

## Chapter Three

### Using Script Files

#### *THE MATLAB WORKSPACE AND THE WORKSPACE WINDOW*

```
>> 'Variables in memory'
```

Typing a string.

```
ans =
Variables in memory
```

The string is assigned to ans.

```
>> a = 7;
>> E = 3;
>> d = [5, a+E, 4, E^2]
```

Creating the variables a, E, d, and g.

```
d =
     5     10     4     9
>> g = [a, a^2, 13; a*E, 1, a^E]
```

```
g =
     7     49     13
    21     1    343
>> who
```

Your variables are:

```
E   a   ans  d   g
```

The who command displays the variables currently in the workspace.

```
>> whos
```

Name	Size	Bytes	Class	Attributes
E	1x1	8	double	
a	1x1	8	double	
ans	1x19	38	char	
d	1x4	32	double	
g	2x3	48	double	

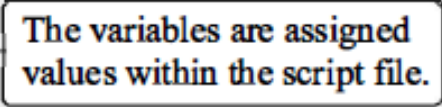
The whos command displays the variables currently in the workspace and information about their size and other information.

```
>>
```

## INPUT TO A SCRIPT FILE

1. The variable is defined and assigned a value in the script file.

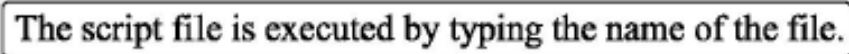
```
% This script file calculates the average points scored in three games.  
% The assignment of the values of the points is part of the script file.  
game1=75;  
game2=93;  
game3=68;  
ave_points=(game1+game2+game3)/3
```



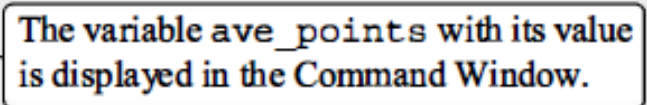
The variables are assigned values within the script file.

The display in the Command Window when the script file is executed is:

```
>> Chapter4Example2  
  
ave_points =  
    78.6667  
>>
```



The script file is executed by typing the name of the file.



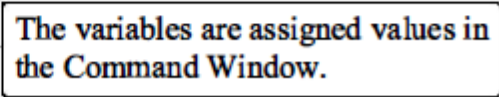
The variable ave\_points with its value is displayed in the Command Window.

2. The variable is defined and assigned a value in the Command Window.

```
% This script file calculates the average points scored in three games.  
% The assignment of the values of the points to the variables  
% game1, game2, and game3 is done in the Command Window.  
  
ave_points=(game1+game2+game3)/3
```

The Command Window for running this file is:

```
>> game1 = 67;  
>> game2 = 90;  
>> game3 = 81;
```



The variables are assigned values in the Command Window.

```
>> Chapter4Example3
ave_points =
    79.3333
>> game1 = 87;
>> game2 = 70;
>> game3 = 50;
>> Chapter4Example3
ave_points =
    69
>>
```

The script file is executed.

The output from the script file is displayed in the Command Window.

New values are assigned to the variables.

The script file is executed again.

The output from the script file is displayed in the Command Window.

3. The variable is defined in the script file, but a specific value is entered in the Command Window when the script file is executed.

```
variable_name = input('string with a message that
is displayed in the Command Window')
```

```
% This script file calculates the average of points scored in three games.
% The points from each game are assigned to the variables by
% using the input command.
game1=input('Enter the points scored in the first game ');
game2=input('Enter the points scored in the second game ');
game3=input('Enter the points scored in the third game ');
ave_points=(game1+game2+game3)/3
```

```
>> Chapter4Example4
Enter the points scored in the first game 67
Enter the points scored in the second game 91
Enter the points scored in the third game 70

ave_points =
    76
>>
```

The computer displays the message. Then the value of the score is typed by the user and the **Enter** key is pressed.

## OUTPUT COMMANDS

### 4.3.1 The disp Command

The `disp` command is used to display the elements of a variable without displaying the name of the variable, and to display text. The format of the `disp` command is:

```
disp(name of a variable) or disp('text as string')
```

