**Algorithmique**

Table des matières

[Introduction 2](#_Toc91332779)

[Chapitre1 : Structure de données : 2](#_Toc91332780)

[1- Variables : 2](#_Toc91332781)

[2- Expressions : 3](#_Toc91332782)

[3- Type de base 3](#_Toc91332783)

[Chapitre 2 : Structure de contrôle : 3](#_Toc91332784)

[1- Alternative : 3](#_Toc91332785)

[2- Boucle-Loop : 5](#_Toc91332786)

[Annexes : 5](#_Toc91332787)

# 

# Introduction

Un programme a la particularité d’avoir une syntaxe.

La fonction : est un bloc d’instructions paramétrables avec une fonctionnalité limitée.

La programmation est l’implémentation ce qui signifie la traduction de la solution algorithmique dans un langage informatique.

Programmer → Implémenter

On note l’existence de 2 catégories de langages :

1. Le langage compilé : le fichier source est traduit pour générer un fichier exécutable
2. Le langage interprété : la traduction du ficher source se fait en temps réel au moment de l’exécution.

Chaque programme doit être vérifier avant sa mise en fonction afin de détecter les anomalies, dans le cas contraire des incidents risquent de surgir et causer des dégâts financiers juridique ou de sécurité...

Dans la phase de conception il est plus productif de savoir avec quel langage le programme sera conçu, cela facilitera plus la conception de l’algorithme, le fait de savoir avec quel langage de programmation on va travailler, cela va guider la logique de l’algorithme et l’alléger d’une certaine façon.

L’algorithme est une suite logique d’itérations ou actions qui permet de répondre à un problème ou tout simplement d’accomplir une tâche (une recette de gâteau).

L’algorithme est nécessaire pour concevoir un programme quelque soit le langage utilisé.

L’algorithme est une chaine de caractères alphanumérique, de signes opératoires (+, -, \*,/, …. ).

Dans la syntaxe de l’algorithme le symbole = désigne une affectation et le symbole == désigne égalité. Il existe aussi des mots réservés qu’on n’a pas le droit d’utiliser pour leur donner une autre signification que celle déjà donnée par le programme ( lire, écrire, Var….). Ainsi que des signes de ponctuation ( DEBUT,FIN,=, ;…..

# Chapitre1 : Structure de données :

Une structure de données permet de stocker la donnée et de l’organiser pour faciliter son traitement et sa manipulation (l’exploiter et la modifier)

Dans un algorithme on doit systématiquement définir les variables.

## 1- Variables :

En algorithme la viable est une triplée qui est composée de :

* Un nom : choisi selon la nature de la variable, il faut qu’il soit cohérent et significatif.
* Une valeur : changeante.
* Un type : renvoie au type de donnée qu’on peut stocker, il est prédéterminé (chaine de caractères, entier, caractère, booléen, réel)

|  |
| --- |
| Var NomDeVariable : Type ; |

## 2- Expressions :

* L’expression est la formule composée de l’ensembles de variables déclarées, de leurs valeurs, et de l’opération de type de variable en question.
* L’affectation : c’est une expression pour stocker une valeur dans une variable

|  |
| --- |
| NomDeVariable = ExpressionDuTypeDeLaVariable ; |

## 3- Type de base

* Booléens : la variable booléenne prend 2 valeurs (vrai, faux) ou (0,1), elle est utilisée dans les tests et dans toutes les expressions de comparaison.

La relation peut être binaire « ET » « OU », ou bien unaire « NON ».

* Entiers : prend comme valeur tous les nombres entiers avec signe « +, -», les opérations + - \*/.
* Réels :  prend comme valeur tous les nombres réels, les opérations + - \*/.
* Chaine de caractères : prend comme valeur tous les caractères imprimables, celles-ci doivent être entre guillemets.
* Remarque : ‟ 1 ″  représente un caractère qui est convertible en nombre ;

1 est un nombre entier ;

1. Est un nombre réel ;

→ Représentent des types de caractères différents.

→ Une chaine de caractères nombre est convertibles en nombre.

* Comparaison : est associée aux opérations <, ≤, ==, =, ≥, >, qui permettent de comparer les valeurs de type : entiers, réel, caractère et booléens.

→ Le résultat de cette comparaison est une valeur booléenne.

# Chapitre 2 : Structure de contrôle :

## 1- Alternative :

* **Séquence :** Les instructions s’exécutent l’une après l’autre sans interruption ou conditions.
* Faire traitement 1 ;
* Faire traitement 2 ;
* ………. ;
* Faire traitement N ;
* **Alternative simple :** SI

La condition est une expression booléenne et logique (vrai, faux, <,>,==)

Les instructions ne sont exécutées que si la condition est vraie

SI (CONDITIONS) ALORS

INSTRUCTIONS 1

INSTRUCTIONS 2

……..

