The script file (saved as Exp6_1) is executed in the Command Window:

```
>> Exp6_1
NdaysTabove75 =
     7
NdaysTbetween65and80 =
    12
datesTbetween50and60 =
     1     4     5     7    21    23
```

For 7 days the temp was above 75.

For 12 days the temp was between 65 and 80.

Dates of the month with temp between 50 and 60.

6.2 CONDITIONAL STATEMENTS

A conditional statement is a command that allows MATLAB to make a decision of whether to execute a group of commands that follow the conditional statement, or to skip these commands. In a conditional statement, a conditional expression is stated. If the expression is true, a group of commands that follow the statement are executed. If the expression is false, the computer skips the group. The basic form of a conditional statement is:

`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.

- Conditional statements can be a part of a program written in a script file or a user-defined function (Chapter 7).
- As shown below, for every `if` statement there is an `end` statement.

The `if` statement is commonly used in three structures, `if-end`, `if-else-end`, and `if-elseif-else-end`, which are described next.

6.2.1 The `if-end` Structure

The `if-end` conditional statement is shown schematically in Figure 6-1. The figure shows how the commands are typed in the program, and a flowchart that symbolically shows the flow, or the sequence, in which the commands are executed. As the program executes, it reaches the `if` statement. If the conditional expression in the `if` statement is true (1), the program continues to execute the

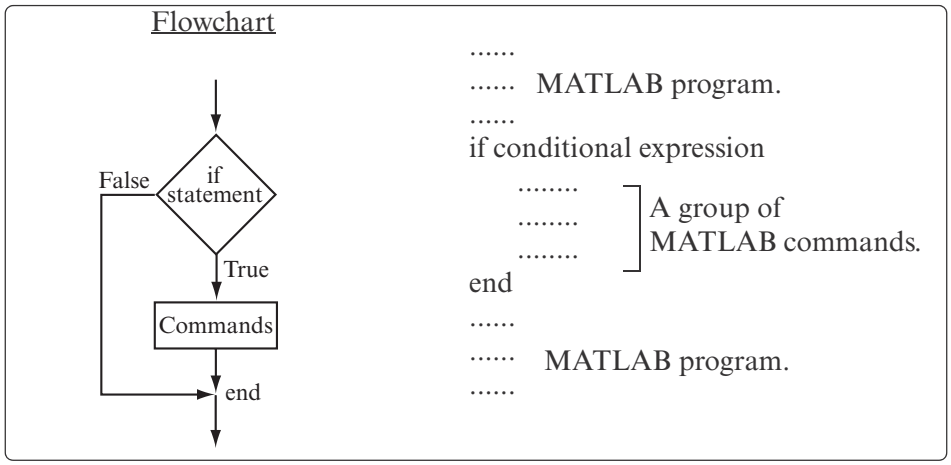


Figure 6-1: The structure of the `if-end` conditional statement.

commands that follow the `if` statement all the way down to the `end` statement. If the conditional expression is false (0), the program skips the group of commands between the `if` and the `end`, and continues with the commands that follow the `end`.

The words `if` and `end` appear on the screen in blue, and the commands between the `if` statement and the `end` statement are automatically indented (they don't have to be), which makes the program easier to read. An example where the `if-end` statement is used in a script file is shown in Sample Problem 6-2.

Sample Problem 6-2: Calculating worker's pay

A worker is paid according to his hourly wage up to 40 hours, and 50% more for overtime. Write a program in a script file that calculates the pay to a worker. The program asks the user to enter the number of hours and the hourly wage. The program then displays the pay.

Solution

The program in a script file is shown below. The program first calculates the pay by multiplying the number of hours by the hourly wage. Then an `if` statement checks whether the number of hours is greater than 40. If so, the next line is executed and the extra pay for the hours above 40 is added. If not, the program skips to the `end`.

```
t=input('Please enter the number of hours worked  ');
h=input('Please enter the hourly wage in $  ');
Pay=t*h;
if t>40
```

```
    Pay=Pay+(t-40)*0.5*h;
end
fprintf('The worker''s pay is  $ %5.2f',Pay)
```

Application of the program (in the Command Window) for two cases is shown below (the file was saved as Workerpay):

```
>> Workerpay
Please enter the number of hours worked  35
Please enter the hourly wage in $  8
The worker's pay is  $ 280.00
>> Workerpay
Please enter the number of hours worked  50
Please enter the hourly wage in $  10
The worker's pay is  $ 550.00
```

6.2.2 *The* `if-else-end` *Structure*

The `if-else-end` structure provides a means for choosing one group of commands, out of a possible two groups, for execution. The `if-else-end` structure is shown in Figure 6-2. The figure shows how the commands are typed in the program, and includes a flowchart that illustrates the flow, or the sequence,