- Every time the `disp` command is executed, the display it generates appears in a new line. One example is:

```
>> abc = [5 9 1; 7 2 4]; A 2x3 array is assigned to variable abc.
>> disp(abc)           The disp command is used to display the abc array.
    5     9     1
    7     2     4      The array is displayed without its name.

>> disp('The problem has no solution.')
The problem has no solution.
>>                    The disp command is used to display a message.
```

```
% This script file calculates the average points scored in three games.
% The points from each game are assigned to the variables by
% using the input command.
% The disp command is used to display the output.

game1=input('Enter the points scored in the first game ');
game2=input('Enter the points scored in the second game ');
game3=input('Enter the points scored in the third game ');
ave_points=(game1+game2+game3)/3;
disp(' ')           Display empty line.
disp('The average of points scored in a game is:')         Display text.
disp(' ')           Display empty line.
disp(ave_points)   Display the value of the variable ave_points.
```

```
>> Chapter4Example5
Enter the points scored in the first game    89
Enter the points scored in the second game   60
Enter the points scored in the third game    82

The average of points scored in a game is:

77
```

An empty line is displayed.  
The text line is displayed.  
An empty line is displayed.  
The value of the variable ave\_points is displayed.

```
yr=[1984 1986 1988 1990 1992 1994 1996];
pop=[127 130 136 145 158 178 211];
tableYP(:,1)=yr';
tableYP(:,2)=pop';
disp('        YEAR        POPULATION')
disp('                (MILLIONS)')
disp(' ')
disp(tableYP)
```

The population data is entered in two row vectors.  
yr is entered as the first column in the array tableYP.  
pop is entered as the second column in the array tableYP.  
Display heading (first line).  
Display heading (second line).  
Display an empty line.  
Display the array tableYP.

When this script file (saved as PopTable) is executed, the display in the Command Window is:

```
>> PopTable
        YEAR        POPULATION
                (MILLIONS)

1984            127
```

Headings are displayed.  
An empty line is displayed.

```
1986            130
1988            136
1990            145
1992            158
1994            178
1996            211
```

The tableYP array is displayed.

**Relational operators:**

Relational operators in MATLAB are:

<b>Relational operator</b>	<b>Description</b>
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
~=	Not Equal to

Some examples are:

```

>> 5>8
ans =
    0
>> a=5<10
a =
    1
>> y=(6<10)+(7>8)+(5*3==60/4)
y =
    2
>> b=[15 6 9 4 11 7 14]; c=[8 20 9 2 19 7 10];
>> d=c>=b
d =
    0    1    1    0    1    1    0
>> b == c
ans =
    0    0    1    0    0    1    0

```

Checks if 5 is larger than 8.

Since the comparison is false (5 is not larger than 8) the answer is 0.

Checks if 5 is smaller than 10, and assigns the answer to a.

Since the comparison is true (5 is smaller than 10) the number 1 is assigned to a.

Using relational operators in math expression.

Equal to 1 since 6 is smaller than 10.

Equal to 0 since 7 is not larger than 8.

Equal to 1 since 5\*3 is equal to 60/4.

Define vectors b and c.

Checks which c elements are larger than or equal to b elements.

Assigns 1 where an element of c is larger than or equal to an element of b.

Checks which b elements are equal to c elements.

```
>> b~=c
ans =
     1     1     0     1     1     0     1
>> f=b-c>0
f =
     1     0     0     1     0     0     1
>> A=[2 9 4; -3 5 2; 6 7 -1]
A =
     2     9     4
    -3     5     2
     6     7    -1
>> B=A<=2
```

Checks which b elements are not equal to c elements.

Subtracts c from b and then checks which elements are larger than zero.

Define a 3 x 3 matrix A.

Checks which elements in A are smaller than or equal to 2. Assigns the results to matrix B.

```
B =
     1     0     0
     1     0     1
     0     0     1
```

6.1 Relational and Logical Operators

**Logical operators:**

Logical operators in MATLAB are:

<u>Logical operator</u>	<u>Name</u>	<u>Description</u>
& Example: A&B	AND	Operates on two operands (A and B). If both are true, the result is true (1); otherwise the result is false (0).
 Example: A B	OR	Operates on two operands (A and B). If either one, or both, are true, the result is true (1); otherwise (both are false) the result is false (0).
~ Example: ~A	NOT	Operates on one operand (A). Gives the opposite of the operand; true (1) if the operand is false, and false (0) if the operand is true.

```
>> 3&7
```

3 AND 7.

```

ans =
    1
>> a=5|0
a =
    1
>> ~25
ans =
    0
>> t=25*((12&0)+(~0)+(0|5))
t =
    50
>> x=[9 3 0 11 0 15]; y=[2 0 13 -11 0 4];
>> x&y
ans =
    1    0    0    1    0    1
>> z=x|y
z =
    1    1    1    1    0    1
>> ~(x+y)
ans =
    0    0    0    1    1    0

```

3 and 7 are both true (nonzero), so the outcome is 1.

5 OR 0 (assign to variable a).

1 is assigned to a since at least one number is true (nonzero).

NOT 25.

