Chapitre 2 : Algorithme séquentiel simple

Dans ce chapitre on va représenter en générales comment créer un algorithme (programme C) à l'aide des opérations séquentielles.

1. Structure générale d'un algorithme

La structure générale d'un algorithme est comme suite :

Algorithme nom_de_l_algorithme // L'en-tête
Const Liste_des_constantes
Type Liste_des_types
Var Liste des variables // Les déclarations
Liste des procédures et fonctions
Début
    Instruction 1
    Instruction 2
Instruction N // Les traitements
Fin

} Le corps de l'algorithme

1.1. Entête

Permet d'identifier l'algorithme. C'est-à-dire, elle sert à lui donner un nom.

1.2. Déclarations :

C'est une liste de tous les objets (constantes, variables, ...) utilisés et manipulés dans le corps de l'algorithme.

1.3. Corps :

Dans cette partie sont placées les opérations et les instructions à exécuter.

Exemple : L'algorithme qui permet de permuter (échanger) les valeurs de deux nombres entiers X et Y. C'est-à-dire la valeur de X va à Y et la valeur de Y va à X.
Exemple : Si, au départ, X= 10 et Y=17, alors à la fin de l'algorithme on aura X=17 et Y=10.

```
Algorithme Permutation
Var X, Y, Z : Entier
Début
Lire( X, Y )
Z ⇐X
X⇐ Y
Y ⇐ Z
Ecrire ( X, Y )
Fin
```

La traduction de l'algorithme Permutation en programme C est donnée ci-dessous :

```
// Programme qui permute les valeurs de deux entiers X et Y.
#include <stdio.h>
int main( )
{
        int X, Y, Z ;
        scanf("%d", &X) ;
        scanf("%d", &Y) ;
        Z = X ;
        X = Y ;
        Y = Z ;
        printf("X = %d et Y = %d\n", X, Y) ;
        return 0 ;
}
```

- **Variables et constantes**

Définition d'une variable

Une variable est un espace mémoire identifié par un nom, destiné à stocker une valeur, qui peut être modifiée durant les traitements. Les variables d'un algorithme contiennent les informations nécessaires à son déroulement.

Définition d'une constante une constante ne prend qu'une unique valeur au cours de l'exécution de l'algorithme.

Définition d'un identificateur

Un identificateur (identifiant) est un nom qui respecte une syntaxe particulière :

- ✓ Il est constitué d'une suite de lettres de l'alphabet (latin) et de chiffres.
- ✓ Il commence, obligatoirement, par une lettre.
- ✓ Le caractère souligné "_" (Le tiret bas, underscore) peut jouer le même rôle qu'une lettre de l'alphabet. Sur le clavier, le caractère "_" se trouve sur la même touche qui permet d'écrire le chiffre 8.

Exemples : x, y, PI, rayon_du_cercle, _x1, _y2, ...

Remarques

✓ Pour faciliter la lisibilité d'un algorithme, il est préférable d'utiliser des noms significatifs.

✓ L'identificateur doit être différent de tous les mots clés (Début, Fin, Si, Alors, Sinon, .).

✓ Les caractères de l'alphabet grec sont interdits. Si on a besoin d'utiliser a, p, S, ou A, on doit les écrire en alphabet latin comme alpha, béta, gamma, delta et ainsi de suite.

✓ Il est interdit d'utiliser des caractères accentués (é, è, î, ç, ', ", .) dans un identifiant.

✓ Il est interdit d'utiliser des indices ou des exposants.

Exemples :

ne pas utilser x_1, x_2 mais plutôt utiliser x1 et x2.

ne pas utilser x^1, x^2 mais plutôt utiliser x1 et x2 ou Xp1 et Xp2.

- **Le type :**

Le type correspond au genre d'information utilisé. Les types standards (prédéfinis) en langage algorithmiques sont : Entier, Réel, Booléen, Caractère, Chaîne (de caractères). Chaque type est muni (dispose) d'un ensemble d'opérations.

Syntaxe de déclaration des constantes

CONST nomConstante1 **=** Valeur1
nomConstante2 **=** Valeur2

nomConstanteN **=** ValeurN

Exemples :
```
Const     Nombre = 15
          Pi = 3.14          // Constante de type ENTIER
          Rep = Vrai         // Constante de type REEL
          Reponse = 'N'      // Constante de type BOOLEEN
          Univ = "Université de Jijel"   // Constante de type CARACTERE
                             // Constante de type CHAINE
```

Remarque : Pour déclarer une constante on définit son nom et sa valeur.