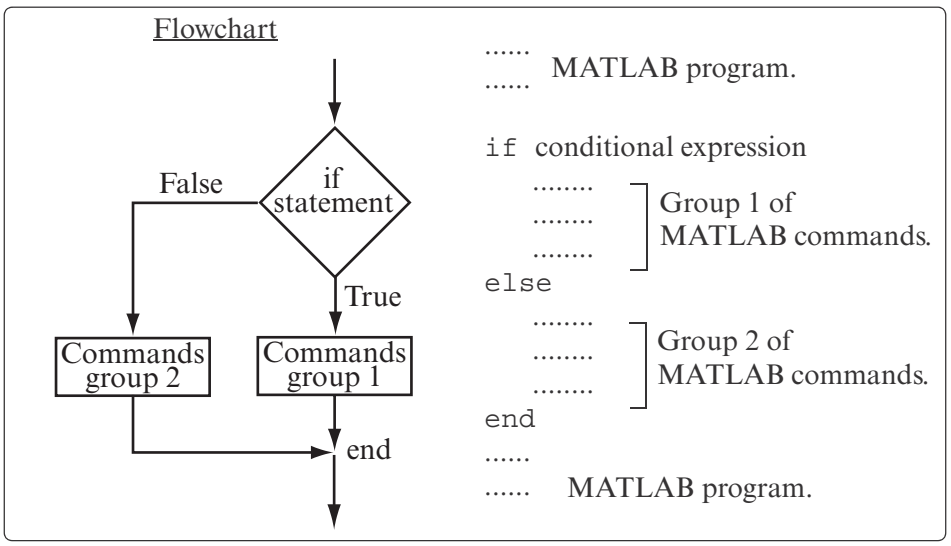


Figure 6-2: The structure of the `if-else-end` conditional statement.

in which the commands are executed. 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 `else` statements and then skips to the `end`. If the conditional expression is false, the program skips to the `else` and then executes group 2 of commands between the `else` and the `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

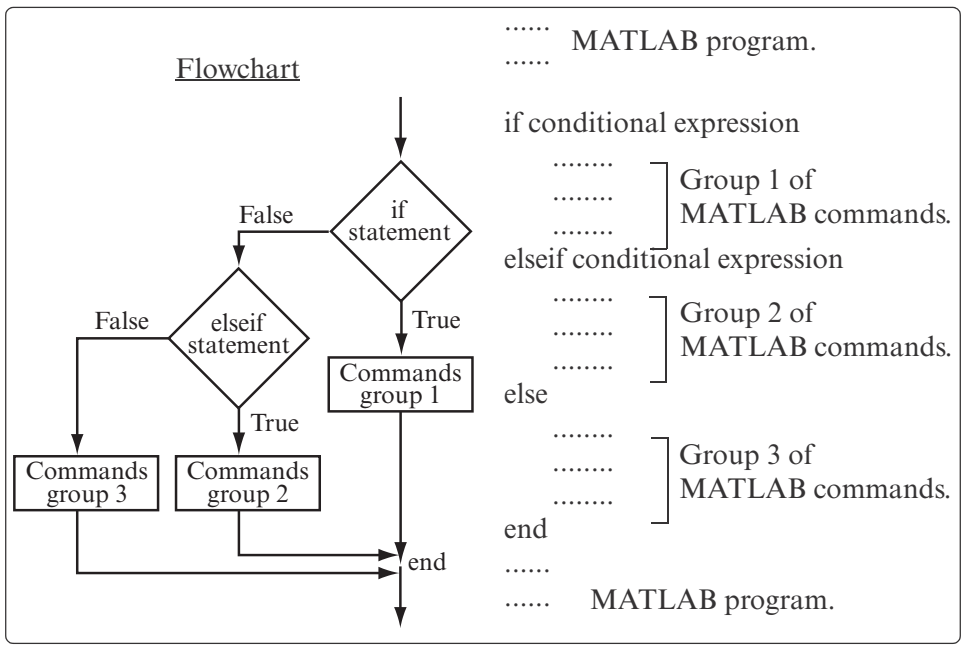


Figure 6-3: The structure of the `if-elseif-else-end` conditional statement.

`elseif` statements and then skips to the `end`. If the conditional expression in the `if` statement is false, the program skips to the `elseif` statement. If the conditional expression in the `elseif` statement is true, the program executes group 2 of commands between the `elseif` and the `else` and then skips to the `end`. If the conditional expression in the `elseif` statement is false, the program skips to the `else` and executes group 3 of commands between the `else` and the `end`.

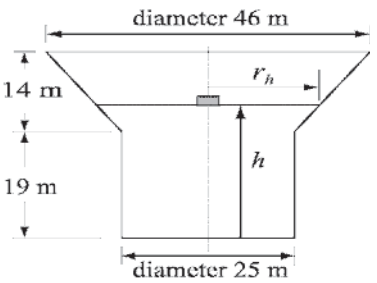
It should be pointed out here that several `elseif` statements and associ-

ated groups of commands can be added. In this way more conditions can be included. Also, the `else` statement is optional. This means that in the case of several `elseif` statements and no `else` statement, if any of the conditional statements is true the associated commands are executed; otherwise nothing is executed.

The following example uses the `if-elseif-else-end` structure in a program.

Sample Problem 6-3: Water level in water tower

The tank in a water tower has the geometry shown in the figure (the lower part is a cylinder and the upper part is an inverted frustum of a cone). Inside the tank there is a float that indicates the level of the water. Write a MATLAB program that determines the volume of the water in the tank from the position (height h) of the float. The program asks the user to enter a value of h in m, and as output displays the volume of the water in m^3.



Solution

For $0 \le h \le 19$ m the volume of the water is given by the volume of a cylinder with height h: $V = \pi 12.5^2 h$.

For $19 \le h \le 33$ m the volume of the water is given by adding the volume of a cylinder with $h = 19$ m, and the volume of the water in the cone:

$$V = \pi 12.5^2 \cdot 19 + \frac{1}{3}\pi(h-19)(12.5^2 + 12.5 r_h + r_h^2)$$

where $r_h = 12.5 + \frac{10.5}{14}(h-19)$.

The program is:

```
% The program calculates the volume of the water in the
water tower.
h=input('Please enter the height of the float in meter  ');
if h > 33
   disp('ERROR. The height cannot be larger than 33 m.')
elseif h < 0
   disp('ERROR. The height cannot be a negative number.')
elseif h <= 19
   v = pi*12.5^2*h;
   fprintf('The volume of the water is %7.3f cubic meter.\n',v)
```