



ME 444

MATLAB® FOR ENGINEERS

Lecturer:

Dr. Nurettin Furkan DOĞAN

Mechanical Engineering Department

4 th Floor Office No: 303

nfdogan@gantep.edu.tr

0342 317 1200- 2569



Office Hours:

Tuesday: 13.30- 15:00

Friday : 10.00- 11.30

CHAPTER 6

CONTROL STRUCTURES-1

- Comparison statements in MATLAB

>	greater than
<	less than
>=	great or equal
<=	less or equal
==	is it equal?
~=	is it not equal?

- If the comparison operation is true, the result will be **1**, if false, the result will be **0**.

- Logical control statements:

	OR
&&	AND
~	NOT

Let's examine the concepts of true / false

- 0 is used to represent the concept of **false**. Any non-zero value can be used to represent the concept of true - so expressions like 5 or 'x' result in "logical true (1)".
- This can cause common errors.
- For example, the following examples will always return true (1). (Because the values on the right side of the logical expressions are non-zero values. Therefore, whatever the result is, the result of the || operation will be true (1)).

```
>> number=12;
```

```
>> number<5 || 6
```

```
1 (true)
```

```
0 (false)
```

```
ans =
```

```
1
```

```
0 (false)
```

```
>> letter='y';
```

```
>> letter=='n' || 'N'
```

```
1 (true)
```

```
ans =
```

```
1
```

Truth Table of Logical Expressions



&&(and)			(or)			!(not)	
x	y	result	X	Y	result	X	result
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

If Conditional Statement



- The **if** statement is used to determine whether a statement or a group of statements can be executed.
- The general syntax is:

```
if conditional statement  
    operations  
end
```

- ✓ conditional statement : consists of control statements.
- ✓ operations : It consists of processes that will run depending on the specified condition. If the condition expression result is true, then the operations will be executed. If not, these processes will not work.

If Conditional Statement

- For example, the following expression checks if the value of the variable is negative. If so, the value of the variable is changed to zero; if not, no action is taken.

```
if num<0
    num=0
end
```

Note here, if the semicolon (;) was not used when assigning the value of the variable, pressing the enter key would print the value of the num variable on the screen.

✓ The if conditional statement can also be used in the command window.

```
>> num = -4;
>> if num < 0
    num = 0
end
num =
    0

>> num = 5;
>> if num < 0
    num = 0
end
>>
```

If Conditional Statement

- The if conditional statement can be used to check whether the entered value is negative, for example, because negative values cannot be used in finding the square root. A zero value can be assigned instead of a negative value entered.

sqrtfunction.m

```
%this program calculates the square root of the entered value.  
num=input('Please enter a number:');  
% equal zero if negative number is entered  
if num<0  
    num=0;  
end  
fprintf('Square root of %.1f is %.1f \n', num,sqrt(num))
```

The **sqrt** function is one of the built-in functions in MATLAB. Without this function, the expression $\text{num}^{1/2}$ could also be used.

If Conditional Statement



- When the previous program is executed:

```
>>sqrtfunction  
Please enter a number: -4.2  
Square root of 0.0 is 0.0  
  
>>sqrtfunction  
Please enter a number : 1.44  
Square root of 1.4 is 1.2
```

If Conditional Statement

- We can make this example more useful. If the entered number has a negative value, we can take the absolute value (absolute value, abs) of that number and then calculate its square root:

sqrtfunction2.m

```
%this program calculates the square root of the entered value.
num=input('Please enter a number:');
% warn the user if negative number is entered and use the absolute %value of the
number
if num<0
    disp('I need to get the absolute value of the number for this number!!');
    num=abs(num);
end
fprintf('Square root of %.1f is %.1f \n', num,sqrt(num))
```

The abs(x) function is one of the built-in functions in MATLAB. Gets the absolute value of x.

If Conditional Statement



- When the previous program is executed:

```
>>sqrtfunction2
```

```
Please enter a number: -25
```

```
For this number I have to get the absolute value of the number!!
```

```
Square root of 25.0 is 5.0
```

If Conditional Statement

QUESTION: Suppose we want to create a vector of increasing integer values from **mymin** to **mymax**. We will write a **createvec** function that takes two input arguments from the keyboard, **mymin** and **mymax**, and creates a vector from **mymin** to **mymax** in 1 increments. First, we need to make sure that **mymin** is less than **mymax**. If not, we will need to change its values before creating the vector. How can we achieve this?

