4. Variables et affectations

Les variables sont des « cases mémoires » dans laquelle une valeur est enregistrée. Cette valeur peut être modifiée tout au long du programme. Une variable est identifiée par un nom (ou identificateur). Ce nom ne doit comporter ni espace, ni accent, ni apostrophe et doit obligatoirement commencer par une lettre. Scilab distingue la casse, c'est-à-dire les majuscules et les minuscules.

Lorsqu'on modifie la valeur d'une variable, on dit qu'on lui **affecte** une valeur. L'**affectation** à la variable `Nom` de la valeur `Valeur` se fait avec la syntaxe suivante :

`-->Nom=Valeur` ou `-->Nom=Valeur;` (sans écho)

S'il n'existe pas de variable appelée `Nom`, cette instruction **crée une variable** `Nom` et lui affecte `Valeur`. S'il existe déjà une variable `Nom`, cette instruction **efface l'ancienne affectation et la remplace** par `Valeur`.

Lorsqu'une variable est créée, son nom apparaît dans le *Navigateur de variables* avec une icône qui indique la nature de son contenu.

⚠ Il faut bien respecter la syntaxe : ce qui se trouve **à gauche** du symbole = est modifié/créé par ce qui se trouve **à droite** du =.

⚠ Si on utilise un nom de variable non défini préalablement dans Scilab à droite de = on provoque une erreur.

Exemple Affectation avec écho :

```
-->x=2*3
 x  =
    6.
```

et sans écho :

```
-->x=2*3;
```

⚠ La valeur d'une variable est modifiée tout au long du programme.
Par exemple quelle est la valeur de `x` à la fin des instructions suivantes ?

```
-->x=2*3;
-->y=4;
-->y=2*y+1;
-->x=x*y;
```

Pour connaître la valeur affectée à une variable existante, il suffit de taper son nom dans la console et de valider. Par exemple sur l'exemple précédent :

```
-->x
 x  =
    54.
```

Pour connaître toutes les variables créées, on utilise la commande `who` (ou `whos` dont l'affichage est différent).

La commande **`clear`** permet d'effacer toutes les variables. La commande `clear x` permet d'effacer uniquement la variable `x`.

flfl fl fi v fl fl fi fi

C'est un **type de données** (comme l'est le type nombre), qui permet de manipuler du texte avec Scilab.

On appelle **chaîne de caractères** (« string » en anglais), une suite de *caractères alphanumériques* (les caractères disponibles sur le clavier d'un ordinateur).

Dans le langage Scilab, les chaînes de caractères sont délimitées par des apostrophes `'`. Lorsqu'une chaîne de caractère contient déjà le caractère `'`, il faut le doubler.

Exemple Par exemple :

```
--> 'Lorsqu''une'
 ans =
    Lorsqu'une
```

On peut affecter une chaîne de caractères à une variable.

Exemple Par exemple :

```
--> a = 'Scilab'
  a =
    Scilab
```

⚠ Scilab distingue une chaîne de chiffres du nombre qu'elle représente. Par exemple `'12'` est différent de `12` : le premier est du type chaîne de caractères et le second du type nombre.

Exemple Par exemple :

```
--> Txt = '12+13'
 Txt =
    12+13
```

L'instruction `Txt='12+13'` affecte à la variable `Txt` la valeur `12+13` (et non pas `25`) qui est une chaîne de cinq caractères. L'icône associée à `Txt` dans le navigateur de variables indique qu'on est en présence d'une chaîne de caractères.

Le nombre de caractères d'une chaîne `A` s'obtient avec l'instruction `length(A)`.

Exemple Par exemple :

```
--> length('toto')
 ans =
    4.
```

Si `A` est une chaîne de caractères, alors `part(A,i)` retourne son i-ième caractère (avec la convention qu'ils sont numérotés à partir de 1).

Exemple Par exemple :

```
--> A = '13=XIII'
  A =
    13=XIII

--> part(A,1)
 ans =
    1
--> part(A,2)
 ans =
    3
--> part(A,3)
 ans =
    =

--> part(A,8)
 ans =
```

Plus généralement, l'instruction `part(A,i:j)` donne la sous-chaîne extraite de `A`, formée des caractères entre les positions i et j (incluses).

Exemple Par exemple :

```
--> Txt = '12+13';

--> part(Txt,2:4)
 ans =
    2+1
```

L'opérateur addition + permet de coller deux chaînes de caractères : c'est l'opération de **concaténation**.

Exemple Par exemple :

```
--> A = 'Toto'
  A =
    Toto

--> B = 'Bonjour '
  B =
    Bonjour

--> B + A
 ans =
    Bonjour Toto
```

6. Les variables booléennes

Le résultat d'une expression logique (c'est-à-dire « vrai » ou « faux ») peut être stocké dans une variable, appelée alors **variable booléenne**.

Les booléens sont `%t` (true) et `%f` (false) pour les saisies et `T` et `F` pour l'affichage.

Les variables booléennes sont le résultat d'opérations logiques comme la comparaison, la négation etc ...

Exemple Par exemple :

```
--> a = %f
  a =
    F
```

Les symboles de comparaison sont :

==	(test d'égalité)	⚠ à ne pas confondre avec l'affectation = !
~=	(différent)	
<	(inférieur strict)	
<=	(inférieur ou égal)	
>	(supérieur strict)	
>=	(supérieur ou égal)	

Exemple Par exemple :

```
--> 3 == 5
 ans =
    F
```

On peut également former des expressions logiques plus compliquées :

~	(négation logique)	
&	(« et » logique)	
|	(« ou » inclusif)	⚠ c'est celui qu'on utilise en mathématiques.

Exemple Par exemple :

```
--> ( 3 == 5 ) & ( 3 == ( 2 + 1 ) )
 ans =
    F

--> ( 3 == 5 ) | ( 3 == ( 2 + 1 ) )
 ans =
    T
```