The outcome is 0 since 25 is true (nonzero) and the opposite is false.

Using logical operators in a math expression.

Define two vectors x and y.

The outcome is a vector with 1 in every position where both x and y are true (nonzero elements), and 0s otherwise.

The outcome is a vector with 1 in every position where either or both x and y are true (nonzero elements), and 0s otherwise.

The outcome is a vector with 0 in every position where the vector x + y is true (nonzero elements), and 1 in every position where x + y is false (zero elements).

**Precedence****Operation**

1 (highest)	Parentheses (if nested parentheses exist, inner ones have precedence)
2	Exponentiation
3	Logical NOT (~)
4	Multiplication, division
5	Addition, subtraction
6	Relational operators (>, <, >=, <=, ==, ~=)
7	Logical AND (&)
8 (lowest)	Logical OR ( )



```
>> x=-2; y=5;
```

Define variables x and y.

```
>> -5<x<-1
```

```
ans =
     0
```

This inequality is correct mathematically. The answer, however, is false since MATLAB executes from left to right.  $-5 < x$  is true (=1) and then  $1 < -1$  is false (0).

```
>> -5<x & x<-1
```

```
ans =
     1
```

The mathematically correct statement is obtained by using the logical operator &. The inequalities are executed first. Since both are true (1), the answer is 1.

```
>> ~(y<7)
```

```
ans =
     0
```

$y < 7$  is executed first, it is true (1), and  $\sim 1$  is 0.

```
>> ~y<7
```

```
ans =
     1
```

$\sim y$  is executed first,  $y$  is true (1) (since  $y$  is nonzero),  $\sim 1$  is 0, and  $0 < 7$  is true (1).

```
>> ~((y>=8) | (x<-1))
```

```
ans =
     0
```

$y \geq 8$  (false), and  $x < -1$  (true) are executed first. OR is executed next (true).  $\sim$  is executed last, and gives false (0).

```
>> ~(y>=8) | (x<-1)
```

```
ans =
     1
```

$y \geq 8$  (false), and  $x < -1$  (true) are executed first. NOT of  $(y \geq 8)$  is executed next (true). OR is executed last, and gives true (1).

## CONDITIONAL STATEMENTS

`if` conditional expression consisting of relational and/or logical operators.

Examples:

```

if a < b
if c >= 5
if a == b
if a ~= 0
if (d<h) & (x>7)
if (x~=13) | (y<0)

```

All the variables must  
have assigned values.

The if- end Structure

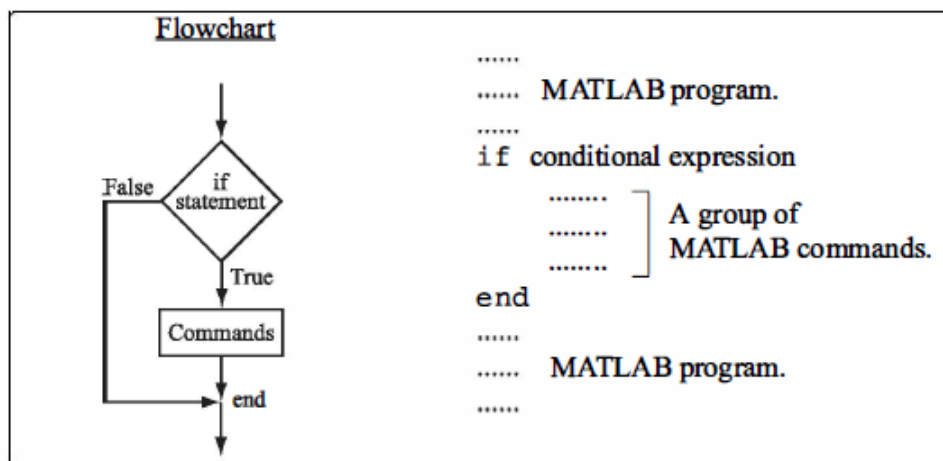


Figure 6-1: The structure of the if-end conditional statement.

