Sommaire

L'ALGORITHMIQUE

Définition

Rappelons ce qu'est un traitement : un procédé qui permet de passer de données (entrées) aux résultats (sorties).

L'***algorithmique*** est la technique des algorithmes. Un ***algorithme*** est la description d'un traitement selon une logique et un formalisme rigoureux. L'algorithme prépare la programmation (les langages de programmation ont des syntaxes similaires à celle des langages algorithmiques).

L'algorithme manipule des données, que l'on nomme ***variables***, décrit des opérations de calcul ou de transfert de données et s'articule grâce à des ***ordres de contrôle*** (faire ceci à telle condition, recommencer, …). Par similitude avec les programmes, les opérations et les ordres de contrôle qui composent un algorithme sont souvent appelés ***instructions***.

Un algorithme est formé d'instructions types combinées entre elles.

Les deux présentations courantes des algorithmes sont l'***organigramme logique*** et le ***langage algorithmique*** : c'est cette seconde forme que nous étudierons.

La structure générale d'un algorithme

Classiquement, un algorithme comprend quatre sections :

- la ***déclaration des variables*** (déclaration des noms attribués aux variables et de leur type) et des ***procédures*** ou des ***fonctions*** externes (des « sous-traitements » de portée générale qui sont réutilisés dans plusieurs algorithmes différents)
- l'***initialisation des variables***, c'est-à-dire l'affectation de valeurs initiales aux variables déclarées (ce qui est impératif pour que l'algorithme puisse fonctionner)
- le traitement proprement dit
- la sortie des résultats et l'arrêt de l'algorithme

La pureté de la structure algorithmique, notamment dans la partie traitements, est la condition d'une bonne lisibilité et, surtout, d'un bon fonctionnement de l'algorithme.

Exemple de structure

```
Algo moy :
        Annonce la variable (entier réel, booléen, texte, date, monétaire)
Var nb1, nb2 : entier                    Entête
            Moyenne : reel
Debut
        SAISIR « entrez la valeur de la 1ère note : », nb1
        SAISIR « entrez la valeur de la 2ème note : », nb2          Corps
        Moyenne ← nb1+nb2/2
        Afficher « La moyenne est de : », moyenne
FIN             Finalisation
```

La déclaration et l'affectation de valeurs

La ***déclaration de variable*** peut adopter la syntaxe suivante :

<type> nomvar1, nomvar2, …, nomvar3

La variable peut être simple ou matricielle (tableau indicé).

Exemples : *entier* *coefficient, n, i, ventes-mensuelles(12), ventes-mois-magasin(12,236)*

décimal tarif, taux-tva
booléen accord
date dt-vente, dt-achat, jours-ouvrables(260)
réel taux-moyen
texte désignation-produit

« ventes-mensuelles » désigne une liste de 12 valeurs repérées par leur rang, soit ventes-mensuelles(1), ventes-mensuelles(2), ventes-mensuelles(3),… ventes-mensuelles(12), en général ventes-mensuelles(i) pour i variant de 1 à 12.
Booléen signifie à deux valeurs possibles (0,1 ou OUI, NON)

L'***affectation de valeur*** à une variable peut se représenter de deux façons :

var ← valeur var := valeur ou var = valeur

Les deux premières notations sont souvent utilisées pour distinguer l'affectation de valeur de l'égalité mathématique. En effet, dans un algorithme, la valeur peut être une autre variable (a ← b signifie que l'on transfère la valeur de b dans a), ou la variable elle-même (a ← a+1 signifie que l'on ajoute 1 à l'ancienne valeur de a qui est « incrémentée »). L'usage du signe « = » conduit dans ce dernier cas à écrire a = a+1 (ce qui peut surprendre en première lecture).

Remarque : pour des raisons de simplicité, nous utiliserons la forme « = ».

La ***déclaration*** d'une ***fonction*** ou d'une ***procédure*** peut se faire de la manière suivante :

fonction nomfonction (par1, par2, …, parn)
procédure nomprocédure (res, par1, par2, …, parn)

par1, par2, … étant des variables déclarées.

Exemples :

procédure marge-HT (achat-HT, vente-TTC, taux-tva).

