



# Programmer avec Matlab





# Les instructions de contrôle

## 2. La boucle conditionnelle « while » :

Pour exécuter une séquence d'instructions de manière répétée consiste à effectuer une boucle tant qu'une condition reste vérifiée. On arrête de boucler dès que cette condition n'est plus satisfaite. Ce processus est mis en œuvre par la boucle **while**.

**Syntaxe :**      **while** *expression logique*  
                          *séquence d'instructions*  
                          **end**

Où

Tant que *expression logique* est vraie le traitement *séquence d'instructions* est exécuté sous forme d'une boucle. Lorsque *expression logique* devient faux, on passe à l'instruction qui suit immédiatement l'instruction de fin de boucle (end).

- *expression logique* : une expression dont le résultat peut être vrai ou faux
- *séquence d'instructions* : le traitement à effectuer tant que *expression logique* est vraie.

# Les instructions de contrôle

## 2. La boucle conditionnelle « while » (exemple) :

Voici un exemple d'utilisation d'une boucle pour calculer  $n!$

```
Command Window

>> n = 5;
k = 1; nfac = 1;
while k <= n
    nfac = nfac*k;
    k = k+1;
end
>> nfac

nfac =

    120
```

# Les instructions de contrôle

## 3. L'instruction conditionnelle « if » :

On a parfois besoin d'exécuter une séquence d'instructions seulement dans le cas où une condition donnée est vérifiée au préalable.

L'instruction conditionnée la plus simple a la forme suivante:

**Syntaxe :**

**if**    *expression logique*  
          *séquence d'instructions*  
**end**

la *séquence d'instructions* n'est exécutée que si le résultat de l'évaluation de l'*expression logique est vraie* (c'est-à-dire vaut 1). Dans le cas contraire on exécute l'instruction qui suit le mot clé end.

Où

- *expression logique* : une expression dont le résultat peut être vrai ou faux
- *séquence d'instructions* : le traitement à effectuer si *expression logique* est vraie.

→ Dans le cas où l'*expression logique* est vraie, après exécution de la *séquence d'instructions* on reprend le programme à l'instruction qui suit le mot clé **end**.

# Les instructions de contrôle

## 4. L'instruction conditionnelle « if » alternative :

Il existe une séquence conditionnée sous forme d'alternatives :

**Syntaxe :**    **if**    *expression logique*  
                          *séquence d'instruction 1*  
                  **else**  
                          *séquence d'instruction 2*  
                  **end**

### Format des conditions en cascade

**if**    *expression logique 1*  
          *séquence d'instruction 1*  
**elseif** *expression logique 2*  
          *séquence d'instruction 2*  
          .....  
**end**  
**end**

### Remarque :

\* Attention à ne **pas** laisser d'espace entre **else** et **if** ; le mot clé est **elseif**.\*

# Les instructions de contrôle

## 4. L'instruction conditionnelle « if » (exemple) :

Voici un exemple d'utilisation d'if alternative :

Command Window

```
>> numex = 6;  
>> if numex == 1  
    A = ones(n);  
elseif numex == 2  
    A = magic(n);  
elseif numex == 3 | numex == 4  
    A = rand(n);  
else  
    error('numero d'exemple non prevu ...');  
end  
numero d'exemple non prevu ...
```

# Les instructions de contrôle

## 5. L'instruction de choix ventilé « switch » :

Si la variable **var** est égale à l'une des constantes  $cst_1, \dots, cst_N$ , ( par exemple  $cst_i$ ) alors la séquence d'instructions correspondante ( ici *séquence d'instructions i*) est exécutée. Le programme reprend ensuite à la première instruction suivant le mot-clé **end**. Si la variable **var** n'est égale à aucune des constantes la *séquence d'instructions par défaut* est exécutée.

**Syntaxe :**

```
switch var
    case cst1,
        séquence d'instructions 1
    case cst2,
        séquence d'instructions 2
    .....
    case cstN,
        séquence d'instructions N
    otherwise
        séquence d'instructions par défaut
end
```

- ❑ **var** : une variable numérique ou une variable chaîne de caractères.
- ❑ **cst<sub>1</sub>, ..., cst<sub>N</sub>** : sont des constantes numérique ou des constantes chaîne de caractères.
- ❑ **séquence d'instructions i** : est la séquence d'instructions à exécuter si le contenu de la variable **var** est égal à la constante **cst<sub>i</sub>** (**var == cst<sub>i</sub>**).

# Les instructions de contrôle

## 5. L'instruction de choix ventilé « switch » ( exemple ) :

```
Command Window

>> grade = 'B';
switch(grade)
case 'A'
    fprintf('Excellent!\n' );
case 'B'
    fprintf('Well done\n' );
case 'C'
    fprintf('Well done\n' );
case 'D'
    fprintf('You passed\n' );
case 'F'
    fprintf('Better try again\n' );
otherwise
    fprintf('Invalid grade\n' );
end
Well done
fx >>
```

```
Command Window

>> numex=4;
>> n=5;
>> switch numex
    case 1,
        A = ones(n)
    case 2,
        A = magic(n)
    case {3,4},
        A = rand(n)
    otherwise
        error('numero d'exemple non prevu ...');
end
A =

    0.7577    0.7060    0.8235    0.4387    0.4898
    0.7431    0.0318    0.6948    0.3816    0.4456
    0.3922    0.2769    0.3171    0.7655    0.6463
    0.6555    0.0462    0.9502    0.7952    0.7094
    0.1712    0.0971    0.0344    0.1869    0.7547
```



# Les instructions de contrôle

## 6. Interruption d'une boucle de contrôle :

- L'instruction **break** interrompt l'exécution du bloc d'instructions en cours d'exécution, et sort totalement de la boucle **for** ou **while**, en ignorant les itérations suivantes.
- En cas de boucles imbriquées, on interrompt seulement l'exécution de la boucle intérieure contenant l'instruction **break**

```
Command Window
>> a = 10;
% while loop execution
while (a < 20 )
    fprintf('value of a: %d\n', a);
    a = a + 1;
    if( a > 15)
        % terminate the loop using break statement
        break;
    end
end
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 15
fx >>
```