The file is saved as Exp6_6 and then executed in the Command Window:

```
>> Exp6_6
V =
   10   17  -27    8    0   -7   24   30   40   -6   12    4
-8   16
```

6.4.2 `while-end` Loops

`while-end` loops are used in situations when looping is needed but the number of passes is not known in advance. In `while-end` loops the number of passes is not specified when the looping process starts. Instead, the looping process continues as long as a stated condition is satisfied. The structure of a `while-end` loop is shown in Figure 6-6.

```
while  conditional expression
    ........
    ........        A group of
    ........        MATLAB commands.
end
```

Figure 6-6: The structure of a `while-end` loop.

The first line is a `while` statement that includes a conditional expression. When the program reaches this line the conditional expression is checked. If it is false (0), MATLAB skips to the `end` statement and continues with the program. If the conditional expression is true (1), MATLAB executes the group of commands that follow between the `while` and `end` commands. Then MATLAB jumps back to the `while` command and checks the conditional expression. This looping process continues until the conditional expression is false.

For a `while-end` loop to execute properly:

- The conditional expression in the `while` command must include at least one variable.
- The variables in the conditional expression must have assigned values when MATLAB executes the `while` command for the first time.
- At least one of the variables in the conditional expression must be assigned a new value in the commands that are between the `while` and the `end`. Otherwise, once the looping starts it will never stop, since the conditional expression will remain true.

An example of a simple `while-end` loop is shown in the following program. In this program a variable `x` with an initial value of 1 is doubled in each pass as long as its value is equal to or smaller than 15.

```
x=1                          Initial value of x is 1.
while x<=15                  The next command is executed only if x <= 15.
    x=2*x                    In each pass x doubles.
end
```

When this program is executed the display in the Command Window is:

```
x =                          Initial value of x.
     1
x =
     2
x =                          In each pass x doubles.
     4
x =
     8
x =                          When x = 16, the conditional expression in the
    16                       while command is false and the looping stops.
```

Important note:

When writing a while-end loop, the programmer has to be sure that the variable (or variables) that are in the conditional expression and are assigned new values during the looping process will eventually be assigned values that make the conditional expression in the while command false. Otherwise the looping will continue indefinitely (indefinite loop). In the example above if the conditional expression is changed to x >= 0.5, the looping will continue indefinitely. Such a situation can be avoided by counting the passes and stopping the looping if the number of passes exceeds some large value. This can be done by adding the maximum number of passes to the conditional expression, or by using the break command (Section 6.6).

Since no one is free from making mistakes, a situation of indefinite looping can occur in spite of careful programming. If this happens, the user can stop the execution of an indefinite loop by pressing the **Ctrl + C** or **Ctrl + Break** keys.

Sample Problem 6-7: Taylor series representation of a function

The function $f(x) = e^x$ can be represented in a Taylor series by $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$.

Write a program in a script file that determines e^x by using the Taylor series representation. The program calculates e^x by adding terms of the series and stopping when the absolute value of the term that was added last is smaller than 0.0001. Use a while-end loop, but limit the number of passes to 30. If in the

30th pass the value of the term that is added is not smaller than 0.0001, the program stops and displays a message that more than 30 terms are needed.

Use the program to calculate e^2, e^{-4}, and e^{21}.

Solution

The first few terms of the Taylor series are:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$$

A program that uses the series to calculate the function is shown next. The program asks the user to enter the value of `x`. Then the first term, `an`, is assigned the number 1, and `an` is assigned to the sum `S`. Then, from the second term on, the program uses a `while` loop to calculate the *n*th term of the series and add it to the sum. The program also counts the number of terms `n`. The conditional expression in the `while` command is true as long as the absolute value of the *n*th `an` term is larger than 0.0001, and the number of passes `n` is smaller than 30. This means that if the 30th term is not smaller than 0.0001, the looping stops.

```
x=input('Enter x ' );
n=1; an=1; S=an;
while abs(an) >= 0.0001 & n <= 30          Start of the while loop.
    an=x^n/factorial(n);                   Calculating the nth term.
    S=S+an;                                Adding the nth term to the sum.
    n=n+1;                                 Counting the number of passes.
end                                        End of the while loop.
if n >= 30                                 if-else-end loop.
    disp('More than 30 terms are needed')
else
fprintf('exp(%f) = %f',x,S)
fprintf('\nThe number of terms used is: %i',n)
end
```

The program uses an `if-else-end` statement to display the results. If the looping stopped because the 30th term is not smaller than 0.0001, it displays a message indicating this. If the value of the function is calculated successfully, it displays the value of the function and the number of terms used. When the program executes, the number of passes depends on the value of `x`. The program (saved as `expox`) is used to calculate e^2, e^{-4}, and e^{21}:

```
>> expox
Enter x 2                                  Calculating exp(2).
```