**Example 1:-**

Write a program to test if the numbers are entered positive.

```

a=input('a=');
if a>0
b=a;
disp('a is positive')
end

```

**Example 2 :-**

W. P. to find the value of  $y = \frac{x^3 - 5x + 2}{x - 3}$ ,  $x \neq 3$

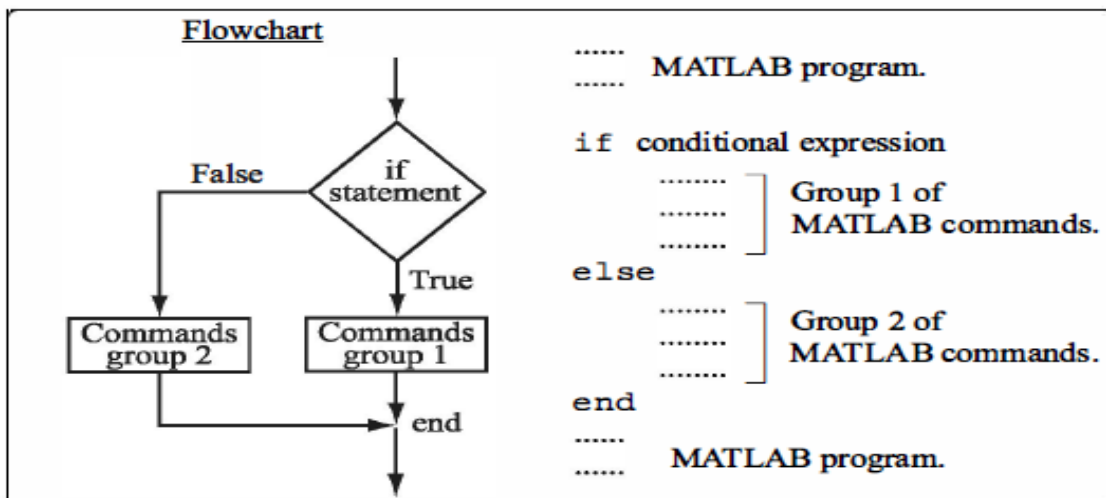
```
x=input('x=');
if (x~=3)
y=(x^3-5*x+2)/(x-3);
disp(y);
end
```

**Example 3:-** W.P. to input the following number 8 5 10 11 18 21

Then print the even number

```
x=input('x=');
if (rem(x,2)==0)
disp(x);
end
```

The if - else - end Structure



**Figure 6-2: The structure of the if-else-end conditional statement.**

**Example 1 :-**

Write a program to input the temperature to either print “normal condition” if temperature greater than 37 or print 'catch a fever'

```
temp=input('enter the measured
temperature:');
if temp >37
disp('normal condition')
else
disp('catch a fever')
end
```

**Example 2 :-**

Write a program to test input x is positive or negative number

```
x=input('x=');
if x>= 0
disp('x is positive');
else
disp('x is negative');
end
```

**Example 3 :-**

Write a program to input number then test if it is even or odd number?

```
x=input('x=');
if(rem(x,2)==0)
disp('x is even');
disp(x);
else
disp('x is odd');
```

```
disp(x);
end
```

### 6.2.3 The if-elseif-else-end Structure

The if-elseif-else-end structure is shown in Figure 6-3. The figure shows how the commands are typed in the program, and gives a flowchart that illustrates the flow, or the sequence, in which the commands are executed. This structure includes two conditional statements (if and elseif) that make it possible to select one out of three groups of commands for execution. The first line is an if statement with a conditional expression. If the conditional expression is true, the program executes group 1 of commands between the if and the

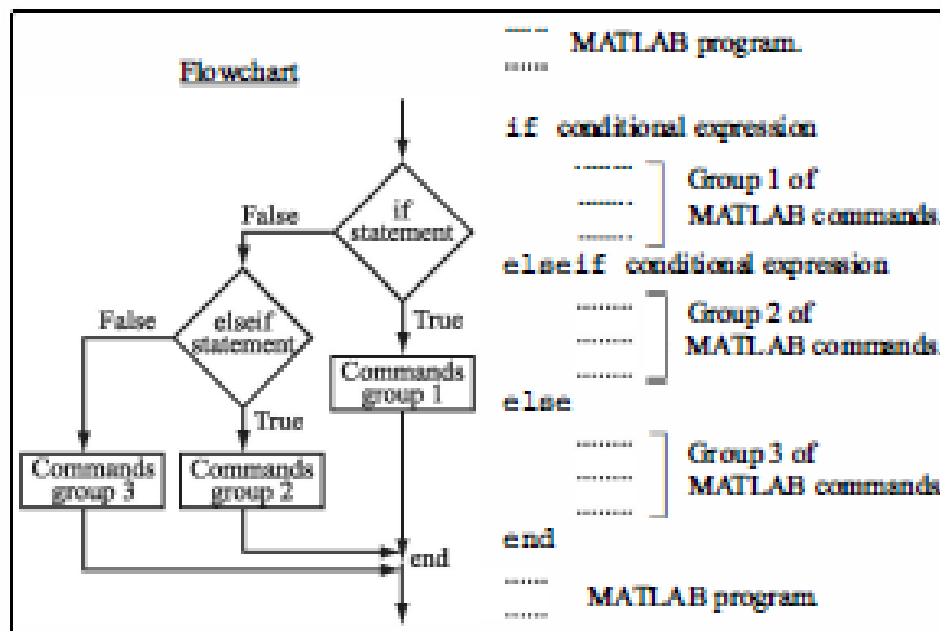


Figure 6-3: The structure of the if-elseif-else-end conditional statement.