Syntaxe de déclaration des variables

VAR nomVariablel, nomVariableNI : Type1 nomVariable2, ..., nomVariableN2 : Type2
nomVariableM, ..., nomVariableNM : TypeM

Exemples :

```
Var   A, B,  C, D : Entier// Déclaration de 4 variables de type   Entier.
      X, Y, Z : Réel       // Déclaration de 3 variables de type   Réel.
      Stop  : Booléen      // Déclaration d'une variable de type   Booléen.
      Reponse : Caractère  // Déclaration d'une variable de type Caractère.
      Nom, prenom : Chaine // Déclaration de 2 variables de type Chaine.
```

Remarque : Pour déclarer une variable on définit son nom et son type.

Les types

Le type correspond au genre d'information utilisé. Les types standard (prédéfinis) en langage algorithmiques sont : Entier, Réel, Booléen, Caractère, Chaîne (suite de caractères). Chaque type est muni (dispose) d'un ensemble d'opérations.

A. Le type Entier

Le type Entier est utilisé pour manipuler des nombres entiers : -17, -8, 0, 13, 22, .
Le type entier est muni des opérateurs suivants :

✓ Les opérateurs arithmétiques: + (addition), - (soustraction), * (multiplication).

✓ La division entière, notée **DIV** , tel que n div p donne **la partie entière du quotient** de la division entière de n par p. Exemple : 23 Div 5 = 4.

✓ Le modulo, noté **MOD**, tel que n mod p donne **le reste** de la division entière de n par p. Exemple : 23 Mod 5 = 3.

Les opérateurs de comparaison classiques : <, < =, > >

D'autres opérations sont définies par des fonctions dont on peut citer :

- **abs**(n) : Cette fonction fournit la valeur absolue de l'entier n.

B. Le type Réel

Le type Réel est utilisé pour manipuler des nombres réels : -3.7, -1.5, 0, 3.0, 1 Le type Réel est muni des opérateurs suivants :

Les opérations arithmétiques classiques : + (addition), - (soustraction), * (multiplication), / (division).

Les opérateurs de comparaison classiques : <, < =, >

D'autres opérations sont définies par des fonctions dont on peut citer :

- **trunc(x)** : Cette fonction donne la partie entière du nombre réel x.
- **round(x)** : Cette fonction donne l'entier le plus proche du nombre réel x.
- **abs(x)** : Cette fonction donne la valeur absolue du nombre réel x.
- **sqrt(x)** : Cette fonction donne la racine carrée du nombre réel x.
- Les fonctions trigonométriques : **sin(x)**, **cos(x)**, **arctg(x), ...**

Remarques

On peut affecter à une variable de type réel une autre variable de type entier. C'est la machine qui prend en charge la conversion de la valeur entière en valeur réelle.

Les variables de type entier et les variables de type réel peuvent être combinées dans des opérations définies sur les réels. Les entiers sont alors automatiquement convertis en réels.

C. Le type Booléen

Il s'agit du domaine dont les seules valeurs sont ***VRAI*** ou ***FAUX.*** Les opérateurs booléen (logiques) définis et les plus utilisés sont : le **NON**, le **ET** et le **OU**. Ils sont définis par les

tables de vérité ci-dessous.

Soient A et B deux variables de type Booléen.

A	NON A
Faux	Vrai
Vrai	Faux

A	B	A ET B
Faux	Faux	Faux
Faux	Vrai	Faux
Vrai	Faux	Faux
Vrai	Vrai	Vrai

A	B	A OU B
Faux	Faux	Faux
Faux	Vrai	Vrai
Vrai	Faux	Vrai
Vrai	Vrai	Vrai

D. Le type Caractère

Le type caractère est un ensemble fini et totalement ordonné de caractères (symboles). Il comporte :

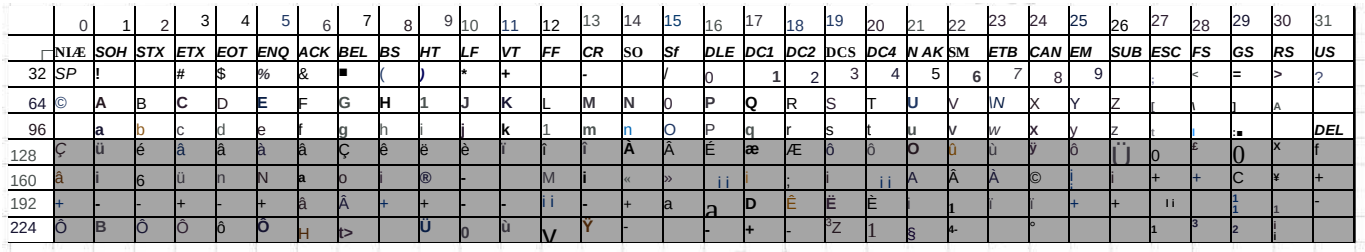
- ✓ Les lettres de l'alphabet latin : a .. z, A .. Z.
- ✓ Les chiffres : 0 .. 9.
- ✓ Les symboles utilisés en tant qu'opérateurs : + - * / < = > ...