SOLUTION:

1. Compare whether the first value entered is less than the second. `mymin < mymax`
2. If not, replace the first value entered with the second value.
3. Create the variable named vector in increments of 1 between mymin and mymax.

If Conditional Statement



SOLUTION: createvec.m FUNCTION

```
%This function creates a vector with an increment of 1 between the  
%two entered values.  
function vector=createvec(mymin,mymax)  
    if mymin>mymax  
        temp=mymin;  
        mymin=mymax;  
        mymax=temp;  
    end  
    %Let's create the variable named vector with the colon operator(:)  
    vector=mymin:mymax;  
end
```

For example:
Let's enter the
values 10 and 3.

→ $10 > 3$

→ temp=10

→ mymin=3

→ mymax=10

Mymin and mymax
have been
replaced.

As a result:

```
vector= [3 4 5 6  
7 8 9 10]
```

If-else Conditional Statement

- The if-else statement chooses between two operations based on the given condition.
- The general writing format is:

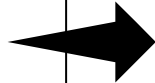
```
if conditional statement  
    operations 1  
else  
    operations 2  
end
```

✓ Only one of the processes will run. If the given condition statement is true, operations 1 will run, otherwise operations 2 will run.

If-else Conditional Statement

- The if-else statement may not always be necessary.
- For example:

```
if num<0
    num=0;
else
    num=num;
end
```



✓ Here, the else statement doesn't actually do anything. Because if the num value is less than zero, it will take the value zero. If not, the num value will already be equal to the initial value for num. Hence, else is non-functional.



```
if num<0
    num=0;
end
```

✓ This code will work.

If-else Conditional Statement

EXAMPLE: Let's write a program that checks if the radius value entered by the user is valid (a positive value) and writes that the entered value is a valid value, otherwise it is an invalid value, then calculates the circle area.

```
%This program determines whether the entered radius value is a  
%valid value or not.
```

```
r=input('Please enter the radius value: ');
```

```
if r<=0
```

```
    fprintf('Sorry th
```

```
else
```

```
    fprintf('The valu
```

```
    fprintf('Area is
```

```
end
```

Command Window

```
>> Untitled8
```

```
Please enter the radius value: 2
```

```
The value 2.00 entered is a valid value.
```

```
Area is 12.57.
```

```
>> Untitled8
```

```
Please enter the radius value: -5
```

```
Sorry the -5.00 value is not valid.
```



If-else Conditional Statement

→ In this example, we can also make the warning/error message appear in red on the screen by using the `error` function.

```
%This program determines whether the entered radius value is a  
%valid value or not.
```

```
r=input('Please enter the radius value: ');
```

```
if r<=0
```

```
    error('Sorry the %.2f value is not valid.\n',r);
```

```
else
```

```
    fprintf('Th
```

```
    fprintf('Ar
```

```
end
```

Command Window

```
>> Untitled8
```

```
Please enter the radius value: -8
```

```
Error using Untitled8 (line 4)
```

```
Sorry the -8.00 value is not valid.
```



Nested If-else Conditional Statement

- Nested if-else statements can be used to choose between more than two actions (an if or if-else statement as another action's)
- The general writing format is:

```
if conditional statement 1
    operation 1
else
    if conditional statement 2
        operation 2
    else
        if conditional statement 3
            operation 3
        (can be continued any number of times.)
    end
end
end
```

Nested If-else Conditional Statement

Example: Multiple ways can be used to express a mathematical expression in MATLAB.

$$\begin{aligned} y = 1 & \rightarrow x < -1 \\ y = x^2 & \rightarrow -1 \leq x \leq 2 \\ y = 4 & \rightarrow x > 2 \end{aligned}$$

1. Using **if**



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

Note here, if we wrote `-1<=x<=2` it would be wrong. Instead, we wrote `x>=-1 && x<=2`.