La procédure est appelée simplement en la citant dans l'algorithme : marge-HT (achat-HT, vente-TTC, taux-tva). Elle fonctionne à l'aide des valeurs prises par les variables achat-HT, vente-TTC et taux-tva au moment où elle s'exécute et calcule res=vente-TTC/(1+taux-tva) – achat-HT. La valeur obtenue pour res se retrouve dans l'algorithme qui a appelé la procédure.

fonction marge-HT (achat-HT, vente-TTC, taux-tva)

La fonction prend la valeur du résultat. Dans ce cas, il faudrait donc écrire res=marge-HT(achat-HT, vente-TTC, taux-tva) dans l'algorithme.

Les fonctions sont le plus souvent les fonctions mathématiques usuelles et sont implémentées directement dans les logiciels. Elles seront alors reprises comme telles dans les algorithmes.

Exemple : *x = sqrt(taux) - racine carrée ou « square root »*
y = sin(x)
m=max(ventes-mensuelles)
a=ent(x) –partie entière de x

Les entrées-sorties

Pour ***lire*** ou ***écrire*** des données depuis ou sur un support informatique (disque, bande, ligne de transmission, papier, …), on écrit :

lire var1, var2, ……, varn
ecrire var1, var2, ……, varn

var1, var2, …. étant des variables déclarées.

Exemples : *lire tarif, taux-tva*
écrire ventes-mensuelles (ce qui signifie écrire les 12 valeurs à la suite)
écrire ventes-mensuelles(3) (écrire la troisième valeur uniquement)

La structure alternative

La ***structure alternative*** permet de mener deux actions différentes selon une condition. La condition peut être simple ou combinée (grâce aux opérateurs logiques ET, OU). Les alternatives peuvent s'imbriquer. L'expression de l'alternative est la suivante :

```
SI <condition>
        ALORS <action condition vraie>
        SINON <action condition fausse>
FINSI
```

Exemples :

```
décimal seuil1, seuil2, taux1, taux2
seuil1=1000
seuil2=3000
taux1=0,05
taux2=0,1
lire achats
SI achats>seuil2
        ALORS remise=achats X taux2
        SINON   SI achats<seuil1
                        ALORS remise=0
                        SINON remise=achats X taux1
                FINSI
FINSI
Ecrire remise
```

L'exemple montre l'importance de la présentation des algorithmes pour leur lisibilité (ici décalage des niveaux de traitement, ou « indentation »), la possibilité d'imbriquer des alternatives et le rôle essentiel de la clôture « FINSI » pour les délimiter.

```
décimal a, b
lire a, b
SI A>0 OU B>0
        ALORS écrire " du positif "
        SINON écrire " tout négatif "
FINSI
```

Cet exemple illustre la possibilité de combiner les conditions, celle d'écrire un résultat à l'intérieur d'une alternative (ou d'une répétitive), et la distinction des valeurs texte qui sont encadrées de guillemets.

Remarque : la structure alternative est fréquente dans les calculs sous tableur, sous la forme =SI(condition ; actionsioui ; actionsinon).

La structure répétitive

La ***structure répétitive*** indique qu'une action doit être répétée. Elle s'exprime ainsi :

```
TANTQUE <condition>
        <actions>
FINTANTQUE
```

Exemple :

```
entier i, valeurs(10)
i=1
TANTQUE i<11
        LIRE valeurs(i)
        I=i+1
FINTANTQUE
ECRIRE valeurs
```

Cet exemple combine une répétitive et l'usage d'un tableau indicé pour lire 10 entiers et les écrire.

Répétitive et alternatives peuvent se combiner en respectant la règle mathématique d'ouverture – fermeture des parenthèses : les mots SI et TANTQUE ouvrent des parenthèses, FINSI et FINTANTQUE les ferment.

Exemple :

```
décimal tarif, taux1, taux2
texte code, désignation
entier seuil
**** un article du fichier comprend code, désignation, prix ****
LIRE seuil, taux1, taux2
TANTQUE il reste un article
        LIRE code, désignation, prix
        SI prix <seuil
                ALORS prix=prix X (1+taux1)
                SINON prix=prix X (1+taux2)
        FINSI
        ECRIRE code, désignation, prix
FINTANTQUE
```

*Dans cet algorithme, le texte entre **** est un commentaire sans effet sur le déroulement du traitement. On y lit successivement tous les articles d'un fichier pour mettre à jour le tarif selon un seuil, puis on réécrit ces articles. La fin du TANTQUE provoque automatiquement le passage à l'article suivant dans le fichier.*

Remarque : d'autres formes de répétitives sont parfois utilisées (on les retrouvent dans les langages de programmation). Par exemple POUR …. FINPOUR, FAIRE …. JUSQU'A.