الكلية	العلوم
القسم	الرياضيات
المادة باللغة الانجليزية	Programming (MATLAB)
المادة باللغة العربية	البرمجة بلغة (ماتلاب)
المرحلة الدراسية	الثانية
اسم التدريسي	صفوت عبدالقادر حمد
عنوان المحاضرة باللغة الانجليزية	The nested if
عنوان المحاضرة باللغة العربية	اذا الشرطية الداخلية
رقم المحاضرة	الرابعة عشر
المصادر والمراجع	MATLAB A Practical Introduction to Programming and Problem Solving
	MATLAB The Language of Technical Computing
	MATLAB numerical computing

University Of Anbar

College Of Science

Math Department

Programing Language (MATLAB)

Lecture 14

The nested if

By:

Safwat A. Hamad

For the 2nd stage Math Department

1. Nested If-Else Statements

The ***if-else*** statement is used to choose between two actions. To choose from more than two actions, the ***if-else*** statements can be nested, meaning one statement inside another. For example, consider implementing the following continuous mathematical function: ***y = f(x):***

$y = 1$ if $x < -1$

$y = x^2$ if $-1 \leq x \leq 2$

$y = 4$ if $x > 2$

The value of y is based on the value of ***x***, which could be in one of three possible ranges. Choosing which range could be accomplished with three separate if statements, as follows:

```
if x < -1
y = 1;
end
if x >= -1 && x <=2
y = x^2;
end
if x > 2
y = 4;
end
```

Note: that the && in the expression of the second if statement is necessary. Writing the expression as -1<= x <= 2 would be incorrect; recall from previews lecturers that that expression would always be true, regardless of the value of the variable x.

As the three possibilities are mutually exclusive, the value of ***y*** can be determined by using three separate if statements. However, this is not very efficient code: all three logical expressions must be evaluated, regardless of the range in which ***x*** falls. For example, if x is less than -1, the first expression is true and 1 would be assigned to ***y***. However, the two expressions in the next two if statements are still evaluated.

Instead of writing it this way, the statements can be nested so that the entire if-else statement ends when an expression is found to be true:

```
if x < -1
    y = 1;
else
% If we are here, x must be >= -1
% Use an if-else statement to choose
% between the two remaining ranges
if x <= 2
    y = x^2;
else
% No need to check
% If we are here, x must be > 2
    y = 4;
end
end
```

By using a nested ***if-else*** to choose from among the three possibilities, not all conditions must be tested as they were in the previous example. In this case, if ***x*** is less than -1, the statement to assign 1 to ***y*** is executed and the ***if-else*** statement is completed, so no other conditions are tested. If, however, ***x*** is not less than -1, the else clause is executed. If the else clause is executed, then we already know that ***x*** is greater than or equal to -1, so that part does not need to be tested.

Instead, there are only two remaining possibilities: either ***x*** is less than or equal to 2, or it is greater than 2. An ***if-else*** statement is used to choose between those two possibilities. So, the action of the else clause was another ***if-else*** statement. Although it is long, all of the above code is one ***if-else*** statement, a nested ***if-else*** statement. The actions are indented to show the structure of the statement. Nesting if-else statements in this way can be used to choose from among 3, 4, 5, 6, ...; the possibilities are practically endless!

This is actually an example of a particular kind of nested ***if-else*** called a cascading ***if-else*** statement. This is a type of nested ***if-else*** statement in which the conditions and actions cascade in a stair-like pattern.

Not all nested ***if-else*** statements are cascading. For example, consider the following (which assumes that a variable ***x*** has been initialized):

```
if x >= 0
    if x < 4
        disp('a')
    else
        disp('b')
    end
else
    disp('c')
end
```

2. The ElseIf Clause

In some programming languages, choosing from multiple options means using nested if-else statements. However, MATLAB has another method of accomplishing this using the ***elseif*** clause.

THE EFFICIENT METHOD

To choose from among more than two actions, the elseif clause is used. For example, if there are n choices (where n > 3 in this example), the following general form would be used:

```
if condition1
    action1
elseif condition2
    action2
elseif condition3
    action3
% etc: there can be many of these
```

```
else
    actionn % the nth action
end
```

The actions of the if, elseif, and else clauses are naturally bracketed by the reserved words if, elseif, else, and end.

For example, a previous example could be written using the elseif clause, rather than nesting if-else statements:

```
if x < -1
    y = 1;
elseif x <= 2
    y = x^2;
else
    y = 4;
end
```

Note that in this example, we only need one end. So, there are three ways of accomplishing the original task: using three separate if statements, using nested if-else statements, and using an if statement with elseif clauses, which is the simplest.

Another example demonstrates choosing from more than just a few options. The following application ***grade*** receives an integer quiz grade, which should be in the range from 0 to 10. The application then returns a corresponding letter grade, according to the following scheme: a 9 or 10 is an 'A', an 8 is a 'B', a 7 is a 'C', a 6 is a 'D', and anything below that is an 'F'. As the possibilities are mutually exclusive, we could implement the grading scheme using separate if statements. However, it is more efficient to have one if-else statement with multiple elseif clauses. Also, the function returns the letter 'X' if the quiz grade is not valid. The application assumes that the input is an integer.

```
grade.m
```

```
application grade
% First, error-check
```

```
    if quiz < 0 || quiz > 10
    grade = 'error must enter 1 to 10 only';
    % If here, it is valid so figure out the
    % corresponding letter grade
    elseif quiz == 9 || quiz == 10
    grade = 'A';
    elseif quiz == 8
    grade = 'B';
    elseif quiz == 7
    grade = 'C';
    elseif quiz == 6
    grade = 'D';
    else
    grade = 'F';
    end
    end
```

In the part of this ***if*** statement that chooses the appropriate letter grade to return, all of the logical expressions are testing the value of the variable quiz to see if it is equal to several possible values, in sequence (first 9 or 10, then 8, then 7, etc.). This part can be replaced by a switch statement.