INSTRUCTIONS N

FINSI

* **Alternative simple :** SI…..alors SINON

Les 2 instructions ne peuvent être exécutées à la fois, mais seulement l’une des 2.

SI (CONDITION) ALORS

BLOC INSTRUCTIONS 1

SINON

BLOC INSTRUCTIONS 2 FINSI

* **Le SWITCH :** Il rend l’algorithme plus lisible, il comprend tous les types de variable ( entier, booléenne..)

Le break dans le switch permet d’arrêter l’exécution des instructions du cas suivant quand le cas précédent est réalisé. Si on ne met pas le break, tous les cas seront exécutés.

SELON QUE ( VariableDeChoix )

CAS val 1 :

BLOC INSTRUCTIONS

Break ;

CAS val 2 :

BLOC INSTRUCTIONS

Break ;

CAS val 3 :

BLOC INSTRUCTIONS

Break ;

….

AUTREMENT : BLOC INSTRUCTIONS

FINSELONQUE

## 2- Boucle-Loop :

La boucle est une instruction ou un bloc d’instructions qui se répètent un nombre donné de fois, elle permet d’éviter de réécrire la même chose avec des données différentes.

On peut distinguer 3 différentes boucles :

* **Pour …..Faire……FinPour :** Cette boucle se caractérise par :
* Le compteur s’incrémente automatiquement
* On sait par avance combien d’itérations qui seront effectuées
* Le pas par défaut est égal à 1

POUR compteur ALLANT DE 1 à 10 FAIRE

BLOC INSTRUCTIONS

FINPOUR

* **TantQue……Faire……FinTantQue :** Cette boucle contient une condition booléenne, et la variable impliquée dans la boucle doit être modifiée dans la boucle, si non soit ça ne rentre jamais ou ça ne sort jamais.

TANTQUE (CONDITION) FAIRE

BLOC INSTRUCTIONS

FINTANTQUE

* **Faire…….TantQue**: la particularité de cette boucle est que le bloc d’instructions s’exécute aumoins une fois

FAIRE

BLOC INSTRUCTIONS

TANTQUE (CONDITION)

* Exemples :
* Somme de deux nombres :

**Algo Somme**

Var x,y,z : entiers

Début

Afficher ( « entrer la valeur de x et y »)

Lire (x,y)

Z= x+y

Afficher ( « somme : » z)

Fin

* Le maximum

**Algo Max**

Var x, y : entier

Début

Afficher (« entrer une valeur de x et y »)

Lire (x,y)

SI (x > y) ALORS

Afficher (« le max est : » +x)

SINON

SI (x < y) ALORS

Afficher (« le max est : » +y)

FINSI

SINON

Afficher (« Les deux nombres sont égaux »)

FINSI

Fin

# Annexes :

**1-Résoudre une équation de premier degré :**

Equation de premier degré : a\*x+b =0 → a\*x=-b

Si a =0 on écrit l’équation comme suit : 0\*x=-b

Si b=0, on écrit l’équation comme suit : 0\*x=0

L’équation est indéterminée

Si b ≠0 on écrit l’équation comme suit : 0\*x= -b

Qui est impossible

Si a≠0 x=-b/a

**1-Algo EquationPremierDegré :**

Var a, b, x : réel

Début

Ecrire (« entrez la valeur de a et b»)

Lire (a,b)

Ecrire (« Equation » + a + « x+ » + b + « = 0 »)

SI (a==0) alors

si (b !=0 ) alors

Ecrire ("Equation impossible »)

SINON afficher "Equation impossible."

SI non

Ecrire (« Infinité de solutions »)

FINSINON

SI NON

X= -b/a

Ecrire (« la solution est : » + x)

FINSINON

Fin

**2-Calcul du factorielle :**

N ! = 1\*2\*3\*4………\*N

0 != 0

**2-AlgoFactirielle**

Var (FACT, N, i) : entier

Début

Afficher (« entrer la valeur de N > 0 »)

Lire (N) SI ( N<0) alor

Ecrire ( «  valeur inconnue »)

exit

Si (N==0) alors

Afficher (« n != »+1)

SINON

Fact=1

Pour i allant de 2 à n faire

Fact= fact\*i

Finpour

Afficher (n+ “ != “+ fact)

Fin

**3-Calcule de la TVA**

TVA = 20%

Montant TTC=HT + (HT\* TVA%) → Montant HT = TTC-(HT\*TVA%) → HT= TTC/1,2

**3-AlgoTVA :**

Var mttc, mht, montant : réel

Nature booléen// prend 2 valeurs soit ht ou ttc

Début

Afficher (« saisissez  un montant positif»)

Lire (x)

TVA= 0,2

Afficher (« est ce que la valeur de x est-elle soumise à la TVA ? »)

SI oui alors

x= x\*0,2

Afficher (« le montant TTC : » x)

Sinon x = x/1,2

Afficher (« le montant HT : » x)

Finsi

Fin