite des actions effectuées par un programme.

Syntaxe : Règles d'écriture d'un langage donné.

Type de données :

Un programme peut être amené à manipuler différents types de données :

- **booléen** : valeur pouvant être soit Vraie, soit Fausse.
- **entiers** : valeur numériques entières pouvant être signées ou non signées (codées sur un ou plusieurs octets).
- **réels** : valeurs numériques codées avec une mantisse et un exposant.
- **caractère** : octet correspondant à un code ASCII.
- **chaîne de caractères** : ensemble de caractères.
- **tableau de données** : ensemble de données de même type (exemple : tableau d'entiers, tableau de réels).

Toutes ces données sont codées sous forme d'octets en mémoire.

Constante : donnée manipulée par un programme et ne pouvant être modifiée.

Exemple : Constante Pi = 3.141559

Variable : donnée manipulée par un programme et pouvant être modifiée. Ce peut être :

- une donnée d'entrée ;
- le résultat final d'un calcul ;
- un résultat intermédiaire de calcul.

Identificateur : nom explicite d'une constante, d'une variable ou d'une fonction.

Exemples : Conversion_BCD, Resultat, Lettre…

2) ORGANISATION D'UN PROGRAMME.

L'algorithme d'un programme est organisé en plusieurs parties :

```
ALGORITHME nom_de_l'algorithme
CONST {Définition des constantes}
VAR {Déclaration de variables}
DEBUT
    {Suite d'instructions}
FIN
```

2.1) Déclaration des constantes

Syntaxe : **Constante** NomConstante = Valeur

Exemples : **Constante** Pi = 3.141559

2.2) Déclaration des variables

Syntaxe : **Variable** NomVariable : [**Type**]

Exemples : **Variable** Rayon : Reel
Variable Compteur : Entier
Variable Lettre : Caractere

3. Les structures simples

3.1. L'affichage : ECRIRE

Cette action permet d'afficher un résultat ou un message sur écran ou sur imprimante pour l'utilisateur.
Syntaxe ECRIRE (paramètre1 [[, paramètre2]…])

3.2. La saisie des données : LIRE

L'action LIRE permet à l'ordinateur d'acquérir des données à partir de l'utilisateur, dans des cases mémoire bien définies (qui sont les variables déclarées).
Syntaxe LIRE (variable1 [[, variable2] …])

3.3. AFFECTATION.

Une affectation consiste à attribuer une valeur à une variable.

La syntaxe générale est la suivante : NomVariable ← Expression

« Expression » peut être :

une constante	surface ← 40
une autre variable	Donnee ← ValeurMemorisee
le résultat d'une fonction	resultat ← racine (nombre)
un calcul portant sur ces différents éléments	surface ← (PI * Carre (Diametre)) / 4

4. Operateurs - conditions

4.1 Opérateurs

Les opérateurs permettent d'élaborer une expression en vue d'effectuer un calcul ou une comparaison.

L'usage des parenthèses est vivement conseillé dans le cas d'expressions complexes.

Nature	Variables utilisées	Notation	Signification
Opérateurs arithmétiques	Entier Réel	+	Addition
		-	Soustraction
		*	Multiplication
		/	Division (réelle)
		DIV	Division entière
		MOD	Reste de la division entière
Opérateurs logiques	Booléen Entier	et	Fonction ET
		ou	Fonction OU
		ouex	Fonction OU EXCLUSIF
		non	Fonction NON
Opérateur de concaténation	Chaîne de caractères	+	Concaténation
Opérateurs de comparaison	Booléen Entier Réel Caractère Chaîne de caractères	=	Egal
		≠	Différent
		<	Inférieur
		>	Supérieur
		≤	Inférieur ou égal
		≥	Supérieur ou égal

4.2. Conditions

Dans les structures algorithmiques qui vont suivre, le terme « Condition » peut représenter :

Une condition simple : Ex : x ≠ 0 Indice ≤ 80

Une condition complexe : Ex : (x > 0) ET ((y > 0) OU (z > 0))

(Indice ≥ 1) ET (Indice ≤ 10) ~ pour 1 ≤ Indice ≤ 10 ~

5) LES STRUCTURES ALGORITHMIQUES.

Les structures algorithmiques sont réparties en 3 catégories :

- succession linéaire d'opérations;
- structures conditionnelles ou de choix : en fonction d'une condition, le programme exécute des opérations différentes;
- structures itératives ou répétitives: sous contrôle d'une condition, une séquence d'opérations est exécutée répétitivement.

5.1. Séquencement linéaire

Les actions successives sont mentionnées les unes après les autres.

Syntaxe Action1 Action2 ... ActionN

Remarque : dans la suite, la notation «Actions » ou «ActionsN » représentera une succession d'actions comme ci-dessus.

5.2. Structures de choix (ou conditionnelles)

5.2.1. STRUCTURE SI ... ALORS ...

Une condition est testée pour déterminer si l'action ou le groupe d'actions suivant doit être exécuté.

Syntaxe **Si** Condition **Alors** Actions **Fin Si**

5.2.2. STRUCTURE SI ... ALORS ...SINON ...

Une condition est testée pour déterminer quelle action ou quel groupe d'actions doit être exécuté.

Syntaxe

Si Condition
Alors Actions1
Sinon Actions2
Fin Si

5.3. Structures itératives (ou répétitives)

5.3.1. STRUCTURE REPETER ... JUSQUA ...

Une action ou un groupe d'actions est exécuté répétitivement jusqu'à ce qu'une condition soit vérifiée.

Syntaxe

Répéter
Action(s)
Jusqu'à condition

Remarque : la vérification de la condition s'effectue **après** les actions. Celles-ci sont donc exécutées au moins une fois.

5.3.2. STRUCTURE TANT QUE ... FAIRE ...

Une action ou un groupe d'actions est exécuté répétitivement tout le temps où une condition est vraie.

Syntaxe

Tant Que Condition **Faire** actions **Fin tant que**

Remarque : la vérification de la condition s'effectue **avant** les actions. Celles-ci peuvent donc ne jamais être exécutées.

5.3.3. STRUCTURE POUR INDICE DE ... A .... FAIRE ...

Une action ou un groupe d'actions est exécuté répétitivement un certain nombre de fois : le nombre dépend des valeurs initiale et finale données à la variable « Indice ».

Syntaxe **Pour** Indice **De** Valeur1 **A** Valeur2 **Faire**
Actions
Finpour

Remarque : les valeurs initiale (Valeur1) et finale (Valeur2) sont comprises. Il est éventuellement possible de spécifier un autre pas d'incrémentation (+2, +10, -1....)