Les caractères de ponctuation : . , ; ! ?

Les caractères spéciaux : @ % & # .

Et d'autres.

La figure ci-dessous montre la table ASCII comportant les 256nt numérotés caractères. Ils sot de 0 à 255. Chaque caractère est stocké en mémoire d'un ordinateur sur un octet.



Pour connaître le rang d'un caractère ('A' par exemple) dans l'ensemble des caractères représenté par cette table, il faut faire la somme des deux nombres se trouvant sur la même ligne (dans la première colonne à gauche) et la même colonne (dans la première ligne en haut).

Exemple : Le rang (l'ordre) du caractère 'A' est égal à 64 + 1 = 65.

Une constante de type caractère est représentée par un et un seul caractère encadré par deux apostrophes simples. Exemples : ',a', 'b', 'C', '!', '#', ...

Les opérations définies par des fonctions sur le type Caractère sont :

✓ **ORD**(c) : Cette fonction renvoie un entier positif correspond au rang du caractère c, dans l'ensemble des caractères. C'est le numéro (le rang) du caractère c.

Exemples : Ord('!') = 33, Ord ('A') = 65, Ord ('a') =97.

✓ **CHR**(i) : C'est la fonction inverse de Ord. Pour un entier positif i, elle renvoie le caractère de rang i. Exemples : Chr(33) = '!', Chr(65) = 'A', Chr(97) = 'a'.

✓ **SUCC**(c) : Cette fonction fournit le caractère qui suit immédiatement le caractère c dans l'ensemble des caractères. Exemples : Succ('a') = 'b', Succ('3') = '4', Succ('%') = '&'.

✓ **PRED**(c) : Cette fonction fournit le caractère qui précède immédiatement le caractère c

Dans l'ensemble des caractères. C'est la fonction inverse de Succ.

Exemples : Pred('b') = 'a', Pred ('4') = '3', Pred ('&') = '%'.

Tous les caractères respectent les principes suivants :

✓ Les caractères alphabétiques (majuscules et minuscules) se suivent et sont ordonnés dans l'ordre alphabétique ; C'est-à-dire 'A' < 'B' < ... < 'Z' < ... < 'a' < 'b' < ... < 'z'.

✓ Les caractères numériques (les chiffres) se suivent et sont ordonnés dans l'ordre croissant ; C'est-à-dire '0' < '1' < . < '9'.

✓ La relation d'ordre est définie sur le type caractère tel que : Si x et y sont deux variables de type caractère alors : x < y si ord (x) < ord (y). A partir de la table ASCII, on peut voir facilement que '0' < ... < '9' < 'A' < ... < 'Z' < 'a' < ... < 'z'.

E. Le type Chaîne

Une chaîne est une suite de caractères. Une chaîne est encadrée par deux apostrophes doubles. Exemples : "Université de Msila", "Département MI", "Algorithmique", ...

✓ On appelle **longueur** d'une chaîne le nombre de caractères de cette chaîne.
"Algorithmique" est une chaîne de longueur 13.

✓ "" : représente la chaine vide de longueur 0. Elle ne contient aucun caractère.

Les fonctions prédéfinies sur les chaînes sont :

✓ **Length**(str) : elle fournit la longueur de la chaîne str.

✓ **Concat**(strl, str2) : elle fournit la chaîne obtenue par concaténation des deux chaînes strl et str2. Exemple : Concat("Module"," Algorithmique") = "Module Algorithmique"

Les opérateurs de relation (<, < =, >, >) sont définis sur le type chaîne tels que :

- ✓ Deux chaînes sont égales si elles sont identiques.
- ✓ La chaîne vide est inférieure à toute autre chaîne.
- ✓ Les chaînes sont ordonnées selon l'ordre lexicographique.
  Exemples : "Cours" < "cours", "cour" < "cours", "courage" < "cours".

Remarque sur les types : Chaque type a une taille et une représentation particulière en mémoire (de l'ordinateur). Il ne faut pas confondre les différentes formes de constantes.
Exemples : 3 (type entier), 3.0 (type réel), '3' (type caractère) et "3" (type chaine).

Contents