2. İç içe **if-else** şart ifadesi kullanarak

```
if x<-1
    y=1;
else
    %If we pass to this part, it should be x>=-1
    if x<=2
        %here if x is greater than -1 but less than 2
        %then continues
        y=x^2;
    else
        %No control needed here.
        %Because if this part of the code is working, x
        %is definitely greater than 2.
        y=4;
    end
end
```

Nested If-else Conditional Statement

Example: Let's write a program that tells how many days the entered month consists of.

```
month=input('Which month do you want to know the number of days?(1-12): ');  
if month==1 || month==3 || month==5 || month==7 || month==10 || month==12  
    disp('This month consists of 31 days');  
else  
    if month==2  
        disp('This month consists of 28 days');  
    else  
        disp('This month consists of 30 days');  
    end  
end
```



```
Command Window  
>> Untitled8  
Which month do you want to know the number of days?(1-12): 3  
This month consists of 31 days  
>> Untitled8  
Which month do you want to know the number of days?(1-12): 2  
This month consists of 28 days  
>> Untitled8  
Which month do you want to know the number of days?(1-12): 4  
This month consists of 30 days
```

elseif Conditional Statement

- MATLAB also has an **elseif** statement that shortens the written code (and reduces the number of **ends**).
- The general writing format is:

```
if conditional statement 1
    operations 1
elseif conditional statement 2
    operations 2
elseif conditional statement 3
    operations 3
(can be continued any number of
times.)
end
```

elseif Conditional Statement

Example: Let's make the program that prints the number of days in the months with this structure. In fact, our program will give an error message to the user, when additional numbers other than 1-12 are entered.

```
month=input('Which month do you want to know the number of days?(1-12): ');  
if month==1 || month==3 || month==5 || month==7 || month==10 || month==12  
    disp('This month consists of 31 days');  
elseif month==2  
    disp('This month consists of 28 days');  
elseif month ==4 || month ==6 || month ==8 || month ==9 || month ==11  
    disp('This month consists of 30 days');  
else error('You entered the wrong number!!');  
  
end
```



Command Window

```
>> Untitled8  
Which month do you want to know the number of days?(1-12): 1  
This month consists of 31 days  
>> Untitled8  
Which month do you want to know the number of days?(1-12): 2  
This month consists of 28 days  
>> Untitled8  
Which month do you want to know the number of days?(1-12): 13  
Error using Untitled8 (line 8)  
You entered the wrong number!!
```

elseif Conditional Statement

Example: Let's write a function that finds whether the entered data is a scalar, vector, or matrix.

```
%This function reports whether the entered value is a
%scalar, vector, or matrix.
%To call the function, findargtype(value) must be written.
function report=findargtype(inputarg)
[ row, column]=size(inputarg);
if row==1 && column==1
    report='scalar';
elseif row==1 || column==1
    report='vector';
else
    report='matrix';
end
end
```

If the size of the entered value (row, column):
(row, column)=(1,1) → scalar
(row, column)≠(1,1) → matrix
(row)=1 OR (column)=1 → vector

Command Window

```
>> findargtype(3)

ans =

    'scalar'

>> findargtype([3 5; 6 4])

ans =

    'matrix'

>> findargtype([ 1 3 4])

ans =

    'vector'
```

elseif Conditional Statement

Example: Write a function that sends the letter equivalent to the main program according to the exam grade entered as an integer. (Let the grade entered in the program be a value between 0 and 10: A if it is 9 or 10, B if it is 8, C if it is 7, D if it is 6, F for other grades. If a value other than these grades is entered, it will send the letter X to the main program.)

```
%This function returns the letter equivalent of the entered grade (1-10)
function let=grade(gpa)
%First check the gpa is valid or not
if gpa<0 || gpa>10
    let='X';
%If the gpa is valid, check for the letter
elseif gpa==9 || gpa==10
    let='A';
elseif gpa==8
    let='B';
elseif gpa==7
    let='C';
elseif gpa==6
    let='D';
else
    let='F';
end
end
```

Command Window

```
>> grade(10)
```

```
ans =
```

```
    'A'
```

```
>> grade(5)
```

```
ans =
```

```
    'F'
```

```
>> grade(15)
```

```
ans =
```

```
    'X'
```

fx

Next week

Chapter 6

Control Structures 2