**Example 1:-** Write a program to input number  $x$  to find the value of  $y$  such that

$$y = \begin{cases} x^2 - 5x + 1 & \text{if } x \geq 3 \\ x + 5 & \text{if } x < 2 \\ x^2 - 7x + 3 & \text{otherwise} \end{cases}$$

```
x=input('x=');
if x>= 3
    y=x^3-5*x+1;
```

```

        disp(y);
elseif x<2
    y=x+5;
    disp(y);

else
    y=x^2-7*x+3;
    disp(y)
end

```

**Example 2:-** Write a program to input the degree of student to print the grade as the following

Degree	0-49	50-59	60-69	70-79	80-89	90-100
Grade	Fail	Pass	Medium	Good	Very good	Excellent

```

x=input('x=');
if x > 100
    % fprintf('input  %-.2f out of the range',x);
    disp('the input number out of the range');
elseif x>= 90 && x<=100
    disp('Grade is Excellent');
elseif x>=80 && x<=89
    disp('Grade is V. Good');
elseif x>=70 && x<=79
    disp('Grade is Good');
elseif x>=60 && x<=69
    disp('Grade is Medium');
elseif x>=50 && x<=59
    disp('Grade is Pass');

```

```

else
    disp('Grade is Fail');
end

```

### Home Works

1- write a program to find the average of the student exams and write the grade Of it as the following:-

No.	Exams
1	40
2	60
3	30
4	50
5	70

Avr.	grade
$100 \leq \text{Avr} \leq 90$	A
$90 < \text{Avr} \leq 80$	B
$80 < \text{Avr} \leq 70$	C
$70 < \text{Avr} \leq 60$	D
$60 < \text{Avr} \leq 50$	E
otherwise	F

2- Write a program to input number  $x$  to find the value of  $y$  such that

$$y = \begin{cases} e^x & \text{if } x = (20 \text{ to } 25) \\ |x| & \text{if } x = (30 \text{ to } 35) \\ \sin x & \text{if } x = (40 \text{ to } 45) \end{cases}$$

3- Write a program to input the values  $x$  and  $y$  to find the value of  $z$

$$z = \frac{(x+y)}{3!} + \frac{(x+y)^3}{5!} + \frac{(x+y)^5}{7!} + \dots + \frac{(x+y)^{21}}{23!} + \frac{(x+y)^{23}}{25!}$$

4- A vector is given by  $V =$

$[5, 17, 15, 8, 0, -7, 12, -3, 20, -6, 6, 4, -7, 16]$ . Write a program that doubles the elements that are positive and are divisible by 3 and divisible by 5, raises to the power of 3 the elements that are negative but greater than  $-5$ .

5- A vector is given by  $V =$

$[13, -4, -3, 1, 15, -8, 12, -18, 5, 21]$ . Write a program to multiply the elements of  $V$  by 2 if such elements are positive and even number, and multiply the elements of  $V$  by  $-5$  if such element are negative and odd number.

6- A list of 30 exam scores is:

31, 70, 92, 5, 47, 88, 81, 73, 51, 76, 80, 90, 55, 23, 43, 98, 36, 87, 22, 61,  
19, 69, 26, 82, 89, 99, 71, 59, 49, 64

Write a computer program that determines how many grades are from 0 to 19, from 20 to 39, from 40 to 59, from 60 to 79, and from 80 to 100.

7- let's say we were asked to write a program that calculates the tip based on amount of bills, using the following rules and the variable 'bill':

- bill is less than \$10
  - Tip is \$1.80



- bill is between \$10 and \$60
  - Tip is %18
- bill is above \$60
  - Tip is %20

Q/ Write a script file using Conditional If-Elseif-Else statements to shows the grade of the score as following:-

Grade	Score
A+	100
A	Score $\geq 90$
B	Score $\geq 80$
C	Score $\geq 70$
D	Score $\geq 60$
F	otherwise

**Example.** Suppose a bank offers annual interest of 3% on balances of less than £5,000, 3.25% on balances of £5,000 or more but less than £10,000, and 3.5% for balances of £10,000 or more. The following program calculates an investor's new balance after one year.